CRYPTOGRAPHIE ET CLOUD COMPUTING – ÉTAT DE L'ART

TANIA MARTIN

Résumé – Impossible de nier les avantages du cloud : réduction des coûts, plus grande évolutivité, meilleure mobilité, déploiement plus rapide et mises à niveau instantanées, pour ne citer qu'eux. Cependant, les risques de sécurité constituent toujours le principal frein à l'adoption du cloud par les organisations et entreprises. L'objectif de cette research note est de présenter les mécanismes cryptographiques connus et existants qui permettent protéger le stockage et le traitement de données dans les environnements cloud.

Abstract – Het is onmogelijk om de voordelen van de cloud te ontkennen. Om er slechts enkele te noemen: vermindering van de kosten, een grotere evolutiviteit, een betere mobiliteit, een sneller deployment en instant updates. De veiligheidsrisico's vormen echter nog steeds de voornaamste reden waarom organisaties en bedrijven de cloud niet volledig omarmen. Deze research note heeft als doel de gekende en bestaande cryptografische mechanismen voor te stellen waarmee de gegevensopslag en - verwerking beschermd kunnen worden in de cloud-omgevingen.

Table des matières

1. Contexte

Devenu une notion inévitable ces dernières années, le cloud computing est passé d'un concept brumeux à la stratégie ICT « du futur » vers laquelle toute organisation va tôt ou tard se tourner. La raison majeure de cet engouement est la facilité d'accès à un parc de ressources informatiques au potentiel presque illimité, tout cela avec un effort minimal de gestion. Une organisation peut ainsi louer les ressources partagées d'un service cloud, devenant alors locataire (appelé *tenant* en anglais) d'infrastructure plutôt que propriétaire. Le partage de telles infrastructures est malheureusement le talon d'Achille des services cloud. En effet, des cyber-attaques ont montré la possibilité d'exploiter la proximité de partage des tenants d'un même service cloud. Les problèmes de sécurité, en particulier de confidentialité et intégrité des données, sont donc la préoccupation majeure des utilisateurs du cloud car leurs données se retrouvent gérées hors du cadre de leur gouvernance. Dans le contexte gouvernemental, de sécurité sociale et soins de santé, ces problèmes sont d'autant plus accrus car ils concernent potentiellement les données sensibles des citoyens et entreprises.

En 2014, l'étude « Cloud Security Guidance » de la section Recherche de Smals avait brièvement étudié diverses techniques cryptographiques intervenant dans la sécurisation des données dans le cloud. L'objectif de cette présente publication est de détailler davantage les mécanismes cryptographiques connus et existants qui s'appliquent aux environnements cloud pour protéger leurs deux applications les plus connues : le stockage et le traitement sur les données.

2. Conditions d'analyse du cloud

Avant d'expliquer en détail les différentes méthodes pour protéger les données envoyées dans un service cloud, il est d'abord important de définir les conditions (architecture, attaquants, propriétés de sécurité) avec lesquelles un tel service est analysé.

2.1. Architecture du cloud

L'architecture du cloud considérée dans cette publication est la suivante. D'un côté se trouve un utilisateur quelconque que nous appellerons Bob, faisant partie d'une entreprise quelconque. De l'autre côté se trouve un service cloud dont Bob se sert pour stocker des données ou effectuer des traitements. Pour ce faire, Bob peut soit être en contact direct avec le service cloud, soit faire appel à une entité intermédiaire, telle qu'un tiers de confiance (*Trusted Third Party* en anglais, TTP) ou qu'une passerelle de sécurité cloud (*Cloud Security Gateway* en anglais, CSG).

La figure ci-dessous introduit les icones qui seront utilisées dans cette publication pour représenter ces différentes entités.

Icone					
Entité	Service cloud	Bob	Entreprise	TTP	CSG

2.2. Attaquants

Cette publication considère un attaquant, que nous appellerons Oscar par la suite, qui possède les capacités suivantes.

- Oscar est capable d'écouter les canaux de communication entre les différentes entités de l'architecture.
- Oscar est capable d'attaquer le service cloud et ainsi obtenir toutes les données stockées chez ce dernier.

2.3. Propriétés de sécurité

Pour comprendre les protections offertes par la cryptographie dans le cloud, il faut considérer la sécurité par rapport à ses trois objectifs.

Confidentialité : Toute donnée sensible reste inintelligible pour tout adversaire ou entité qui n'est pas de confiance.

Intégrité : Toute altération non autorisée (fortuite, illicite ou malveillante) des données doit être détectable.

Disponibilité : Les propriétaires des données sont assurés de toujours avoir accès à leurs données.

Etant donné que la disponibilité est généralement résolue dans les environnements de cloud computing d'aujourd'hui par des moyens non-cryptographiques, nous nous concentrons par la suite uniquement sur la confidentialité et l'intégrité dans le cloud.

La **confidentialité** peut être assurée par du chiffrement cryptographique. En effet, une fois que des données sont chiffrées, elles ne « signifient » en soi plus rien. La clé qui a été utilisée pour faire ce chiffrement est la seule information qui peut aider à recouvrir les données en clair.

L'**intégrité** peut être, quant à elle, assurée par une fonction de hachage ou une signature digitale. En effet, une fois que des données originales spécifiques sont hachées ou signées digitalement, une altération des données originales peut être détectée car la version hachée/signée des données ne correspond plus à la version originale.

3. Stockage protégé dans le cloud

Prisé tant par les particuliers que par les entreprises, le stockage de données est une des utilisations premières du cloud. Cela sollicite des capacités de stockage « à la demande » et accessibles depuis divers lieux (p. ex. à partir d'appareils mobiles). Cela demande également des équipements de sauvegarde, et des capacités de synchronisation permettant aux clients d'avoir toujours accès à la dernière version de leurs données indépendamment de l'appareil utilisé (PC, smartphone ou encore tablette), et tout ceci sans la nécessité d'entretenir du matériel hardware ou de maintenir à jour des outils logiciels.

3.1. Exemples d'utilisation

Les cas d'utilisation suivants présentent les avantages majeurs de ces services de stockage dans le cloud.

► Copie

Prenons l'exemple de Charles, un commercial travaillant la plupart de son temps à l'extérieur. Le mois passé, il a voyagé pendant deux semaines en Europe pour démarcher de nouveaux clients. Durant ces deux semaines, il a pris beaucoup de notes et commencé à rédiger quelques esquisses pour des projets potentiels. Son voyage d'affaire a été très productif, mais il s'est malheureusement fait voler la valise contenant son laptop pendant qu'il attendait son vol de retour. Charles ne se soucie pas de la fuite des données accumulées durant les deux semaines, car il utilise toujours un disque dur chiffré ; il est plutôt triste car il a perdu tout son travail et tous ses résultats obtenus pendant ces deux semaines.

En utilisant la fonctionnalité de copie dans le cloud, Charles n'aurait finalement pas eu de problème : son laptop aurait envoyé dans le cloud la moindre copie de chaque modification de document ou de chaque création de nouveau document. Une fois rentré de voyage, Charles aurait pu récupérer toutes ses données sur un nouveau laptop.

► Versioning

Prenons l'exemple de Bernard, qui est à la tête d'un cabinet d'architectes. Ces jours-ci, il recherche désespérément le plan d'un bâtiment qu'il a dessiné l'année dernière. C'est la future maison d'un client qui change d'avis comme de chemises, forçant Bernard à faire beaucoup de modifications, et ceci de façon quasi hebdomadaire. La semaine dernière, le client a demandé une refonte complète de la maison en partant du plan du 25 juin 2014. Depuis lors, Bernard cherche la version numérique de ce plan. Malgré qu'il fasse des copies de ses données, même sur des disques externes, Bernard n'arrive pas à retomber sur cette version en particulier.

En utilisant la fonctionnalité versioning d'un service cloud, qui garderait toutes les versions de tous ses fichiers, Bernard aurait été capable de retrouver le plan du bâtiment de n'importe quelle date.

► Synchronisation

Prenons l'exemple de Sophie, une chef de projet qui a travaillé toute la journée d'hier à son bureau pour une présentation qu'elle donne aujourd'hui. Dans le train qui l'amène au lieu de sa présentation, Sophie souhaite améliorer ses slides et peaufiner sa présentation. Or elle se rend compte qu'elle n'a pas la dernière version des slides sur son laptop. Et pour cause : à son bureau, elle travaille sur un desktop PC, et elle a uniquement copié la version de midi de la présentation sur sa clé USB (et non pas la dernière version de fin de journée).

En utilisant la fonctionnalité de synchronisation d'un service de stockage dans le cloud, Sophie aurait pu éviter ce problème. En effet, la synchronisation permet de connecter plusieurs appareils entre eux. Ainsi son desktop PC aurait envoyé dans le cloud chaque modification effectuée par Sophie, et son laptop aurait pu récupérer, en temps réel, toutes ces modifications. La synchronisation détecte et gère également les conflits de version (p. ex. quand un fichier est modifié en parallèle sur deux appareils).

► Partage

Prenons l'exemple de Patricia, une photographe indépendante. Elle est en train d'écrire un guide touristique avec Phil, un écrivain talentueux. Ils travaillent ensemble via email, mais ils sont tous les deux mécontents de cette méthode : le guide étant composé de plus de 100 fichiers, il est trop lourd pour être envoyé par email et les deux auteurs s'arrachent les cheveux pour maintenir la synchronisation des fichiers sur leurs ordinateurs respectifs.

En utilisant la fonctionnalité de partage d'un service cloud, Patricia et Phil pourraient simplement éditer leur travail, et chaque ordinateur copierait le résultat immédiatement dans le cloud. Ainsi tous les ordinateurs connectés à ce dossier de partage téléchargeraient automatiquement des copies des fichiers modifiés : Patricial et Phil seraient tout le temps synchronisés, sans aucun effort. Notez que cela peut générer des conflits de version, mais c'est généralement très bien géré par les services cloud.

La suite de cette section détaille les techniques cryptographiques les plus connues et efficaces pour protéger les données stockées dans le cloud.

3.2. Chiffrement classique

Cette section couvre le chiffrement dit classique dans sa globalité, tant le chiffrement symétrique tel que l'AES[1] que le chiffrement asymétrique tel que le cryptosystème RSA[2]. On considère que les algorithmes cryptographiques utilisés pour le chiffrement classique sont dits « forts », c'est-à-dire avoir été examinés par des experts, agences

[1] *Advanced Encryption Standard*.

[2] Nommé par les initiales de ses trois inventeurs : Rivest, Shamir et Adleman.

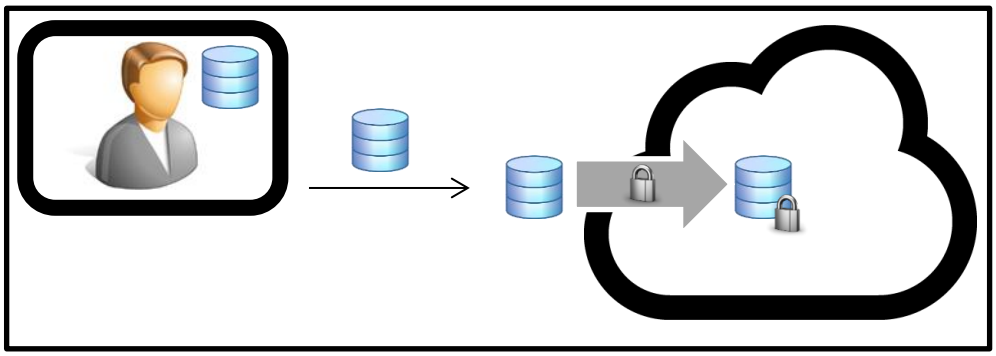
gouvernementales et organisations tierces de sécurité, et reconnus comme très difficiles (pour ne pas dire impossible) à casser.

Cette section prend également en compte le fait qu'une fonction de hachage ou une signature digitale peut aussi être appliquée sur les données (en plus du chiffrement).

Il est important de séparer deux cas bien particuliers d'utilisation du chiffrement classique : celui côté serveur et celui côté client.

3.2.1. Chiffrement côté serveur

Cette technique de chiffrement est la plus répandue actuellement chez les services cloud. Comme illustré dans la figure ci-dessous, l'utilisateur Bob envoie ses données « en clair » à stocker au service cloud. C'est le service cloud qui se charge de chiffrer les données avant leur stockage. Le service cloud peut aussi éventuellement appliquer une fonction de hachage ou une signature digitale aux données à stocker.



Dans cette configuration-là, la gestion des clés utilisées pour le chiffrement doit obligatoirement se faire côté serveur, où deux cas distincts sont possibles.

- Stockage non-sécurisé des clés

  Le premier cas est la pire situation envisageable : les clés sont stockées de façon non sécurisée au même endroit que les données chiffrées de Bob. Si Oscar est capable de s'emparer des données chiffrées, alors il est certainement aussi capable de s'emparer des clés, et peut ainsi retrouver les données en clair de Bob.

  ↘ **La confidentialité des données vis-à-vis d'Oscar n'est pas assurée.**
  ↘ **L'intégrité des données vis-à-vis d'Oscar n'est pas assurée non plus si les clés utilisées pour la fonction de hachage ou signature digitale ne sont également pas protégées.**

- Stockage sécurisé des clés

  L'autre situation envisageable est de stocker les clés de façon sécurisée, que ce soit au même endroit ou pas que les données chiffrées de Bob. Une technique consiste en l'utilisation d'un HSM[3], (cf. exemple en figure ci-contre), appareil électronique considéré comme inviolable où il est possible de générer/stocker/protéger des clés et d'effectuer des opérations cryptographiques avec ces dernières. Les HSM répondent à des standards de sécurité élevés (p. ex. Critères Communs EAL4+) et représentent donc un gage respectable de sécurité. Une autre technique consiste en l'utilisation d'un *keystore* logiciel chiffré, c.-à-d. un répertoire électronique contenant les clés secrètes et protégé par un mot de passe ou tout autre mécanisme cryptographique, tel que le standard PKCS#12[4]. Avec ces deux techniques de protection, même si Oscar s'empare des données chiffrées de Bob et du HSM ou keystore, il est dans l'incapacité de recouvrir les clés, donc les données en clair de Bob.

↗ **La confidentialité des données vis-à-vis d'Oscar est donc assurée.**
↗ **L'intégrité des données vis-à-vis d'Oscar est aussi assurée si les clés utilisées pour la fonction de hachage ou signature digitale sont également protégées.**

Notons que peu importe l'entité qui gère les clés (le service cloud, l'utilisateur, ou une TTP) et peu importe la méthode de chiffrement utilisée (symétrique ou asymétrique), le niveau de sécurité reste identique pour chaque cas de figure présenté ci-dessus.

De plus, il n'y a **pas de confidentialité ni d'intégrité vis-à-vis du service cloud** dans tous les cas. En effet, que les clés soient dans un environnement protégé ou pas, étant donné que les opérations cryptographiques se font côté serveur (chez le service cloud), le service cloud « voit » nécessairement les données en clair avant leur traitement.

3.2.2. Chiffrement côté client

Cette autre technique de chiffrement est moins répandue, malgré qu'elle permette à l'utilisateur d'avoir un contrôle complet sur ses données. Comme illustré dans la figure ci-dessous, l'utilisateur Bob chiffre ses données de son côté, puis envoie ses données chiffrées au service cloud pour stockage. Bob peut aussi éventuellement appliquer une fonction de

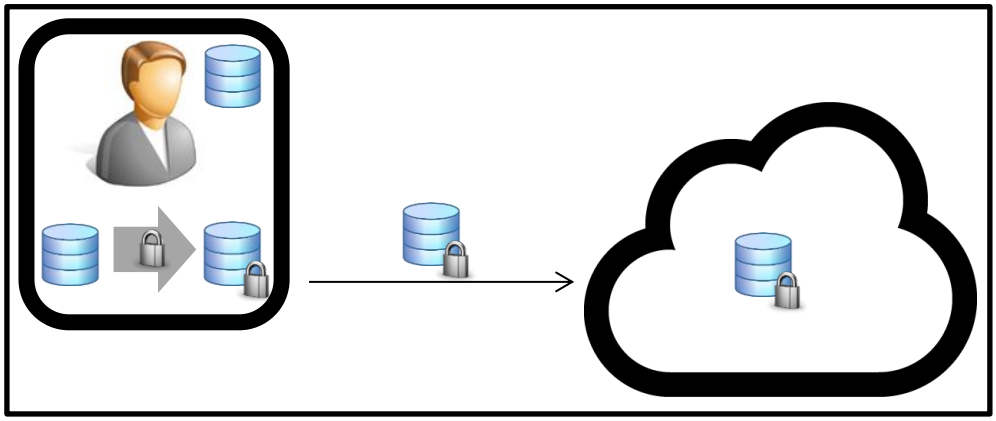
[3] *Hardware Security Module*.

[4] https://tools.ietf.org/html/rfc7292

hachage ou une signature digitale à ses données avant de les envoyer au service cloud.

Remarque : Même si les clés se trouvent côté client, et ne tombent pas dans les mains de notre adversaire type Oscar, il n'est pas exclu de les protéger localement.

Dans cette configuration-là, la gestion des clés cryptographiques doit se faire côté client (c.-à-d. chez l'utilisateur) pour que l'utilisateur garde le maximum de contrôle sur ses données. Le service cloud ne connait donc ni les clés, ni les données en clair : il ne « voit » que les données chiffrées.



↗ **La confidentialité et l'intégrité des données vis-à-vis du service cloud et d'Oscar sont donc assurées.**

Si l'utilisateur est l'unique entité qui gère ses clés, alors cela peut devenir dangereux. En effet, si Bob perd ou oublie ses clés, les données chiffrées sont indéchiffrables et les données en clair sont perdues définitivement. Une solution à ce problème est que les clés soient gérées par une TTP qui ait un mécanisme de *key escrow* : cela permet, dans certaines circonstances extrêmes ou graves, à la TTP de retrouver les clés perdues ou oubliées et ainsi de réobtenir les données en clair. Bien sûr, c'est à utiliser avec parcimonie.

Un autre point faible est que les clés doivent être installées sur tous les appareils du client, sinon Bob sera incapable de déchiffrer ses données à partir de ses différents appareils.

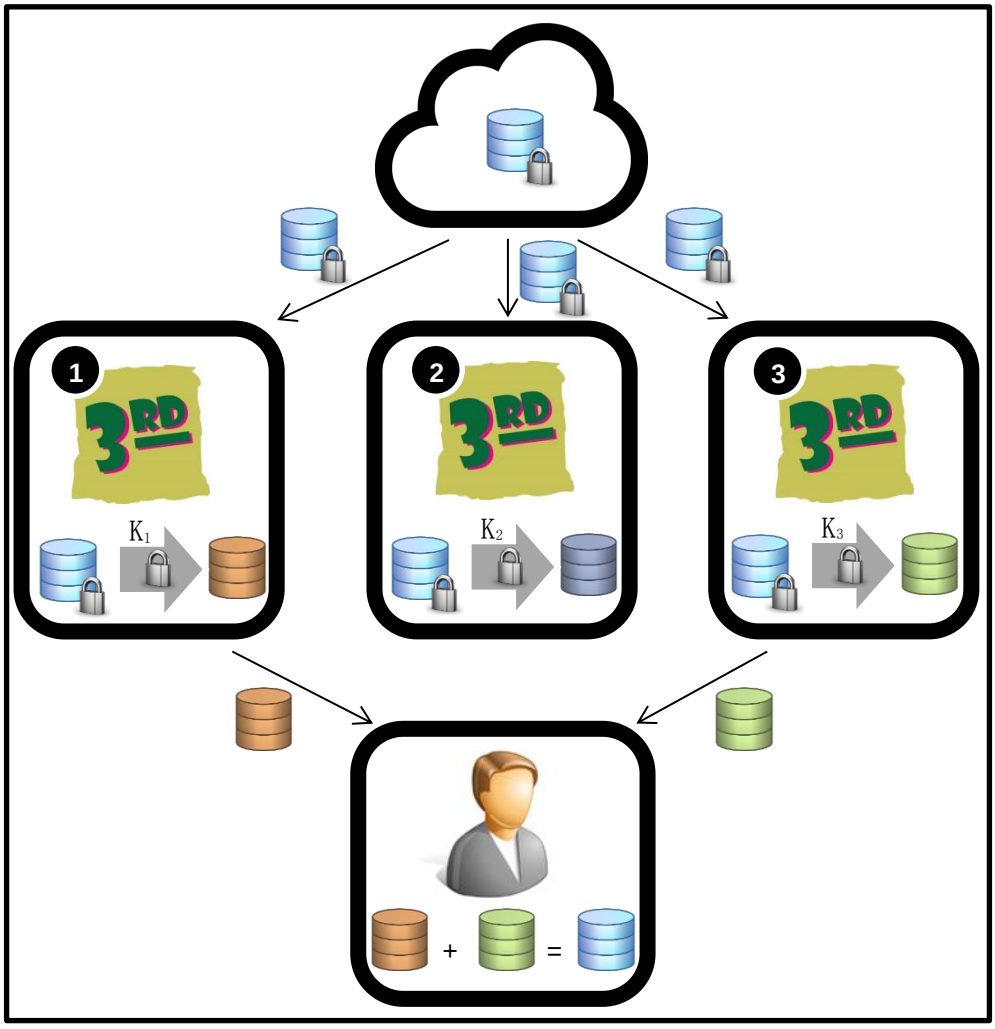
Un exemple de produit existant fournissant un chiffrement classique côté client est BoxCryptor[5] (en combinaison avec un service cloud).

[5] https://www.boxcryptor.com/en. Pour le lecteur intéressé, un quick review du produit a été effectué par la section Recherche de Smals : https://www.smalsresearch.be/publications/document/?docid=117.

3.3. Chiffrement threshold

Une autre méthode pour stocker des données de façon sécurisée dans le cloud est d'utiliser le chiffrement *threshold*[6]. Pour le définir, on peut dire que c'est la rencontre entre un système de chiffrement à clé publique classique et un système de partage de secrets.

Le but du chiffrement threshold est de partager un secret entre n entités, chaque entité ayant seulement une partie du secret, et seulement t entités (telles que $n \geqslant t \geqslant 2$) doivent partager leur connaissance pour reconstruire le secret. Etant basé sur du chiffrement à clé publique, le secret est une clé privée K correspondante à une clé publique P. Les parties du secret distribuée aux n entités sont les n sous-clés privées $(K_1, K_2, \cdots, K_n)$.



[6] Seuil en français.

L'application dans le cadre du stockage protégé de données dans le cloud est la suivante. La clé P est une information publique, et est donc utilisée par Bob pour chiffrer une donnée qu'il souhaite envoyer dans le cloud : le chiffrement[7] se fait donc côté client. Chaque TTP dispose d'une sous-clé privée obtenue lors de l'initialisation du système. Ainsi, lorsque Bob souhaite récupérer une donnée (et qu'il en a le droit), il demande à t TTP de collaborer pour déchiffrer la donnée demandée par Bob. N'importe quel ensemble de t TTP peuvent se réunir pour déchiffrer un message ; mais si moins de TTP collaborent, la donnée ne peut pas être retrouvée.

Dans l'exemple illustré ci-dessus, $n=3$ et $t=2$. Les trois TTP participant au chiffrement threshold reçoivent les données chiffrées de Bob (en bleu sur la figure) du service cloud. Elles peuvent ensuite déchiffrer une sous-partie de ces données. Mais chaque sous-partie déchiffrée par une TTP n'a aucune signification seule : les TTP sont incapables de récupérer seules les données originales car la méthode utilisée pour le chiffrement les bloque dans cette opération. Même si Oscar, après avoir compromis le service cloud, s'allie avec une TTP malintentionnée, il est incapable de récupérer les données originales de Bob. Par contre, Bob est lui capable de récupérer ses données si au moins deux TTP (ici la n°1 et la n°3) lui fournissent leur sous-partie.

Remarque : Même si les clés se trouvent côté client, et ne tombent pas dans les mains de notre adversaire type Oscar, il n'est pas exclu de les protéger localement.

↗ **Comme les opérations cryptographiques se font côté client, la confidentialité et l'intégrité des données vis-à-vis du service cloud, d'Oscar et d'une TTP sont assurées quand $n \geqslant t \geqslant 2$.**

Pour plus de détails sur le chiffrement threshold, le lecteur intéressé peut consulter la publication[8] de la section Recherche de Smals ou encore le projet Vitalink[9], un exemple concret belge utilisant le chiffrement threshold, développé par Smals de le contexte des soins de santé.

vitalink
gegevens delen, van vitaal belang

3.4. Preuve de stockage, preuve de récupérabilité

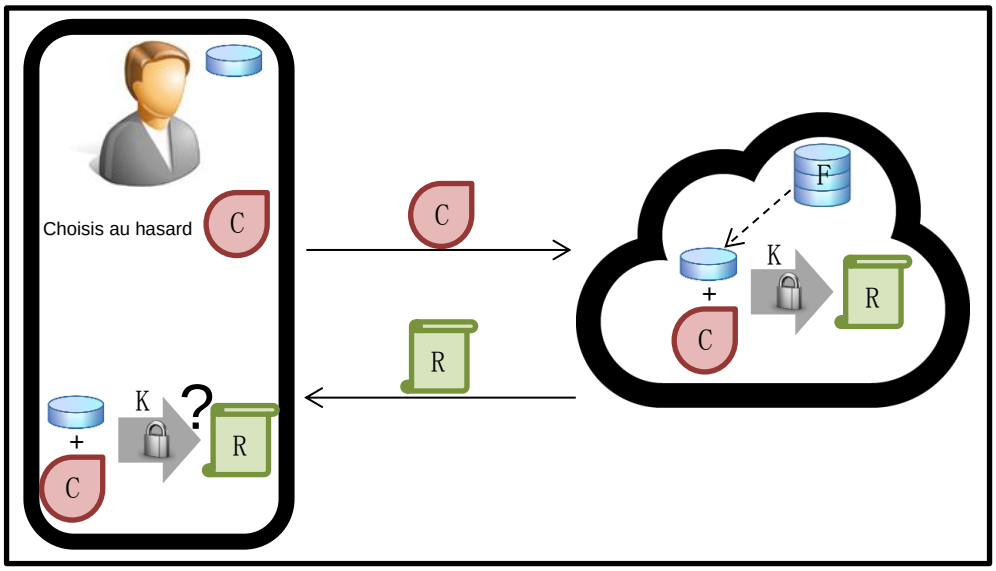
Dans une autre veine, la preuve de stockage ou preuve de récupérabilité permet à un utilisateur de stocker un fichier F au sein d'un service cloud, et de s'assurer plus tard que le service cloud possède toujours le fichier F et qu'il est capable de le récupérer. C'est donc une façon de s'assurer de l'**intégrité** de ce fichier F sans avoir à le télécharger pour vérification. Ce

[7] Bob peut aussi éventuellement appliquer une fonction de hachage ou une signature digitale à ses données avant de les envoyer au service cloud.

[8] https://www.smalsresearch.be/publications/document/?docid=60

[9] http://www.vitalink.be/VitaStart.aspx/

concept a été introduit par Juels et Kaliski dans un article[10] publié à la conférence ACM CCS 2007.



La figure ci-dessus est une illustration de ce type de preuve. Pour s'assurer que le service cloud possède toujours le fichier F que Bob y a transféré, Bob et le service cloud effectuent un « protocole de challenge-réponse ». Bob (le vérifieur) envoie un challenge/question C au service cloud (le prouveur). Ce dernier calcule ensuite une réponse/preuve R qui dépend du challenge C, mais aussi d'une valeur représentative de F connue de Bob et d'une clé secrète K partagée entre Bob et le service cloud[11]. A la réception de la preuve R, Bob vérifie si elle est correcte.

↘ **Comme il n'y a pas de chiffrement des données, aucune confidentialité n'est assurée par ce type de méthode.**

↗ **Comme le calcul de la preuve se fait côté serveur, l'intégrité des données vis-à-vis d'Oscar n'est assurée que si la clé secrète K est protégée.**

Même si la preuve de stockage est une méthodologie reconnue en théorie, il semblerait qu'elle n'ait pas été implémentée en pratique.

[10] http://www.arijuels.com/wp-content/uploads/2013/09/JK07.pdf

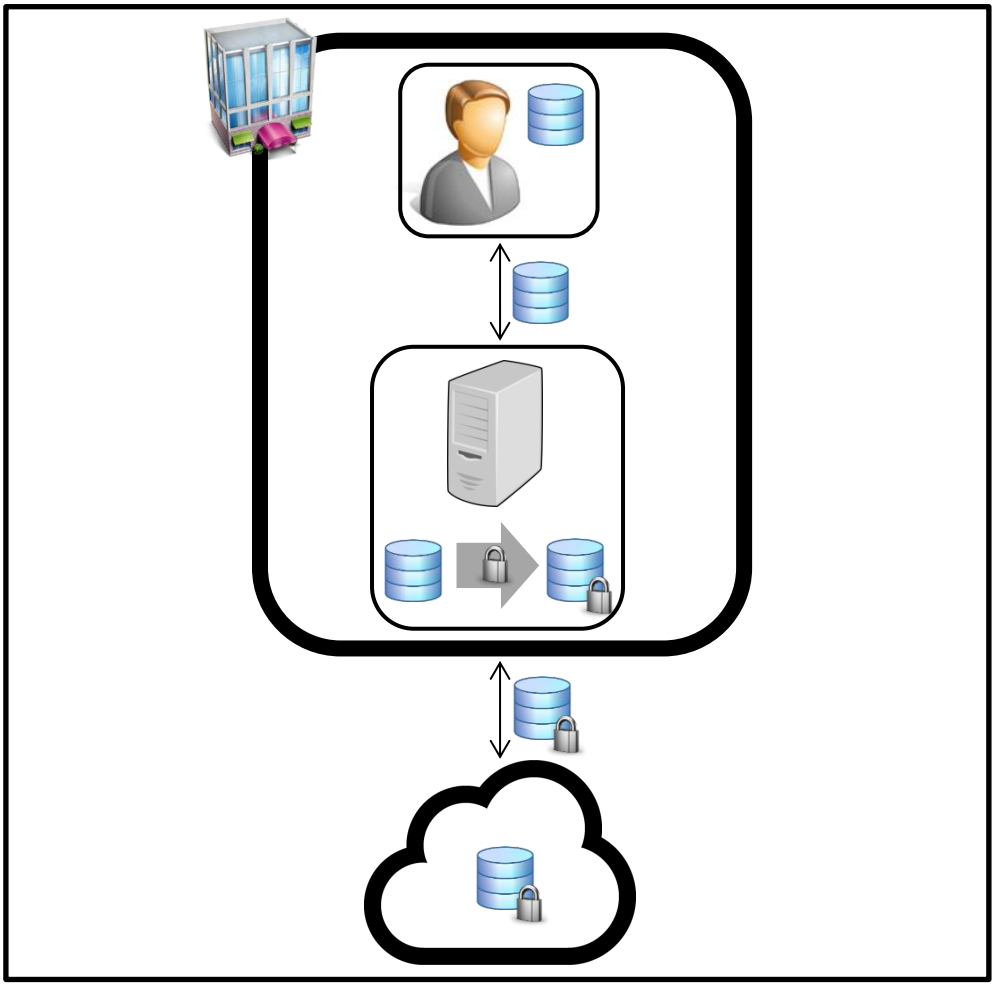
[11] Notez que K peut aussi représenter une paire de clés publique/privée.

3.5. Cloud security gateway

Enfin une autre technique pour protéger les données stockées dans le cloud est l'utilisation d'une *Cloud Security Gateway* (CSG). Cette technique a déjà fait l'objet de plusieurs publications[12] de la part de la section Recherche de Smals en 2012.

Une CSG est un outil qui se trouve entre un utilisateur et un service cloud, généralement au sein de l'entreprise de l'utilisateur dans le cadre professionnel. C'est un intermédiaire de confiance du point de vue sécurité, communément sous forme de proxy, qui chiffre/déchiffre les données qui transitent par elle. Comme illustré dans la figure ci-dessous, son utilisation est assez directe : l'utilisateur Bob envoie ses données en clair à la CSG, et celle-ci chiffre ces données puis les envoie au service cloud.

Remarque : Même si les clés se trouvent côté client, et ne tombent pas dans les mains de notre adversaire type Oscar, il n'est pas exclu de les protéger localement.



[12] Management summary et présentation https://www.smalsresearch.be/publications/document/?docid=26 https://www.smalsresearch.be/publications/document/?docid=27

Dans cette configuration-là, la gestion des clés cryptographiques doit se faire côté client (c.-à-d. chez la CSG), pour que l'entreprise garde le maximum de contrôle sur les données de Bob. Le service cloud ne connait donc ni les clés, ni les données en clair : il ne « voit » que les données chiffrées.

↗ **Comme les opérations cryptographiques se font côté client, la confidentialité et l'intégrité des données vis-à-vis du service cloud et d'Oscar sont assurées.**

Dans le marché émergent mais déjà bien implanté des CSG, on peut trouver des solutions proposées par CipherCloud, Netskope, Perspecsys, Protegrity, Skyhigh Networks ou encore Vaultive. Chaque fournisseur de CSG propose plusieurs produits, chacun étant spécialement conçu pour un service de stockage bien précis : par exemple Netskope et Skyhigh Networks proposent des produits pour Box, Dropbox et Google Drive.

La section 4.5 montrera que les CSG sont également utilisables quand on souhaite faire du traitement protégé dans le cloud.

3.6. Conclusion du stockage protégé

Finalement, le chiffrement classique côté client est la meilleure protection pour le stockage de données dans le cloud : la confidentialité et l'intégrité des données y sont assurées vis-à-vis d'Oscar et même du service cloud. On retrouve souvent mentionné dans la littérature que l'inconvénient majeur de cette méthode est que toutes les opérations cryptographiques se font chez le client, ce qui est susceptible d'impacter les performances de rapidité en termes de chiffrement/déchiffrement des données. C'est un point à nuancer : on a pu constater qu'en pratique l'impact était négligeable. Par contre, un réel inconvénient déjà évoqué est que Bob est seul responsable des clés cryptographiques : s'il les perd/oublie, alors il se peut que ses données chiffrées soient irrécupérables.

L'utilisation d'une CSG est aussi une très bonne protection. L'avantage par rapport au chiffrement côté client est que les opérations cryptographiques ne se sont plus faites par Bob, mais par la CSG, allégeant la charge de Bob. Un autre avantage est que la CSG devient le responsable des clés de chiffrement, ce qui allège la responsabilité de Bob. Dans ce cas-là, les données en clair restent dans le périmètre de l'entreprise, ce qui peut être vu comme un désavantage : Bob pourrait souhaiter que ses données soient strictement confidentielles, par exemple aussi vis-à-vis des autres employés de l'entreprise. Un autre inconvénient d'une CSG est que cette dernière pourrait être un point unique de défaillance (*Single Point of Failure* en anglais, SPOF) : si la CSG tombe en panne et que l'architecture de l'entreprise ne tient pas compte de la notion de SPOF, alors cela pourrait paralyser **tous les utilisateurs** de l'entreprise (et pas seulement Bob). Enfin le plus grand inconvénient est qu'une CSG doit être configurée pour un ou plusieurs services cloud

désirés (chaque service cloud étant lui-même un peu unique en termes de fonctionnement).

Le choix entre un chiffrement côté client ou une CSG doit donc être bien clairement étudié suivant le cas précis de souhait d'utilisation.

4. Traitement protégé dans le cloud

L'autre grande utilisation prisée par les particuliers et entreprises est le traitement de données dans le cloud. Accessibles également « à la demande » et depuis n'importe où, cela peut parfois solliciter de très grandes capacités de traitement, tout cela sans le besoin du client d'investir dans des équipements couteux. De nombreux services cloud SaaS[13], PaaS[14] et IaaS[15] sont basés sur le traitement de données.

4.1. Exemples d'utilisation

Les cas d'utilisation suivants présentent les avantages majeurs de ces services de traitement dans le cloud.

 ► Email

Prenons l'exemple d'Emilie, une chef de projet qui se tient au courant des avancées de son équipe à toute heure du jour et de la nuit. Le mois dernier, elle est partie en vacances « sac-à-dos » pendant 3 semaines et a dû amener son laptop du bureau (très encombrant malgré tout) pour pouvoir accéder à ses emails via Outlook.

En utilisant un service d'email dans le cloud, Emilie n'aurait pas eu besoin d'alourdir ses bagages avec un laptop. Elle aurait pu aisément consulter ses emails professionnels au travers des navigateurs web d'autres ordinateurs simplement connectés à Internet, par exemple à ses hôtels de passage, dans un cyber-café, ou encore dans une bibliothèque publique.

 ► Création d'apps

Prenons l'exemple d'Alexis, un développeur de logiciels sous-payé ayant une vie sociale bien remplie. Dernièrement, il a commencé à développer, durant le peu de temps libre lui restant, un jeu en ligne promis à un grand avenir. Avant même commencer son développement, il a dû investir dans des serveurs (difficiles à caser dans son petit appartement) et apprendre à les maintenir en état de fonctionnement optimal et à les sécuriser.

En utilisant un service de création d'apps dans le cloud, Alexis n'aurait pas dû se soucier de l'infrastructure, du matériel, du middleware sous-jacent,

[13] *Software as a Service*.

[14] *Platform as a Service*. Pour plus de détails, le lecteur peut consulter la research note de Smals https://www.smalsresearch.be/publications/document/?docid=100.

[15] *Infrastructure as a Service*.

ni même les entretenir. Le service cloud lui aurait fourni une installation prête à l'emploi en moins de 10 minutes, et Alexis aurait pu commencer à développer son jeu en ligne immédiatement.

► Calcul numérique intensif

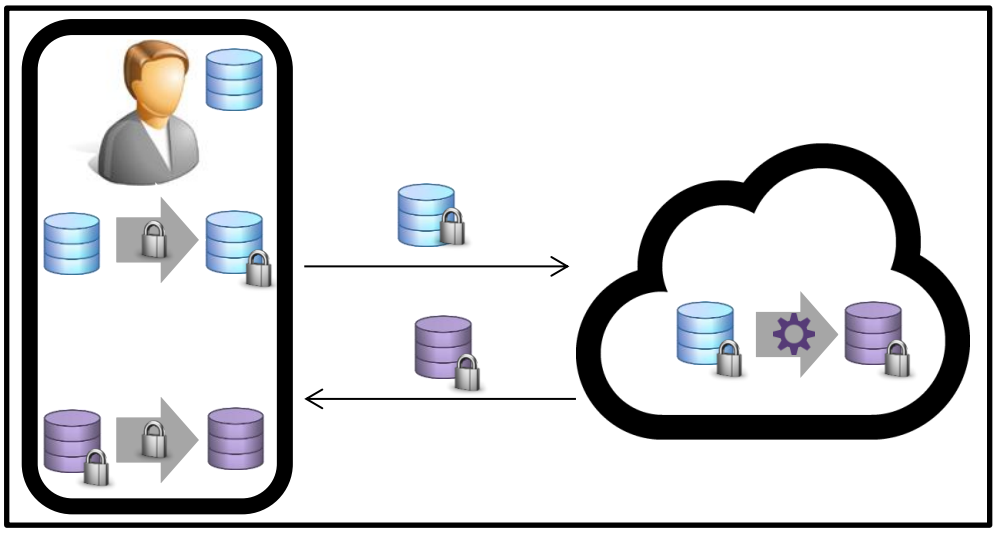
Prenons l'exemple de Chris, chercheur universitaire dans un département d'informatique dont le budget est très limité. Le département a déjà investi, il y a quelques années, dans des serveurs qui sont partagés par tout le département et qui ne peuvent donc être utilisés qu'avec modération par les chercheurs. Pour les dernières simulations de sa thèse de doctorat, Chris a dû donc faire attention à l'utilisation des serveurs en optimisant au mieux le code de ses tests.

En utilisant un service de calcul numérique intensif (dit HPC, *High Performance Computing* en anglais), Chris aurait pu s'épargner tout ce travail supplémentaire sur ses tests. En effet, des offres très intéressantes et peu chères dans le cloud permettent de s'approvisionner en un clic en instances de serveurs HPC.

La suite de cette section détaille les techniques cryptographiques les plus connues et efficaces pour protéger les traitements de données dans le cloud.

4.2. Chiffrement homomorphique

Le problème avec le chiffrement classique de données est que, tôt ou tard, il faut déchiffrer les données si l'on souhaite les utiliser, que ce soit pour éditer un document Word ou interroger une base de données financière. Et c'est à ce moment-là que les données deviennent vulnérables à des attaques potentielles. Bien qu'encore majoritairement théorique à l'heure actuelle, le chiffrement homomorphique est la technique la plus célèbre pour résoudre ce problème et faire du traitement protégé dans le cloud.



En réalité, le chiffrement homomorphique permet d'effectuer des traitements sur des données sans avoir besoin de les déchiffrer. La figure ci-dessus illustre grossièrement comment la technique peut être utilisée avec un service cloud lorsque Bob souhaite faire de lourds traitements sur sa base de données dans un service cloud. Tout d'abord, Bob chiffre homomorphiquement sa base de données (en bleu sur la figure) avec un secret dont il est le seul à connaitre. Il envoie ensuite ses données chiffrées au cloud. Ce dernier effectue les traitements sur les données chiffrées, génère les résultats eux aussi chiffrés (en violet sur la figure), et les envoie à Bob. Pour retrouver les résultats en clair des traitements effectués par le cloud, Bob déchiffre simplement ce que lui a envoyé le service cloud.

Remarque : Même si les clés se trouvent côté client, et ne tombent pas dans les mains de notre adversaire type Oscar, il n'est pas exclu de les protéger localement.

↗ **Comme c'est un chiffrement côté client, la confidentialité des données vis-à-vis du service cloud et d'Oscar est donc assurée.**

↘ **L'intégrité des données n'est, quant à elle, pas assurée car les données chiffrées sont malléables[16].**

En regardant de plus près la littérature, on se rend compte que la majorité des chiffrements homomorphiques sont partiels, c'est-à-dire qu'ils permettent d'exécuter certains traitements sur les données chiffrées, mais pas d'autres. Cependant en 2009, Craig Gentry, chercheur à IBM, a été le premier à proposer un chiffrement homomorphique total (dit FHE, *Fully Homomorphic Encryption* en anglais), donc étant capable d'exécuter n'importe quel traitement. A partir de là, le FHE est devenu en quelque sorte le « saint graal »[17] de la cryptographie pour les traitements externalisés, tels que ceux potentiellement exécutés au sein du cloud.

> *« It's like one of those boxes with the gloves that are used to handle toxic chemicals. All the manipulation happens inside the box, and the chemicals are never exposed to the outside world. »*
>
> (Craig Gentry, parlant de son chiffrement FHE)

[16] Un chiffrement est dit « malléable » lorsqu'un attaquant peut transformer une donnée chiffrée c_1 (correspondante à la donnée en clair m_1) en une autre donnée chiffrée c_2 qui équivaut à une donnée en clair m_2 qui a du sens, et ceci sans en apprendre davantage sur la donnée en clair originale m_1.

[17] http://dl.acm.org/citation.cfm?id=1666445

Malheureusement, le chiffrement de Gentry n'est pas efficace en termes de performances de traitements : Gentry a estimé qu'effectuer une recherche Google se basant sur son chiffrement prendrait mille milliard (10^{12}) plus de temps que la version standard de Google.

Depuis lors, le chiffrement de Gentry a été peu à peu amélioré, mais la route est longue. La meilleure implémentation des différents chiffrements FHE est celle que l'on peut trouver dans l'article « *Homomorphic Evaluation of the AES Circuit (Updated implementation, 5 January 2015)* »[18] basé sur la librairie HElib[19] : 4 minutes sur un laptop standard pour faire une opération cryptographique de l'AES en FHE. Notez que la librairie HElib est maintenue par des cryptographes de renom sous licence GPL[20], et il ne semble pas que l'industrie souhaite encore l'implémenter.

4.3. Protocole SPDZ

De la même façon que l'on peut faire du chiffrement threshold, il est aussi possible d'exécuter un calcul partagé (dit MPC, *Multi-Party Computing* en anglais). Le MPC est une sous-division de la cryptographie qui permet à plusieurs utilisateurs d'effectuer un traitement commun où chacun effectue sa part du calcul basé sur ses propres données secrètes en input sans avoir besoin de révéler ces dernières aux autres utilisateurs.

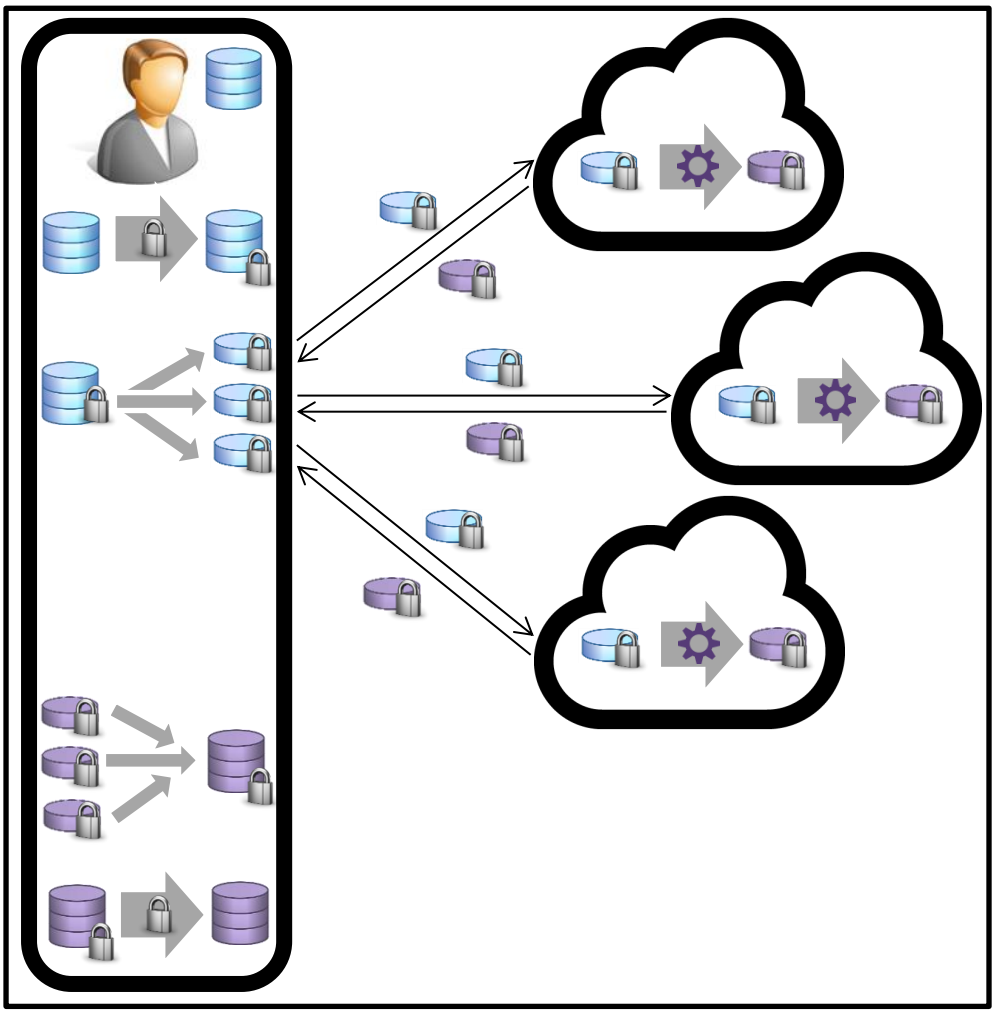
Dans la famille du MPC, le protocole SPDZ (prononcé « speedz ») est une découverte majeure pour le calcul partagé dans le contexte du cloud computing grâce à son utilisation du chiffrement homomorphique. Il est le résultat d'une recherche commune entre les universités de Bristol (Royaume-Uni) et d'Aarhus (Danemark)[21]. L'avantage du protocole SPDZ par rapport au chiffrement homomorphique est que les calculs sont distribués, et donc exécutés beaucoup plus rapidement.

[18] http://eprint.iacr.org/2012/099.pdf

[19] https://github.com/shaih/HElib

[20] Aussi connue sous le nom *GNU General Public License*. Pour plus de détails, le lecteur intéressé peut consulter http://www.gnu.org/licenses/gpl-3.0.fr.html.

[21] D'abord http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.359.5753&rep=rep1&type=pdf (CRYPTO 2012) puis https://eprint.iacr.org/2012/642.pdf (ESORICS 2013)



Dans le contexte du cloud, Bob souhaite faire exécuter son lourd traitement à plusieurs services cloud, sans que cela ne compromette ses données. Comme illustré grossièrement dans la figure ci-dessus, Bob chiffre d'abord homomorphiquement sa base de données (en bleu sur la figure) avec un secret qu'il est le seul à connaitre. Il divise ensuite ses données suivant le nombre de services cloud utilisés (dans la figure, il y en a 3), et il envoie chaque sous-division de données chiffrées à un service cloud. Ces derniers effectuent les traitements sur les sous-divisions de données chiffrées, génèrent les résultats des sous-divisions eux aussi chiffrés (en violet sur la figure), et les envoient à Bob. Pour retrouver le résultat final en clair, Bob regroupe les résultats des sous-divisions envoyés par les services cloud et déchiffre simplement le tout.

Remarque : Même si les clés se trouvent côté client, et ne tombent pas dans les mains de notre adversaire type Oscar, il n'est pas exclu de les protéger localement.

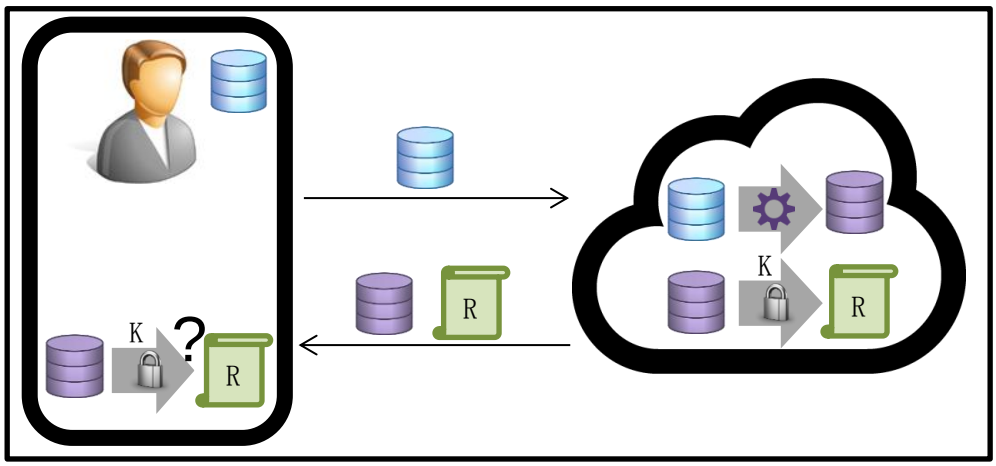
↗ **Comme c'est un chiffrement côté client, la confidentialité des données vis-à-vis du service cloud et d'Oscar est donc assurée.**

↘ **L'intégrité des données n'est, quant à elle, pas assurée car les données chiffrées sont malléables (à cause du chiffrement homomorphique).**

Le protocole SPDZ a reçu un accueil chaleureux auprès de nombreuses entreprises IT, en particulier des secteurs de la sécurité et de la finance. Il est maintenant dans le processus de commercialisation via Dyadic Security Limited[22], société cofondée par Smart and Lindell (deux professeurs de sciences informatiques, Smart étant un des créateurs du protocole SPDZ).

4.4. Calcul vérifiable

De la même façon que l'on peut faire une preuve de stockage, il est aussi possible d'effectuer un calcul vérifiable (dit VC, *Verifiable Computing* en anglais). Cette méthodologie permet à un utilisateur d'externaliser des traitements au sein d'un service cloud, et de s'assurer plus tard que le service cloud a bel et bien effectué les bons traitements. C'est donc une façon de s'assurer de l'**intégrité** de ces traitements. Ce concept a été introduit par Gennaro, Gentry, Parno dans un article[23] publié à la conférence CRYPTO 2010.



[22] https://www.dyadicsec.com/

[23] http://research.microsoft.com/pubs/138316/submitted-CRYPTO-2010-camera-ready.pdf

La figure ci-dessus est une illustration de ce type de vérification. Pour s'assurer que le service cloud a bien exécuté le bon traitement demandé par Bob (en violet sur la figure), le service cloud (le prouveur) calcule une preuve $\mathbb{R}$ du bon traitement[24] qui dépend d'une clé secrète $\mathbb{K}$ partagée entre Bob et le service cloud[25]. A la réception de la preuve $\mathbb{R}$, Bob vérifie si elle est correcte.

Point important à retenir ici : vérifier la preuve doit être moins long pour Bob que d'exécuter lui-même le traitement, sinon il n'y a aucun intérêt à utiliser ce genre de méthodologie.

↘ **Comme il n'y a pas de chiffrement des données, aucune confidentialité n'est assurée par un VC.**

↗ **Si un VC est combiné avec un FHE, alors la confidentialité des données vis-à-vis du service cloud et d'Oscar est assurée.**
↗ **Comme le calcul de la preuve se fait côté serveur, l'intégrité des données vis-à-vis d'Oscar n'est assurée que si la clé secrète $\mathbb{K}$ est protégée.**

Même si le calcul vérifiable est une méthodologie reconnue en théorie, il semblerait qu'elle soit malheureusement trop couteuse en pratique.

4.5. Cloud security gateway

Cette technique a déjà été présentée à la section 3.5 de cette note. Elle peut également être utilisée pour protéger les traitements de données effectués dans le cloud.

Comme chaque CSG est configurée spécialement pour un ou plusieurs services cloud, les fonctionnalités de chaque service cloud sont conservées. Par exemple dans le cadre d'un service d'email, Bob peut faire ses requêtes habituelles à la CSG comme s'il interagissait avec le vrai service d'email (p. ex. trier les mails par auteur, faire des recherches sur base de mots-clés, etc.) ; la CSG va ensuite convertir ces requêtes pour que le service cloud les interprète et y réponde correctement ; la CSG va ensuite reconvertir les réponses du service en données dites en clair, donc lisibles et compréhensibles pour Bob.

[24] La preuve faite par le service cloud peut être soit une *Probabilistically Checkable Proof* (PCP), soit un *Succinct Non-interactive Argument of Knowledge* (SNARK). La description de ces deux méthodes est hors du cadre de cette note.

[25] Notez que $\mathbb{K}$ peut aussi représenter une paire de clés publique/privée.

↗ **Comme les opérations cryptographiques se font côté client, la confidentialité et l'intégrité des données vis-à-vis du service cloud et d'Oscar sont assurées.**

Dans le marché des CSG pour protéger les traitements dans le cloud, on retrouve les mêmes fournisseurs qu'en section 3.5, à savoir CipherCloud, Netskope, Perspecsys, Protegrity, Skyhigh Networks ou encore Vaultive. Ici aussi chaque fournisseur de CSG propose plusieurs produits, chacun étant spécialement conçu pour un service cloud bien précis : par exemple CipherCloud propose des produits pour Gmail, Salesforce, Amazon Web Services, et Netskope propose des produits pour Yammer, Evernote ou encore Office 365.

4.6. Conclusions du traitement protégé

Si l'on ne recherche que la confidentialité, alors le chiffrement homomorphique et ses dérivés tels que le protocole SPDZ sont les techniques les plus sûres pour protéger les traitements dans le cloud. Par contre, ceci n'est vrai que pour un certain nombre d'opérations bien réduites, le FHE étant toujours au stade de recherche scientifique, donc pas encore implémenté en pratique par les industriels.

L'utilisation d'une CSG est également une très bonne protection. Elle garde néanmoins les mêmes avantages (charge de calculs et clés cryptographiques chez la CSG) et inconvénients (non-confidentialité au sein de l'entreprise, SPOF, coûts) que ceux énumérés en section 3.6.

Le choix entre un chiffrement homomorphique ou une CSG doit donc être bien clairement étudié suivant le cas précis de souhait d'utilisation et le niveau de sécurité voulu.

5. Conclusions

Il existe une multitude de mécanismes cryptographiques qui permettent de protéger le stockage et le traitement des données dans les environnements cloud. Suivant le niveau de sécurité voulu, alors différentes techniques peuvent être déployées ; chiffrement classique, threshold, preuve de stockage ou cloud security gateway pour le stockage ; chiffrement homomorphique, protocole SPDZ (et plus généralement le calcul partagé), calcul vérifiable ou encore cloud security gateway pour le traitement. Chaque fin de section apporte une conclusion sur les solutions présentées.

Aucune solution n'est encore parfaite, et la recherche académique doit encore perfectionner toutes ces techniques pour qu'elles soient davantage infaillibles. En particulier, le développement et l'implémentation pratique du chiffrement homomorphique est attendu au tournant, car cette technique peut amener une sécurité et une facilité de calcul difficilement égalable.

Finalement, même si aucune solution n'est encore parfaite, la plupart sont envisageables pour protéger les données dans le cloud, à condition que les clés cryptographiques utilisées soient sous le contrôle de l'utilisateur/organisation.

La section Recherche de Smals produit régulièrement des publications couvrant de nombreux domaines du marché IT actuel. Vous pouvez obtenir ces publications via le site web de la section Recherche :

http://www.smalsresearch.be