Appendix C

Résumé détaillé

C.1 Contexte

L'informatique décisionnelle (BI) n'a cessé de croitre depuis plus de vingt ans, mais l'avènement récent de l'Infonuagique permet désormais de déployer des analyses de données encore plus facilement. Alors que la construction d'un système de BI traditionnelle nécessite généralement un investissement initial important avec l'Infonuagique et le modèle de paiement à la demande, les utilisateurs peuvent ponctuellement consacrer de petites quantités de ressources en échange d'un avantage en temps. Cette tendance est actuellement proposée par de nombreuses offres "de service BI" avec des enjeux économiques élevés.

Bien que l'Infonuagique soit actuellement en plein essor, la sécurité des données reste une des principales préoccupations des utilisateurs d'Infonuagique et des futurs utilisateurs. Certains aspects de la sécurité sont héritées des architectures distribués classiques, par exemple, l'authentification, les attaques de réseau et l'exploitation de certaines vulnérabilités, mais D'autres sont directement liés au nouvel environnement du Infonuagique, par exemple, la fiabilité d'un fournisseur de services de Infonuagique ou d'un sous-traitant, l'efficacité de la disponibilité et les mashups incontrôlées [2–4]. Dans le contexte particulier du cloud BI, la protection des données privées a une grande importance. Jusqu'à présent, les questions de sécurité ont été traitées par les fournisseurs de service (CSPs). Mais avec la multiplication des CSPs et des sous-traitants dans de nombreux pays, les questions juridiques complexes se posent, ainsi qu'une autre question fondamentale: la confiance. A savoir si la confiance doit être accordés aux CSPs ou finalement déplacer la prise en charge de la sécurité vers les utilisateurs finaux, avec les coûts générés.

Les risques de la sécurité des données stockées dans les nuages (surtout de type publics) sont représentés dans la figure C.1. Les données de l'utilisateur pourraient être supprimées, endommagées ou perdues pour plusieurs raisons. Premièrement, certains CSPs ont la politique de prendre le plus de profit. Par conséquent, les données non modifiées ou non utilisées peuvent être supprimées afin de servir d'autres clients. Deuxièmement, la perte de données peut aussi être causée par exemple, un incident involontaire, électrique ou réseau, ou intentionnel, par exemple, l'entretien ou la sauvegarde du système. En plus, les architectures du cloud basée sur la virtualisation possèdent des failles et ne sont pas suffisamment protégés contre les attaques. Enfin, tous les CSPs ne peuvent pas garantir à 100% la disponibilité des données, bien que certaines entreprises du cloud doivent fonctionner sur une base 7/24. Ainsi, la confidentialité des données, la disponibilité et l'intégrité sont les principaux enjeux en matière de sécurité des données dans les nuages.

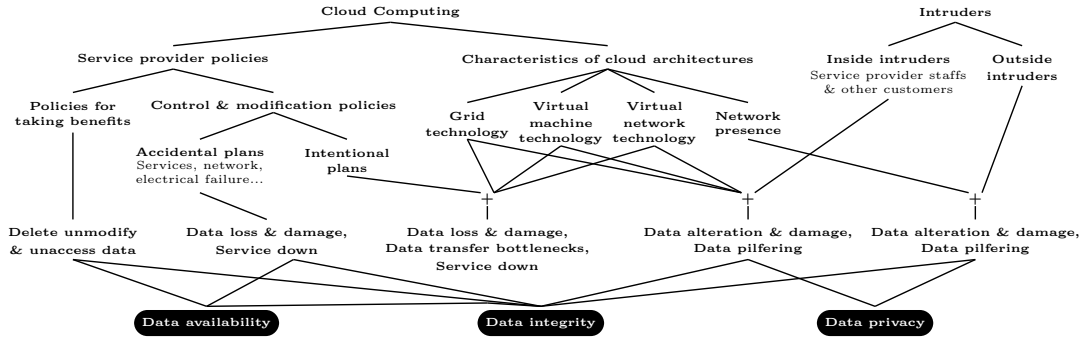

FIGURE C.1: Les risques (ou failles) de la sécurité des données entreposées dans les nuages

Dans le contexte du cloud BI, les entrepôts de données (DWs) dans les nuages ne doivent pas seulement être fortement protégés, mais aussi efficacement actualisés et analysés par le traitement d'analyse en ligne (OLAP). De là, pendant que les CSPs doivent optimiser la qualité de service et le profit, les utilisateurs cherchent à réduire les coûts de stockage et d'accès dans le modèle de paiement à la demande. Ainsi, pour les entrepôts de données dans les nuages, le compromis entre la sécurité des données et l'analyse OLAP à grande échelle pose un grand défi [4, 5].

C.2 Motivation et contribution

Les recherches existantes pour résoudre la confidentialité, la disponibilité et l'intégrité des données proposent des solutions basées sur la cryptage, l'anonymisation, la réplication ou la vérification des données. Seule l'utilisation du partage de clés secrètes (mono ou multi) permet de résoudre simultanément tous les problèmes de sécurité (confidentialité

des données, de disponibilité et d'intégrité). Toutefois, le stockage/mise à jour/Accès aux données avec le partage de clé secrète peut être difficile à mettre en œuvre et plus coûteux, car ces approches répliquent les données n fois et ne permettent pas l'accès aux données cryptées.

Toutes les approches basées sur le partage de clés secrètes permettent de crypter les données qui ne peuvent pas être déchiffrées par un seul CSP, ni aucun intrus qui arriverait à pirater un CSP. Cependant, une coalition ou de la compromission d'au moins t CSPs rompt le secret. Pour la disponibilité des données, toutes les approches permettent l'accès aux partages de $t \leq n$ CSPs, à savoir, les données partagées sont encore disponibles lorsque jusqu'à $n - t$ de CSPs sont indisponibles, à cause de défaillances techniques ou même par la malveillance. Toutefois, aucune approche ne permet la reconstitution des données partagées même si un seul CSP est indisponible, entravant ainsi les capacités de mise à jour des DBs dans les nuages. Bien que les approches de partage de secrets disposent de tous les opérateurs d'interrogation de base DB, aucun ne gère OLAP. Cependant, quelques approches garantissent effectivement l'intégrité des données, grâce à la vérification de seulement le code interne (pour vérifier si les CSPs sont malveillants). Enfin, une seule approche apporte des solutions pour réduire le volume de stockage global de sorte qu'il tombe bien sous n fois que des données originales, et ainsi diminuer les coûts financiers de stockage dans le modèle de paiement à la demande. Toutefois, son coût d'accès aux données reste élevé parce que les données interrogées doivent être entièrement reconstruites à l'avance.

Pour répondre à toutes ces questions, nous proposons deux nouvelles approches qui reposent sur un partage de clé secrète de base-p (bpVSS) et sur un mécanisme flexible vérifiables de partage de clés secrète (fVSS). A notre connaissance, bpVSS et fVSS sont les premières approches à base de partage de clés secrètes qui permettent l'exécution d'opérateurs OLAP sur DW ou cubes partagés sans reconstruire toutes les données en premier, et tout en minimisant le volume global des données partagées à moins de n fois que des données originales. Elles disposent également tous les deux (pour détecter des données incorrectes avant décryptage) de signatures pour la vérification des données interne (pour vérifier si les CSPs sont malveillants) et externe. En plus, fVSS est la première approche qui garantit qu'aucun groupe de CSPs ne peut contenir suffisamment da données partagées pour reconstruire le secret. fVSS permet également le rafraîchissement du DW lorsque l'un ou plusieurs CSPs est indisponible, et permet aux utilisateurs de régler le volume de données partagées de chaque CSPs ainsi sont optimisés les coûts par rapport aux différentes politiques de tarification des CSPs.

C.3 bpVSS: Base-p partage vérifiable de clé secrète

bpVSS est un nouveau schema vérifiable de clé secrète de type (t, n) base-p programme. Comme toutes les approches fondées sur le partage de secret, bpVSS partage les données sur n CSPs (Figure C.2), t est le nombre nécessaire de données partagées pour reconstruire les données originales. Chaque CSP ne stocke qu'une partie des données partagées, qui ne sont pas exploitables, ni par le CSP, ni par tout intrus, parce qu'elles ont été transformées par une fonction mathématique. Bien qu'on améliore le coût de traitement à travers cette approche, les données doivent être décryptées. Les résultats sont mathématiquement transformés sans que l'utilisateur soit au courant et ainsi elles seront reconstruites en informations significatives. Les données individuelles cryptées et les résultats de traitement étant chiffrées, leur transfert à travers les réseaux des CSP est donc sûr. Ainsi, la confidentialité de l'utilisateur est préservée à chaque point d'accès externe (réseaux, fournisseurs). La disponibilité est également garantie parce que les données peuvent encore être reconstituées si $n - t$ CSPs disparaissent.

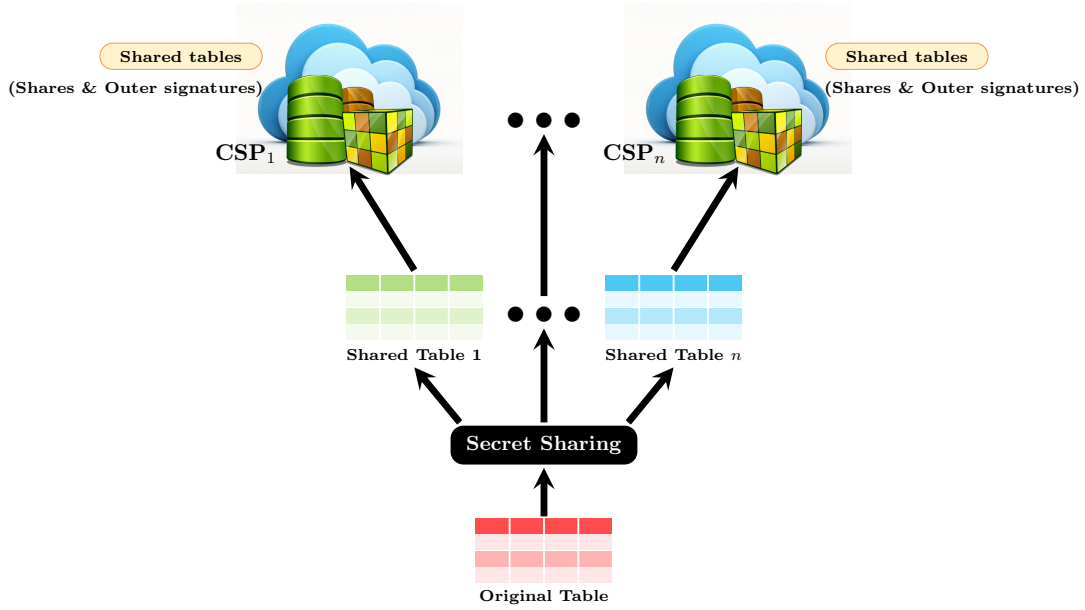

FIGURE C.2: Approche bpVSS

Dans bpVSS, un élément de données (un entier décimal) se transforme en un nombre entier de base-p tel que p est inférieur à la valeur de données. Ensuite, tous les chiffres d'un nombre entier de base-p sont cryptées à la fois par n distinctes t variables des équations linéaires f_i. Le volume des données partagées est beaucoup moins important que le volume de ses données, car les coefficients de f_i et base-p chiffres sont contrôlés à plus faible que défini par l'utilisateur paramètre p. Par conséquent, le volume totale des données partagées dans tous les n CSPs est inférieure n fois au volume de données.

Le volume de données partagées et donc le coût de stockage sont réduits au minimum par rapport aux autres approches.

Contrairement à toutes les approches, deux types de signatures (signatures internes et externes) sont incorporées dans bpVSS pour vérifier l'honnêteté des CSPs, l'exactitude des données et des données partagées. Les signatures internes créées à partir d'une fonction homomorphe pour aider à vérifier l'exactitude des données au cas où certains CSPs ne sont défaillants. Cependant, ils sont cachés dans des données partagées (ils font partie des données partagées), et donc pas de stockage supplémentaire pour garder les signatures internes. Les signatures extérieures créées à partir d'une fonction à sens unique permettent de vérifier les données partagées incorrectes ou erronées avant de reconstituer les données. Par conséquent, aucune données partagèes erronée n'est transférée à l'utilisateur pour la reconstruction. Cela permet de minimiser à la fois le coût de transfert de données et le coût de l'informatique du côté de l'utilisateur. Contrairement aux signatures intérieures, les signatures extérieures sont stockées dans des attributs supplémentaires dans des tables partagées.

Dans bpVSS, chaque table d'un DW partagée est stockée dans une base de données relationnelle chez un CSP, chaque valeur d'attribut est chiffrée indépendamment. Ainsi, bpVSS aide à mettre en œuvre un modèle logique DW, à savoir, en étoile, en flocon de neige ou en un schéma de constellation. Chaque DW partagé se base sur le même schéma que l'original de DW, mais le type et la taille de chaque attribut dans les tables partagées diffèrent des tables originales. Tous les types d'attributs sont en effet transformés en nombres entiers à l'exception des booléens qui ne sont pas cryptés pour préserver le coût de calcul et de stockage des données. Cependant, des tables partagées ont un plus grand nombre d'attributs que les tables d'origine, car les attributs de signatures extérieures sont également stockés dans des tables partagées. Toutes ne sont pas cryptées. Ils aident à détecter les enregistrements dans différentes tables partagées de chaque CSP. En plus, ils aident à regrouper les enregistrements partagés et les résultats dans le processus de restauration des données.

Pour gérer les types de données usuelles figurant dans les bases de données, nous chiffrons et traitons chaque valeur d'attribut de façon indépendante. Chaque valeur d'attribut (par exemple, réelles, caractères, chaînes de caractères, les chaînes binaires) est d'abord transformée en un ou plusieurs entiers en fonction du type de données. Ensuite, les nombres entiers sont chiffrés par bpVSS. Par exemple, un caractère est transformé en un nombre entier positif à travers son code ASCII. Comme certaines approches similaires, bpVSS permet l'analyse des données sur les données partagées. Pour optimiser le coût de l'informatique et le coût de transfert des données du côté de l'utilisateur lors de l'analyse des données, les requêtes de correspondance exacte et

les fonctions d'agrégation peuvent être directement effectuées sur les données partagées. Les résultats agrégés sont alors transférés à l'utilisateur pour la reconstruction. Toutes les opérations OLAP de base (roll-up, drill-down, some slice and dice, pivot and drill-across) peuvent également être appliquées directement sur les données partagées (cubes de nuages) des CSPs, avec des résultats en cours de reconstruction chez l'usager. Les cubes sont physiquement stockés dans des tables relationnelles qui sont partagées entre les CSPs, en conservant la même structure.

C.4 fVSS: Mécanisme de flexibilité de partage de secret vérifiables

fVSS est un schéma de (t, n) partage vérifiable flexible de clé secrète. Comme une approche basée sur le partage de données, fVSS découpe en n CSPs données partagées (Figure C.3), dont t sont nécessaires pour reconstruire les données originales. Par conséquent, ce qui garantit à la fois la confidentialité des données et la disponibilité de celles-ci.

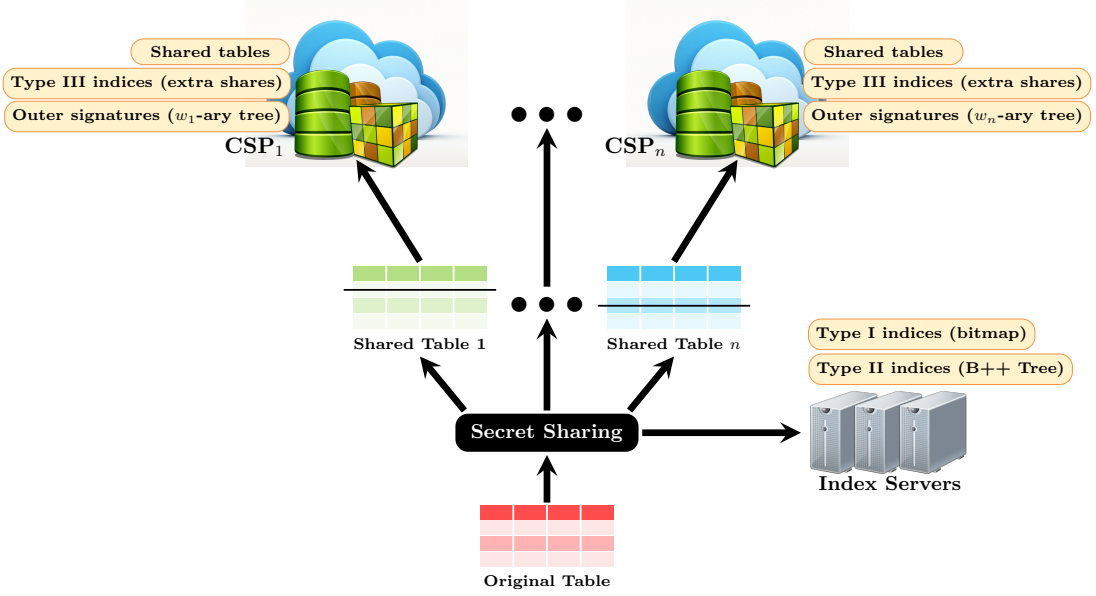

FIGURE C.3: Approche fVSS

Pour optimiser le volume de données partagées et donc le coût, nous partageons un morceau de données de moins que n fois. Seuls $n - t + 2$ parts d'une partie de données sont construits à partir d'un polynôme de degré t, qui est construit par un polynôme par interpolation de Lagrange en utilisant une signature intérieure et les valeurs semi-aléatoires $t - 2$ appelé donnée méta encryptée. Étant donné que chaque élément de la donnés est partagé à seulement quelques n CSP, fVSS est le premier partage de

secret flexible qui permet aux utilisateurs de régler le volume des données partagées en fonction des politiques de tarification CSP. Le volume déséquilibré des données partagées à chaque CSP aide en effet à minimiser le stockage et de calcul des coûts dans le modèle de paiement à la demande par la conception. En plus, contrairement à tous les autres systèmes de partage de secret, fVSS permet la mise à jour des données partagées en cas de défaillance de CSPs, tout simplement en ne sélectionnant pas les CSPs par défaut pour le partage de nouvelles données. fVSS atteint alors un niveau de sécurité plus élevé, car il peut protéger les données, même si toutes les CSPs sont malveillants ou indisponibles. Aucun groupe de CSPs peut contenir suffisamment de données partagées pour reconstruire le secret si $n < 2 \times t - 2$. En effet, $n < 2 \times t - 2 \Leftrightarrow nt + 2 < t$, soit le nombre de données partagées est inférieur au nombre de données partagées nécessaire à la reconstruction.

L'intégrité des données est renforcée avec des signatures à la fois internes et externes qui aident à détecter les erreurs dans les données partagées et les résultats de requête. Les signatures intérieures cachées dans des données partagées sont vérifiées après la reconstruction avec une fonction à sens unique, comme dans bpVSS. Les signatures extérieures sont stockées dans une structure de données arborescente. Ces dernières sont créées à l'aide d'une fonctions à sens unique, qui aident à réduire le volume de la signature extérieure et permettent une mise à jour à chaque fois qu'un DW partagé est rafraîchie. Par conséquent, la mise à jour des signatures extérieures accélère le stockage ainsi les coûts sont réduits au minimum. Plusieurs types de vérifications de signature externe nécessaires (vérification des enregistrements, vérification d'ensemble d'enregistrements, vérification de table, l'ensemble de tables, et de vérification DW) sont disponibles sur demande, parce que les signatures extérieures sont créées indépendamment des différentes combinaisons de enregistrements partagés ou des tables partagées avec plusieurs fonctions de cryptage. Par ailleurs, les signatures extérieures à différents niveaux dans l'arborescence de la signature extérieure peuvent être créées et vérifiées pour réduire le taux de données incorrectes non détectées, ainsi l'intégrité est améliorée.

Étant donné que chaque table d'un DW partagée est stockée dans une base de données relationnelle d'un CSP, chaque valeur d'attribut dans chaque document est chiffrée indépendamment, notre approche permet d'appliquer un modèle logique de DW, à savoir, étoile, flocon de neige et de schémas de constellation. Un DW partagé porte le même schéma que l'original de DW. Cependant, tous les types d'attributs sont transformés en réels par le processus de partage des données. Contrairement à bpVSS, les signatures extérieures sont stockées en dehors des tables partagées, à savoir, dans l'arborescence de la signature. Ainsi on a moins d'enregistrements partagés à crypter de n'importe quel enregistrement original et moins de tables partagés pour les stocker. Ainsi, le nombre d'enregistrements dans une table partagée est inférieur à celui de la

table d'origine, et diffère de celle des données partagées des autres CSPs. Pour améliorer les performances de la requête et réduire les coûts de stockage et de calcul, le nombre d'enregistrements partagés dans la table partagée doit être ajusté pour respecter les politiques de tarification des CSPs.

Comme d'autres approches de base de données sécurisées, fVSS permet l'analyse de données sur les données partagées. Pour analyser les données, des requêtes SQL traditionnelles peuvent être effectuées sur les données partagées des CSPs sans décryptage, puis seulement les résultats sont reconstruits pour l'utilisateur. Cela permet de réduire les coûts de communication et de calcul chez l'usager. Trois types d'indices permettent d'améliorer les performances des requêtes. Les indices de type I sont des bitmaps stockés dans le serveur d'index. Ils stockent des emplacements de données partagées et sont utilisés dans les requêtes SUM et AVG. Comme dans certaines approches existantes, les indices de type II sont des arbres B+ stockés sur le serveur de l'indice. Ils sont utilisés dans correspondance exacte, la portée et MAX, MIN, médiane, le mode et les requêtes COUNT. Les indices de type III sont cryptées et stockées chez CSP. Ils sont exploités pour les requêtes instance de stdDev et la variance. Comme bpVSS, fVSS se base sur le stockage des cubes de données qui permettent d'optimiser le temps de réponse et la bande passante lors de l'exécution des opérations ROLAP. Contrairement à bpVSS, fVSS crée un cube partagé d'enregistrements et indices sans reconstruire les données. Enfin, les cubes partagés peuvent être actualisés, même si certains CSPs disparaissent.

Comme bpVSS, fVSS se base sur le stockage des cubes de données qui permettent d'optimiser le temps de réponse et la bande passante lors de l'exécution des opérations ROLAP. En plus dans fVSS, les cubes sont créés directement dans le nuage et mis à jour par les données partagées et indices seulement. Cependant, les données partagées sont réelles et les signatures extérieures ne sont pas stockées dans des cubes du cloud. Depuis le Infonuagique, les cubes sont construits à partir des données partagées, ils sont physiquement stockés dans des tables qui doivent être partagées à tous les n CSPs, parce que les méta-données partagées ne sont pas disponibles. Cependant, les cubes de cloud peuvent être actualisés, même si certains CSP sont défaillants. En plus de références de dimensions usuelles et d'agrégats, ils peuvent inclure des attributs supplémentaires qui sont effectivement intégrés de type indices III.

C.5 Sécurité, performance et analyse des bpVSS fVSS

Dans cette section, nous discutons de la sécurité et la performance de bpVSS et fVSS. Par sécurité des données, nous entendons la confidentialité, la disponibilité et

l'intégrité des données. Le volume des données cryptées et le coût du temps de traitement sont attribuées à la performance.

La confidentialité des données est le principal problème de sécurité sur lequel nous nous concentrons. De par leur conception, nos approches renforcent la protection des données basées sur le partage des données et garantissant ainsi qu'elles ne peuvent pas être déchiffré par un seul CSP ou un intrus qui pirater un CSP. L'approche fVSS garantit qu'aucun groupe de CSPs ne peut avoir suffisamment de données partagées pour reconstruire les données d'origine si $n < 2 \times t - 2$. Toutefois, dans le cas où un intrus peut voler une partie à partir d'au moins CSP, la probabilité d'apparition du secret dépend de t et $\|p\|$ (en bpVSS). Pour atteindre une protection plus élevée, t et $\|p\|$ devraient être les grands entiers. Non seulement la sécurité mais aussi la performance dépendent de t et $\|p\|$. Lorsque t est grand, nos approches produisent de petites données partagées, ce qui permet de minimiser la consommation de mémoire et le temps d'exécution lors du chargement et de l'accès aux données. Cependant, t peut ne pas être trop grand, parce que le nombre des CSPs est limité dans la pratique. De même, $\|p\|$ ne devrait pas être supérieur à la taille maximale d'un élément de données dans bpVSS. Lorsque t et $\|p\|$ sont affectés à de grands entiers, le volume global des données partagées est très large, et donc le coût de stockage de données est élevé. Pour atteindre la plus grande sécurité avec le plus bas coût possible du stockage, t devrait être un grand entier et $\|p\|$ devrait égale à $\|d_{max}/(t-1)\|$, où $\|d_{max}\|$, est la taille de données secrètes la plus importante. En revanche, t n'a pas d'impact sur volume global de données partagées du fVSS, qui est fixé à deux fois, une première fois avec le volume de données d'origine et une deuxième fois lorsque $n = t$. Par conséquent, aucune tradoff entre la confidentialité et le coût de stockage avec fVSS. Toutefois, lorsque t est grand, l'efficacité du partage des données et de la reconstruction est négativement impacté.

En ce qui concerne la disponibilité, nos approches, toujours par conception, permettent de reconstruire les secrets, à savoir, les données partagées de la requête, lorsque $n - t$ CSPs est indisponible. En outre, fVSS permet également la mise à jour des données partagées dans le cas où plus $t - 2$ CSPs sont indisponibles, tout simplement en partageant de nouvelles données entre $n - t + 2$ CSPs disponibles. Toutefois, la part globale du volume augmente avec n lorsque t est fixé. Le temps de partage chez l'utilisateur augmente également avec n. Ainsi, pour atteindre la disponibilité des données tout en minimisant le volume global des données partagées et de temps de partage, n devrait être proche de t. Cependant, n et t doivent respecter le condition $n < 2 \times t - 2$ et $t > 2$ dans fVSS pour atteindre la protection la plus élevée (aucun groupe de CSP ne peut briser le secret) et pour garantir que de nouvelles données peuvent être partagées même si certains CSP sont défaillants.

Pour assurer l'intégrité, nos approches permettent à la fois de vérifier l'exactitude des données cryptées et l'intégrité des CSPs, avec l'aide de deux vérifications de code externe et interne. L'efficacité des signatures internes et externes est plus élevée lorsque leurs tailles sont grandes. Depuis $\|s_{in}\| = \|p\|$ et $\|p\|$ concerne t en bpVSS, $\|s_{in}\|$ doit satisfaire à la condition $\|s_{in}\| = \|p\| = \|d_{max}/(t-1)\|$ pour atteindre la meilleure efficacité et le plus bas coût de stockage possible. $\|s_{in}\|$ n'a pas d'impact sur la taille des données partagées en fVSS, mais $\|s_{in}\|$ devrait être un grand entier et supérieure à une moitié de la taille des données afin de détecter tous les morceaux de données incorrectes (Section 5.3.2.3.1).

Contrairement à la taille de la signature intérieure, la taille externe augmente avec la signature $\|s_{out}\|$ lorsque t et n sont fixés. Par conséquent, l'efficacité et le volume des signatures extérieures doivent être équilibrés. Les signatures internes et externes travaillent ensemble lors de la reconstruction de données avec bpVSS. Pour détecter tous les morceaux de données incorrectes, $\|s_{out}\|$ devrait être d'au moins 6 bits. Ce résultat est obtenu lorsque le volume global de la signature est de 0,75 Go (données d'origine est de 1 Go et le volume de l'action globale maximale est de 3,1250 Go). Dans fVSS, les signatures extérieures sont vérifiées sur demande et se séparent de la signature intérieure. L'efficacité et le volume des signatures extérieures augmentent avec $\|s_{out}\|$. Ainsi, $\|s_{out}\|$ devrait être un grand entier. A partir des signatures extérieures peuvent être vérifiés sur demande sur plusieurs niveaux (par exemple, enregistrements partagés, l'ensemble d'enregistrements partagés, tables partagées, l'ensemble de tables partagées, et le DW partagé) de l'arbre de la signature w_i-aire, toutes les données partagées incorrectes peuvent être détectées si les signatures sont vérifiées à au moins trois niveaux (Section 5.3.2.3.1). En outre, w_i impacts volume de signature. Ainsi, il devrait réduire le volume de la signature, et donc le coût de stockage.

Nos approches permettent de partager les données numériques (par exemple, les entiers et les réels) et non les non numériques (par exemple, les caractères et les chaînes de caractères) des données du DW. Pour accéder aux données partagées, ils permettent à tous les types de requêtes (correspondance exacte, la portée, l'agrégat et le regroupement des requêtes) sur les données partagées. En outre, ils permettent également le traitement directement sur les données partagées en créant des cubes de données partagées. Toutefois, $n/(t-1)$ et n fois le volume initial de cube sont nécessaires pour stocker une partie du nuage dans bpVSS et fVSS. Étant donné que le nombre d'enregistrements dans un cube de nuage est égal aux différentes combinaisons de toutes les valeurs de dimension, il peut être énorme. Par conséquent, les cubes dans le cloud doivent stocker uniquement les enregistrements qui sont souvent accessibles pour réduire le volume de stockage, qui devient un problème de sélection de vue matérialisée. D'autres enregistrements peuvent être directement extraits de la DW partagé.

Enfin, non seulement le paramétrage, mais aussi le volume déséquilibré des données partagées aide à réduire la taille du modèle ainsi que le stockage et le coût de traitement dans fVSS. Ainsi, le plus grand volume de données partagées doit être conservé auprès du CSP le moins cher. Cependant, la taille maximale de la machine virtuelle doit être attribué aux CSPs qui stockent les plus gros volumes de données partagées, à réduire l'écart entre les temps d'exécution les plus bas et les plus élevés du CSPs. Le partage de données ou l'accès fonctionne parallèlement sur les CSPs, le temps total de traitement est en effet représenté par le traitement individuel le plus important.

C.6 Étude comparative

Dans cette section, nous expérimentons nos approches proposées dans les chapitres 3 et 4. En effet, nous les comparons avec les deux états de l'algorithme de l'état de l'art (Thompson et al approche [58] et Hadavi et al approche [21] présentée dans la chapitre 2), car seuls ces approches se concentrent sur les trois aspects de la sécurité (confidentialité, disponibilité et intégrité des données) et respectent aussi la performance lorsque les requêtes agrégées sont exécutées. La performance de toutes les approches expérimentales est mesurée avec un volume de données partagées, le partage de données/temps de la reconstruction, le temps de réponse de la charge de travail et le volume de données transféré pour confirmer l'étudier théoriquement dans la chapitre 5. Dans les expériences, nous utilisons l'indice de référence pour un schéma en étoile et un paramètre p qui varie (en bpVSS) avec $t = 3$, $n = 4$ et $w = 100$ (en fVSS).

Tableau C.1 caractéristiques de volume de données, temps d'exécution, le volume de transfert de données et les coûts financiers de toutes les approches expérimentales. Les coûts financiers sont estimés à partir des politiques de tarification de CSP représentés dans le Tableau 5.1 (Chapitre 5) et le serveur d'index utilise le même prix de la CSP le plus cher pour obtenir le meilleur service. Notez que fVSS-I et fVSS-II sont les parties qui traitent le déséquilibre des données partagées et les stratégies de déséquilibre de données partagées dans fVSS.

Lorsque SSB DB dimensionnement 757 Mo est partagé avec des approches expérimentales, volume de stockage fVSS (tout type de données) est la plus faible. Bien bpVSS a réussi à minimiser le volume global de données partagées, son volume de stockage global (tous les types de données) est encore plus grande que celle de l'approche Hadavi et al, parce qu'il construit d'énorme volume de signatures extérieures.

Toutefois, lorsque le coût de stockage financier est estimé à partir du volume de stockage, le coût de stockage fVSS-II et bpVSS sont les moins importants. bpVSS coût

TABLE C.1: Comparaison des approches de partage de base de données

	Thompson	**Hadavi**	**bpVSS**	**fVSS-I**	**fVSS-II**
Storage volume (GB)	4.14	2.43	2.62	2.34	**2.27**
Data transfer volume (MB)	323.88	23.90	13.51	12.36	**12.34**
Data sharing time (min)	58.42	27.05	40.35	**23.62**	30.05
Data reconstruction time (min)	35.16	24.50	33.59	**17.95**	34.09
Data access time (min)	5.81	**2.34**	8.51	3.61	5.69
Storage cost ($/month)	$0.5584	$0.3023	$0.2973	$0.3168	**$0.2651**
Data transfer cost ($)	$0.0302	$0.0025	**$0.0012**	$0.0014	$0.0014
Data sharing cost ($)	$0.1214	$0.0935	**$0.0933**	$0.1242	$0.0996
Data reconstruction cost ($)	$0.4092	$0.2599	$0.3964	$0.2641	**$0.2071**
Data access cost ($)	$0.0887	**$0.0314**	$0.1204	$0.0520	$0.0372

de stockage est inférieure à celle de l'approche Hadavi et al bien bpVSS volume de stockage est supérieure à celle de l'approche Hadavi et al, parce bpVSS ne stocke rien au niveau du serveur d'index qui est le prix de l'unité la plus chère (le serveur d'index est pas dans la piscine CSP bpVSS). De même, étant donné que le volume de stockage de fVSS-I au niveau du serveur d'index est supérieure à celle de Hadavi et al approche, le coût de stockage de fVSS-I est plus élevée que celle de l'approche Hadavi et al bien que le volume de stockage global de fVSS-I est inférieure à celle de l'approche de Hadavi et al.

Pour le partage de données, notre SSS est inefficace pour le partage de données. Le temps de traitement de partage des données de nos approches se situe entre celle de l'approche la plus efficace (Hadavi et al) et la pire approche (Thompson et al). Cependant, le coût de calcul financier pour le partage de données est la plus faible, parce que bpVSS n'a pas de charge de travail au niveau du serveur d'index. De même, le coût de calcul financier de fVSS-I est le plus élevé, bien que les données de fVSS-I en temps partagé est inférieure à celle de l'approche de Thompson et al, parce que les données fVSS-I partage le temps au niveau du serveur d'index est plus élevée que celle de approche de Thompson et al. La stratégie de déséquilibre en fonction de la taille de la machine aide fVSS-II à réduire le coût de calcul financier par rapport à fVSS-I.

Pour la reconstruction des données, fVSS-I reconstitue des données plus rapidement grâce à la stratégie de déséquilibre du coût de calcul financier pour reconstruire des données avec fVSS-II sont les plus bas, bien que le temps de la reconstruction fVSS-II est pas plus bas. bpVSS est inefficace pour reconstruire des données, parce que bpVSS vérifie l'exactitude des données partagées en cours reconstruire. Avec le temps d'exécution supplémentaire pour vérifier les données partagées, le temps de la reconstruction bpVSS et le coût de calcul financier sont un peu inférieure à celle de l'approche de Thompson et al, qui est la pire des solutions pour reconstruire des données.

Pour accéder à des données qui est le l'usage principal du DW, notre SSS est le plus efficace lorsque le volume et le coût financier de transfert de données de départ sont

respectées. Le volume de transfert des données et le coût financier de notre proposition d'attribution de signatures sont seulement environ la moitié de celle de l'approche Hadavi et al soit environ 4% de celle de l'approche de Thompson et al. Cependant, le volume de transfert des données est toujours un problème pour toutes les approches basées sur le partage de secrets. Le goulot d'étranglement du réseau peut être à cause l'utilisateur, car un volume énorme de données est transféré in/out de l'utilisateur. Nous allons illustrer ce problème par l'exemple. Lorsque le pire Q1.1 vol requête sont exécutés avec fVSS-II étant l'approche la plus efficace pour le volume de transfert de données, les données sont transférés environ 7,13 MB de l'utilisateur et transférés sur environ 28.51 les MB de l'utilisateur. Ainsi, le volume global de transfert de données dans et hors de l'utilisateur est 35.64 Mo soit environ 4,71% du volume SSB DW original (757 Mo). Si DW dimensionnement 10 TB est partagée et une seule requête est exécutée, les données sont transférées in/out à l'utilisateur d'environ 480 Go. Ce problème se produit parce que nos systèmes de règlement peuvent fonctionner seules les requêtes de reconstruction exactes, agrégées et triées sur les données partagées, mais ils ne peuvent pas exécuter des requêtes complexes sur les données partagées. Cependant, pour résoudre ce problème, nous envisageons l'exécution de requêtes complexes sur les données partagées en matière de recherche future.

Bien que notre attribution de signatures est la plus efficace pour réduire le volume de transfert de données de départ et le coût financier, ils ne sont pas efficaces pour les temps d'exécution et le coût de calcul financier lorsque l'accès aux données avec toutes les requêtes SSB. Le temps de réponse de la charge de travail bpVSS et le coût de calcul sont les plus élevés, car il doit vérifier les données partagées avant de reconstituer les données. Le temps de réponse de la charge de travail et le calcul des coûts fVSS se situe entre celle de l'approche la plus efficace de Hadavi et al et l'approche la moins efficace celle de Thompson et al. Le temps de réponse de la charge de travail fVSS n'est pas plus élevé que celui de l'approche de Hadavi et al, parce fVSS effectue le traitement sur des réels mais l'approche Hadavi et al effectue le traitement sur des entiers.

Notez que le partage de données et de temps d'accès de notre SSS n'est pas plus bas parce que les expériences sont menées avec n et t sont de petits entiers ($n = 5$ et $t = 4$). Si n et t sont assez grands, le volume de données partagées de chaque CSP sera réduit jusqu'à ce que le temps d'exécution pour le partage et l'accès aux modifications de données au plus bas, parce que le volume de données partagées et le temps d'exécution de chaque CSP ne diminuent pas lorsque n et t diminuer.

Enfin, fVSS-II prend en considération le déséquilibre du volume de données partagées et applique une stratégie de déséquilibre sur le modèle qui permet de réduire considérablement les coûts financiers à l' exception des coûts de transfert de données qui

sont calculés à partir de fVSS-I. Cependant, dans l'expérience, les données partagées fVSS-II et les données d'accès plus lent que fVSS-I parce que l'écart entre les temps d'exécution les plus bas et plus élevés au CSP est élevé. Cela arrive parce que le volume des données partagées ne correspond pas à la puissance de la machine qui les exécute. Ainsi, nous visons dans les travaux futurs la conception d'un outil semi-automatiquement qui aide les utilisateurs à ajuster le volume des données partagées de chaque CSP, par rapport aux coûts, mais également par rapport à la qualité de service. Bien que fVSS (deux stratégies) et la comparaison des approches garantie de disponibilité des données lorsque certains CSP sont défaillants, ils ne peuvent pas accéder aux données avec des requêtes SSB si le serveur d'index échoue. En revanche, c'est possible avec bpVSS p. En outre, seules nos approches empêchent le transfert de données partagées erronées lorsque les données sont accessibles par la vérification des signatures extérieures.

C.7 Perspective

Dans cette section, nous discutons de certaines questions en suspens qui devraient être abordées dans nos approches pour améliorer la sécurité, les performances et le coût.

Tout d'abord, nous avons l'intention d'améliorer la confidentialité des données, puisque dans bpVSS, les noms de table, les noms d'attributs, des clés primaires et étrangères sont diffusés en clair. Ainsi, les données chiffrées peuvent être attaqués avec, par exemple, banburismus, références ou méthodes de fréquence (avec l'aide d'une base de connaissances). Ainsi, pour atteindre une plus grande sécurité, les noms de table, les noms d'attributs (y compris dans fVSS) et les touches doivent être cryptées pour cacher le schéma de données. Aucune méthode SSS actuelle ne peut atteindre cet objectif parce que chaque nom de la table (après que son nom soit crypté) doit être différent des autres noms de tables si elles sont stockées dans le même CSP ou noeud. De même, les noms d'attributs chiffrés et les clés primaires chiffrés dans chaque table partagée doivent être différents. En outre, les touches chiffrées dans un couple des clefs étrangères primaire devraient être différentes pour cacher la relation entre les tables cryptées. Par conséquent, nous prévoyons d'utiliser une fonction injective et un moyen de crypter les noms de table, les noms des attributs et des clés dans chaque table à chaque noeud.

Deuxièmement, nous cherchons à minimiser le risque de goulot d'étranglement et d'optimiser le temps de réponse aux requêtes. Bien que nos approches permettent des requêtes de correspondance exacte, portée et d'agrégation en cours de traitement sur des données partagées, ils ne peuvent pas exécuter des requêtes composites, à savoir, les requêtes impliquant les deux opérateurs de correspondances et d'agrégation exactes. Par conséquent, lorsque les requêtes complexes sont exécutées, de grands volumes de données

sont transférés entre l'utilisateur et les CSPs (chapitre 6). Dans bpVSS, ce problème peut se produire parce que les deux résultats de type vrai positif et faux positif de requêtes de correspondance exacte sont transférés au filtre l'utilisateur. Ensuite, seuls vrais positifs sont transférés aux CSP pour traiter l'agrégation. Ainsi, nous prévoyons d'éliminer les faux positifs de la correspondance et d'utiliser que des requêtes précises afin d'améliorer les recherches des requêtes complexes exécutées sur les enregistrements. Dans fVSS, toutes les requêtes portées sur les correspondances exactes et certaines requêtes agrégées doivent fonctionner sur indices (bitmaps et B+ arbres) stockées sur le serveur d'index. En outre, le stockage des données partagées est aléatoire. Ainsi, nous devrions chercher une solution pour organiser le stockage des enregistrements sans l'aide d'indices.

Troisièmement, nous avons l'intention d'améliorer encore le coût de notre solution dans le nuage avec le modèle de paiement à la demande. Les expériences fVSS (Chapitre 6) montrent que les coûts élevés (coûts de stockage, de calcul et de transfert des données) sont payés pour le serveur d'index. Cependant, le temps d'exécution est élevé si le volume de données est déséquilibré avec la puissance de la machine, parce que le partage de données ou l'accès fonctionne parallèlement sur les CSPs et le temps total d'exécution est le temps d'exécution individuelle le plus important. Par conséquent, nous devrions éliminer le serveur d'index et de concevoir un outil qui aide les utilisateurs en semi-automatiquement à ajuster le volume des données partagées de chaque CSP, par rapport aux coûts, mais aussi la qualité de service. Cela est possible si l'emplacement des données partagées est organisé comme dans le paragraphe précédent.

Enfin, puisque les politiques de tarification de CSP et d'entretien sont susceptibles d'évoluer rapidement, nous visons à concevoir une méthode pour ajouter et supprimer des CSPs de/vers l'ensemble CSP dans fVSS, avec les plus bas coûts de mise à jour possibles, tout en préservant l'intégrité des données. Dans tous les systèmes de règlement, y compris nos approches, tout CSP peut être immédiatement retiré sans aucun impact, même si $n \geq t$. En revanche, quand un nouveau CSP est ajouté, les données partagées de toutes les données déjà existantes doivent être partagées chez le nouveau CSP pour gérer la cohérence des données. Les données partagées enregistrées chez le nouveau CSP doivent être construits à partir de t CSP existants et donc, le processus de construction est coûteux, à la fois dans le temps d'exécution et de bande passante. Pour éviter cela, nous pouvons encore exploiter les méta-données partagées dans fVSS, mais les fonctions de construire des méta- données partagées dans la nouvelle et les CSPs existants doivent être redéfinies de manière dynamique afin de relier tous les CSPs, qui est un défi.

Nos approches peuvent être appliquées pour stocker et analyser les données massives. Les aspects de volumétrie des données, de la rapidité et l'hétérogénéité (variété), peuvent être abordées et traitées par nos approches.

Tout d'abord, notre approche est une approche de base de données distribuée où le volume de données partagées individuelles diminue lorsque le nombre de CSPs/participants augmente. Cependant, le nombre n de CSP est limité dans la pratique, et le volume des données partagées de chaque CSP peut ne pas être assez petit pour être stocké dans une base de données. Ainsi, pour stocker un grand volume de données, nous changeons la stratégie de conception de "un participant par un CSP" à "beaucoup de participants par un CSP".

Deuxièmement, dans nos approches, l'accès aux données fonctionne parallèlement avec plusieurs participants, et seuls les éléments de données nécessaires sont reconstruits par l'utilisateur. Cependant, lorsque le nombre de Participants augmente, le temps d'accès à chaque participant diminue, parce que le volume de données partagées diminue et le nombre d'enregistrements partagés peut également diminuer (en fVSS). Ainsi, nos approches, en particulier fVSS, peuvent interroger des flux de données. Toutefois, les flux de données partagés peuvent être un problème, parce que tous les éléments de données doivent être cryptés pour l'utilisateur des polynômes et des augmentations de temps de chiffrement avec le nombre de participant. En plus, le nombre de participant doit être suffisamment grand pour stocker un grand volume de données. Ainsi, la complexité de cryptage doit tomber sous polynomiale pour permettre aux données de flux de partage. Nous prévoyons que ce sera résolu par quelques recherches connexes (par exemple, des champs numériques de données, les champs de cryptographie) à l'avenir.

Troisièmement, nos approches peuvent partager tous les types de données si les éléments de données sont identifiés par une clé et leur type peut être converti en entiers, réels, des caractères, des chaînes ou des chaînes binaires. Par exemple, les fichiers de texte ne peuvent être partagés avec nos approches que si la clé générée est identique à chacun des fichiers et du cryptage des chaînes dans des fichiers texte. Par conséquent, nous pouvons partager ces types de données: textuelles ou de documents XML, des images, des vidéos ou des données graphiques.

Bibliography

[1] Abhishek Parakh and Subhash Kak. Online data storage using implicit security. *Information Sciences*, 179(19):3323–3331, September 2009.

[2] James Bret Michael, Phillip A. Laplante, Jeffery E. Payne, Paul E. Black, and Jeffrey M. Voas. Does security trump reliability? *IEEE Computer*, 46(11):84–86, November 2013.

[3] David S.L. Wei, San Murugesan, Sy-Yen Kuo, Kshirasagar Naik, and Danny Krizanc. Enhancing data integrity and privacy in the cloud: An agenda. *IEEE Computer*, 46(11):87–90, November 2013.

[4] Richard Chow, Philippe Golle, Markus Jakobsson, Elaine Shi, Jessica Staddon, Ryusuke Masuoka, and Jesus Molina. Controlling data in the cloud: Outsourcing computation without outsourcing control. In *1st ACM Cloud Computing Security Workshop (CCSW 2009), Chicago, USA*, pages 85–90, 2009. ISBN 978-1-60558-784-4.

[5] Radu Sion. Towards secure data outsourcing. pages 137–161, 2008.

[6] Dorothy Elizabeth Robling Denning. *Cryptography and data security.* Addison-Wesley, 1982.

[7] Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *2000 IEEE Symposium on Security and Privacy, berkeley, USA*, pages 44–55, 2000.

[8] Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou. Secure ranked keyword search over encrypted cloud data. In *30th IEEE International Conference on Distributed Computing Systems (ICDCS 2010), Genoa, Italy*, pages 253–262, 2010.

[9] Cong Wang, Ning Cao, Kui Ren, and Wenjing Lou. Enabling secure and efficient ranked keyword search over outsourced cloud data. *IEEE Transactions on Parallel and Distributed Systems*, 23(8):1467–1479, December 2011.

[10] Chun-I Fan and Shi-Yuan Huang. Controllable privacy preserving search based on symmetric predicate encryption in cloud storage. *Future Generation Computer Systems*, 29(7):1716–1724, September 2013.

[11] Hakan Hacıgümüş, Bala Iyer, and Sharad Mehrotra. Efficient execution of aggregation queries over encrypted relational databases. In *9th International Conference DASFAA 2004, Jeju Island, Korea*, pages 125–136, 2004.

[12] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. CryptDB: Protecting confidentiality with encrypted query processing. In *23th ACM Symposium on Operating Systems Principles (SOSP 2011), Cascais, Portugal*, pages 85–100, 2011.

[13] Stephen Tu, M. Frans Kaashoek, Samuel Madden, and Nickolai Zeldovich. Processing analytical queries over encrypted data. *PVLDB*, 6(5):289–300, March 2013.

[14] Carlos Aguilar Melchor, Guilhem Castagnos, and Philippe Gaborit. Lattice-based homomorphic encryption of vector spaces. In *IEEE International Symposium on Information Theory (ISIT 2008), Toronto, Canada*, pages 1858–1862, 2008.

[15] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *41st annual ACM symposium on Theory of computing (STOC 2009), Bethesda, USA*, pages 169–178, 2009.

[16] Nadia Bennani, Ernesto Damiani, and Stelvio Cimato. Toward cloud-based key management for outsourced databases. In *34th Annual Computer Software and Applications Conference Workshops (COMPSACW 2010), Seoul, Korea*, pages 232–236, 2010.

[17] Haibo Hu, Jianliang Xu, Chushi Ren, and Byron Choi. Processing private queries over untrusted data cloud through privacy homomorphism. In *27th IEEE International Conference on Data Engineering (ICDE 2011), Hannover, Germany*, pages 601–612, 2011.

[18] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography: The case of hashing and signing. In *14th Annual International Cryptology Conference (CRYPTO 1994),Santa Barbara, USA*, pages 216–233, 1994.

[19] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography and application to virus protection. In *27th Annual ACM Symposium on Theory of Computing (STOC 1995), Las Vegas, USA*, pages 45–56, 1995.

[20] Cong Wang, Qian Wang, Kui Ren, and Wenjing Lou. Ensuring data storage security in cloud computing. In *17th International Workshop on Quality of Service (IWQoS 2009), Charleston,USA*, pages 1–9, 2009.

[21] Shiyuan Wang, Divyakant Agrawal, and Amr El Abbadi. A comprehensive framework for secure query processing on relational data in the cloud. In *8th VLDB International Conference on Secure Data Management (SDM 2011), Berlin, Germany*, pages 52–69, 2011.

[22] Amos Beimel. Secret-sharing schemes: A survey. In *3rd International Conference on Coding and Cryptology (IWCC 2011), Qingdao, China*, pages 11–46, 2011.

[23] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[24] G. R. Blakley. Safeguarding cryptographic keys. In *National Computer Conference (AFIPS 1979), Monval, USA*, pages 313–317, 1979.

[25] C. Asmuth and J. Bloom. A modular approach to key safeguarding. *IEEE Transactions on Information Theory*, 29(2):208–210, March 1983.

[26] Sorin Iftene. General secret sharing based on the chinese remainder theorem with applications in e-voting. *Electronic Notes in Theoretical Computer Science*, 186: 67–84, July 2007.

[27] Lein Harn and Changlu Lin. Strong (n, t, n) verifiable secret sharing scheme. *Information Sciences*, 180(16):3059–3064, August 2010.

[28] Abhishek Parakh and Subhash Kak. Space efficient secret sharing for implicit data security. *Information Sciences*, 181(2):335–341, January 2011.

[29] Yan-Xiao Liu, Lein Harn, Ching-Nung Yang, and Yu-Qing Zhang. Efficient (n, t, n) secret sharing schemes. *Journal of Systems and Software*, 85(6):1325–1332, January 2012.

[30] Chou-Chen Yang, Ting-Yi Chang, and Min-Shiang Hwang. A (t,n) multi-secret sharing scheme. *Applied Mathematics and Computation*, 151(2):483–490, April 2004.

[31] Chao-Wen Chan and Chin-Chen Chang. A scheme for threshold multi-secret sharing. *Applied Mathematics and Computation*, 166(1):1–14, July 2005.

[32] Atsushi Waseda and Masakazu Soshi. Consideration for multi-threshold multi-secret sharing schemes. In *2012 International Symposium on Information Theory and its Applications (ISITA 2012), Honolulu, USA*, pages 265–269, 2012.

[33] Shi Runhual, Huang Liusheng, Luo yonglong, and Zhong Hong. A threshold multi-secret sharing scheme. In *IEEE International Conference on Networking, Sensing and Control (ICNSC 2008), Sanya, China*, pages 1705–1707, 2008.

[34] Satoshi Takahashi and Keiichi Iwamura. Secret sharing scheme suitable for cloud computing. In *27th international conference on advanced information networking and applications (AINA 2013), Barcelona, Spain*, pages 530–536, 2013.

[35] Bu ShanYue and Zhou Hong. A secret sharing scheme based on NTRU algorithm. In *Wireless Communications, Networking and Mobile Computing (WiCom 2009), Beijing, china*, pages 1–4, 2009.

[36] Ren-Junn Hwang and Chin-Chen Chang. An on-line secret sharing scheme for multi-secrets. *Computer Communications*, 21(13):1170–1176, September 1998.

[37] Dawei Zhao, Haipeng Peng, Cong Wang, and Yixian Yang. A secret sharing scheme with a short share realizing the (t,n) threshold and the adversary structure. *Computers and Mathematics with Applications*, 64(4):611–615, August 2012.

[38] Z. Eslami and J. Zarepour Ahmadabadi. A verifiable multi-secret sharing scheme based on cellular automata. *Information Sciences*, 180(15):2889–2894, August 2010.

[39] Jian-jie Zhao, Jianzhong Zhang, and Rong Zhao. A practical verifiable multi-secret sharing scheme. *Computer Standards and Interfaces*, 29(1):138–141, January 2007.

[40] Massoud Hadian Dehkordi and Samaneh Mashhadi. An efficient threshold verifiable multi-secret sharing. *Computer Standards and Interfaces*, 30(3):187–190, March 2008.

[41] Massoud Hadian Dehkordi and Samaneh Mashhadi. New efficient and practical verifiable multi-secret sharing schemes. *Information Sciences*, 178(9):2262–2274, May 2008.

[42] Guiqiang Chen, Huanwen Wang, Liqin Wang, and Yue Jin. A distributed multi-secret sharing scheme on the (t,n) threshold. In *2nd International Conference on Network Computing and Information Security (NCIS 2012), Shanghai, China*, pages 358–364, 2012.

[43] Chunqiang Hu, Xiaofeng Liao, and Xiuzhen Cheng. Verifiable multi-secret sharing based on LFSR sequences. *Theoretical Computer Science*, 445:52–62, August 2012.

[44] Shudong Li, Hong Lai, Wiaobo Wu, and Shuwu Jiang. Novel space efficient secret sharing for implicit data security. In *8th International Conference on Information Science and Digital Content Technology (ICIDT 2012), Jeju, Japan*, pages 283–286, 2009.

[45] Ting-Yi Chang, Min-Shiang Hwang, and Wei-Pang Yang. An improvement on the Lin-Wu (t,n) threshold verifiable multi-secret sharing scheme. *Applied Mathematics and Computation*, 163(1):169–178, April 2005.

[46] T.-Y.Lin and T.-C.Wu. (t,n) threshold verifiable multi secret sharing scheme based on factorisation intractability and discrete logarithm modulo a composite problems. *IEEE Computer and Digital Techniques*, 146(5):264–263, September 1999.

[47] Jun Shao and Zhenfu Cao. A new efficient (t,n) verifiable multi-secret sharing (VMSS) based on YCH scheme. *Applied Mathematics and Computation*, 168(1): 135–140, September 2005.

[48] Wei Chen, Xiang Long, Yuebin Bai, and Xiaopeng Gao. A new dynamic threshold secret sharing scheme from bilinear maps. In *International Conference on Parallel Processing Workshops (ICPPW 2007), Xi-An, China*, page 19, 2007.

[49] Angsuman Das and Avishek Adhikari. An efficient multi-use multi-secret sharing scheme based on hash function. *Applied Mathematics Letters*, 23(9):993–996, September 2010.

[50] Shiuh-Jeng Wang, Yuh-Ren Tsai, and Chien-Chih Shen. Verifiable threshold scheme in multi-secret sharing distributions upon extensions of ECC. *Wireless Personal Comunications*, 56(1):173–182, January 2011.

[51] Ziba Eslami and Saideh Kabiri Rad. A new verifiable multi-secret sharing scheme based on bilinear maps. *Wireless Personal Comunications*, 63(2):459–467, March 2012.

[52] Shanyue Bu and Ronggeng Yang. Novel and effective multi-secret sharing scheme. In *International Conference on Information Engineering and Applications (IEA 2012)*, pages 461–467, 2013.

[53] Shanyue Bu and Ronggeng Yang. Novel and effective multi-secret sharing scheme. In *2nd International Conference on Information Engineering and Applications (IEA 2012), Dalian, China*, pages 461–467, 2012.

[54] Fatih Emekci, Divyakant Agrawal, and Amr El Abbadi. [abacus.

[55] Fatih Emekci, Divyakant Agrawal, Amr El Abbadi, and Aziz Gulbeden. Privacy preserving query processing using third parties. In *22nd IEEE International Conference on Data Engineering (ICDE 2006), Atlanta, USA*, pages 27–37, 2006.

[56] Divyakant Agrawal, Amr El Abbadi, Fatih Emekci, and Ahmed Metwally. Database management as a service: Challenges and opportunities. In *25th IEEE International Conference on Data Engineering (ICDE 2009), Shanghai, China*, pages 1709–1716, 2009.

[57] Mohammad Ali Hadavi, Ernesto Damiani, Rasool Jalili, Stelvio Cimato, and Zeinab Ganjei. AS5: A secure searchable secret sharing scheme for privacy preserving database outsourcing. In *ESORICS DPM/SETOP 2012 International Workshops, Pisa, Italy*, pages 201–216, 2012.

[58] Brian Thompson, Stuart Haber, William G. Horne, Tomas Sander, and Danfeng Yao. Privacy-preserving computation and verification of aggregate queries on outsourced databases. In *9th International Symposium on Privacy Enhancing Technologies (PETS 2009), Seattle, USA*, pages 185–201, 2009.

[59] Mohammad Ali Hadavi and Rasool Jalili. Secure data outsourcing based on threshold secret sharing: Towards a more practical solution. In *VLDB 2010 PhD Workshop, Singapore*, pages 54–59, 2010.

[60] Mohammad Ali Hadavi, Morteza Noferesti, Rasool Jalili, and Ernesto Damiani. Database as a service: Towards a unified solution for security requirements. In *36th IEEE Annual Conference on Computer Software and Applications Conference Workshops (COMPSACW 2012), Izmir, Turkey*, pages 415–420, 2012.

[61] Graham Cormode and Divesh Srivastava. Anonymized data: Generation, models, usage. In *26th IEEE International Conference on Data Engineering (ICDE 2010), Long Beach, USA*, pages 1015–1018, 2010.

[62] Erin E. Kenneally and Kimberly Claffy. Dialing privacy and utility: A proposed data-sharing framework to advance internet research. *IEEE Security and Privacy*, 8(4):31–39, July/August 2010.

[63] Latanya Sweeney. K-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, October 2002.

[64] Latanya Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):571–588, October 2002.

[65] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *ACM SIGMOD international conference on Management of data (SIGMOD 2005), Baltimore, USA*, pages 49–60, 2005.

[66] Rinku Dewri and Indrajit Ray. K-anonymization in the presence of publisher preferences. *IEEE Transactions on Knowledge and Data Engineering*, 23(11):1678–1690, November 2011.

[67] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), Mar 2007.

[68] Jeff Sedeyao. Enhancing cloud security using data anonymization; intel report, 2012. URL `http://www.intel.ie/content/www/ie/en/it-management/intel-it-best-practices/enhancing-cloud-security-using-data-anonymization.html`.

[69] Prasanna Padmanabhan, Le Gruenwald, Anita Vallur, and Mohammed Atiquzzaman. A survey of data replication techniques for mobile ad hoc network databases. *The VLDB Journal*, 17(5):1143–1164, August 2008.

[70] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *SIAM Journal of Applied Math*, 8(2):300–304, June 1960.

[71] J. Thomas and E. Schwarz. Generalized reed solomon codes for erasure correction in SDDS. In *Workshop on Distributed Data and Structures (WDAS 2002), Paris, France*, pages 75–86, 2002.

[72] Mohamed Helmy Megahed, Dimitrios Makrakis, and Bidi Ying. SurvSec: A new security architecture for reliable network recovery from base station failure of surveillance [wsn.

[73] Witold Litwin and Rim Moussa. LH: A highly available distributed data storage. In *30th International Conference on Very Large Data Bases (VLDB 2004), Toronto, Canada*, page 1289–1292, 2004.

[74] Hovav Shacham and Brent Waters. Compact proofs of retrievability. In *14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT 2008), Melbourne, Australia*, pages 90–107, 2008.

[75] Kevin D. Bowers, Ari Juels, and Alina Oprea. Proofs of retrievability: Theory and implementation. In *ACM Workshop on Cloud Computing Security (CCSW 2009), Chicago, USA*, pages 43–54, 2009.

[76] Juan A. Garay, Rosario Gennaro, Charanjit Jutla, and Tal Rabin. Secure distributed storage and retrieval. *Theoretical Computer Science*, 243(1-2):363–389, July 2000.

[77] Ari Juels and Burton S. Kaliski Jr. PORs: Proofs of retrievability for large files. In *14th ACM conference on Computer and Communications Security (CCS 2007), Alexandria, USA*, pages 584–597, 2007.

[78] Qian Wang, Cong Wang, Jin Li, Kui Ren, and Wenjing Lou. Enabling public verifiability and data dynamics for storage security in cloud computing. In *14th European Conference on Research in Computer Security (ESORICS 2009), Saint-Malo, France*, pages 355–370, 2009.

[79] Kevin D. Bowers, Ari Juels, and Alina Oprea. HAIL: A high-availability and integrity layer for cloud storage. In *16th ACM conference on Computer and Communications Security (CCS 2009), Chicago, USA*, pages 187–198, 2009.

[80] Cong Wang, Sherman S.M. Chow, Qian Wang, Kui Ren, and Wenjing Lou. Privacy-preserving public auditing for secure cloud storage. *IEEE Transactions on Computers*, 62(2):362–375, December 2011.

[81] J. He and E. Dawson. Multistage secret sharing based on one-way function. *Electronics Letters*, 30(19):1591–1592, September 1994.

[82] Torben P. Pedersen. A threshold cryptosystem without a trusted party. In *Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1991), Brighton, UK*, pages 522–526, 1991.

[83] Tang Chunming and Zheng an Yao. A new (t,n) threshold secret sharing scheme. In *International Conference on Advanced Computer Theory and Engineering (ICACTE 2008), Phuket, Thailand*, pages 920–924, 2008.

[84] Amir Herzberg, Stanisław Jarecki, Hugo Krawczyk, and Moti Yung. Proactive secret sharing or: How to cope with perpetual leakage. In *15th Annual International Cryptology Conference (CRYPTO 1995), Santa Barbara, USA*, pages 339–352, 1995.

[85] Josh Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *8th Annual International Cryptology Conference (CRYPTO 1988), Santa Barbara, USA*, pages 27–35, 1990.

[86] Mehrdad Nojoumian, Douglas R. Stinson, and Morgan Grainger. Unconditionally secure social secret sharing scheme. *Information Security, IET*, 4(4):202–211, December 2010.

[87] Mehrdad Nojoumian and Douglas R. Stinson. Socio-rational secret sharing as a new direction in rational cryptography. In *3rd International Conference Decision and Game Theory for Security (GameSec 2012), Budapest, Hungary*, pages 18–37, 2012.

[88] Tao Zheng, Haitong Wu, Hao Wen Lin, and Jeng-Shyang Pan. Application of belief learning model based socio-rational secret sharing scheme on cloud storage. In *6th*

International Conference on Genetic and Evolutionary Computing (ICGEC 2012), Kitakyushu, Japan, pages 15–18, 2012.

[89] Mehrdad Nojoumian and Douglas R. Stinson. Social secret sharing in cloud computing using a new trust function. In *10th Annual International Conference on Privacy, Security and Trust (PST 2012), Paris, france*, pages 161–167, 2012.

[90] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *27th annual symposium on foundations of computer science, Toronto, Canada*, pages 162–167, 1986.

[91] Chumming Tang and Zheng an Yao. Definition and construction of multi-prover zero-knowledge arguments. In *International conference on communications and mobile computing (CMC 2009), Yunnan, China*, pages 375–379, 2009.

[92] Guang Gong and L. Harn. Public-key cryptosystems based on cubic finite field extensions. *IEEE Transactions on Information Theory*, 45(7):2601–2605, November 1999.

[93] Guang Gong, Lein Harn, and Huapeng Wu.

[94] Patrick O'Neil, Elizabeth O'Neil, Xuedong Chen, and Stephen Revilak. The star schema benchmark and augmented fact table indexing. In *1st Technology Conference on Performance Evaluation and Benchmarking (TPCTC 2009), Lyon, France*, pages 237–252, 2009.

[95] Hotea Solutions. TPC-H, 2015. URL `http://www.tpc.org/tpch/`.

[96] Hotea Solutions. TPC-DS, 2015. URL `http://www.tpc.org/tpcds/`.