جامعة سيدي محمد بن عبد الله بفاس
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵉⴷⵉ ⵎⵓⵃⵎⴷ ⴱⵏ ⵄⴱⴷⵍⵍⴰⵀ ⴱ ⴼⴰⵙ
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH DE FES

كلية العلوم والتقنيات فاس
ⵜⴰⵖⵢⴻⵍⵜ ⵏ ⵜⵜⵓⵜⵜⵓⵔⵉⵏ ⴰ ⵜⵓⵔⵉⵔⵜ ⵏ ⵜⵓⵙⵙⵉⵔⵜⵉⵏ
FACULTÉ DES SCIENCES ET TECHNIQUES DE FÈS

Centre d'Etudes Doctorales : Sciences et Techniques de l'Ingénieur

N° d'ordre *17/2020*

THESE DE DOCTORAT

Présentée par

Mr : Imad EL GHOUBACH

Spécialité : **Informatique**

Sujet de la thèse : **La Sécurité des Données dans le Cloud Computing**

Formation Doctorale : Sciences de l'ingénieur Sciences Physiques, Mathématiques et Informatique

Thèse présentée et soutenue le 15 juillet 2020 devant le jury composé de :

Nom Prénom	Titre	Etablissement	
Arsalane ZARGHILI	PES	Faculté des Sciences et Techniques Fès	Président
Sabir ESSAID	PH	Ecole Nationale Supérieure d'Electricité et Mécanique Casablanca	Rapporteur
Hussain BEN-AZZA	PH	Ecole Nationale Supérieure d'Arts et Métiers Meknès	Rapporteur
Azeddine ZAHI	PH	Faculté des Sciences et Techniques Fès	Rapporteur
Said NAJAH	PH	Faculté des Sciences et Techniques Fès	Examinateur
Khalid ZENKOUAR	PH	Faculté des Sciences et Techniques Fès	Examinateur
Fatiha MRABTI	PH	Faculté des Sciences et Techniques Fès	Directeurs de thèse
Rachid BEN ABBOU	PH	Faculté des Sciences et Techniques Fès	

Laboratoire d'accueil : **Systèmes Intelligents et Applications**

Etablissement : **Faculté des Sciences et Techniques de Fès**

Centre d'Etudes Doctorales : Sciences et Techniques de l'Ingénieur

Résumé de la thèse

Le cloud computing s'est largement développé au cours de ces dernières années. Cette technologie a acquis une grande popularité dans le domaine informatique. Ceci revient à la capacité des fournisseurs de cloud d'assurer une évolutivité et des ressources « infinies » avec un accès distribué et à la demande. L'accès à distance aux services cloud permet aux utilisateurs du cloud d'ignorer l'existence de l'infrastructure informatique qui la sous-tend. En raison de la virtualisation et de la multi-localisation, l'infrastructure cloud devient de plus en plus complexe et invisible par rapport aux centres de données traditionnels. Cette ambiguïté rend la gestion de la fourniture, la surveillance, la sauvegarde, la reprise après incident des services et en particulier la gestion de la sécurité plus compliquée. Pour cette raison, il y a un manque croissant de confiance dans les infrastructures cloud.

Cette dissertation vise à surmonter ce dilemme, tout en tenant compte de deux préoccupations de sécurité des données, à savoir; la confidentialité des données et l'intégrité des données.

En premier lieu, nous nous focalisons sur le problème de confidentialité des données, un enjeu assez compliqué à cause du partage de données flexible dans un groupe dynamique d'utilisateurs. Pour répondre à cette préoccupation, nous avons, d'une part, proposé une nouvelle amélioration de la méthode P2E, reposant sur l'utilisation de la cryptographie basée sur les attributs, où nous avons étudié l'impact de l'utilisation d'une structure d'accès plus générale sur la performance et l'évolutivité de partage. Grâce aux propriétés de la structure d'accès à seuil, cette contribution a démontré sa capacité de gérer des accès assez complexes, tout en ayant un impact négligeable sur la performance. D'autre part, nous définissons ESS-ODER, une solution à base de cryptographie basée sur les attributs, qui propose la délégation de la tâche de déchiffrement vers un serveur proxy pour répondre aux limitations de certains terminaux à puissance limitée. En effet, ESS-ODER permet un partage sécurisé des données dans un environnement cloud tout en maintenant une gestion de contrôle d'accès flexible.

Deuxièmement, nous abordons le problème de la preuve de possession de données (PDP). En fait, le client cloud doit disposer d'une méthode efficace pour effectuer des vérifications d'intégrité périodiques à distance sans conserver les données localement.

Afin de satisfaire cette exigence de sécurité, nous définissons SERDAS, une solution fondée sur le chiffrement basé sur l'identité, qui permet à l'utilisateur de déléguer, d'une manière sécurisée, la vérification de ses données à un auditeur tiers. Cette méthode assure l'intégrité des données tout en ayant un nombre illimité de vérifications.

Mots clés : Cloud Computing ; Stockage sécurisé des données ; Confidentialité ; Intégrité ; Chiffrement basé sur les attributs ;

Abstract

Cloud computing has grown significantly in recent years. This technology has gained great popularity in the computer field. This boils down to the ability of cloud providers to provide "infinite" scalability and resources with distributed and on-demand access. Remote access to cloud services allows cloud users to overlook the existence of the IT infrastructure behind it. Due to virtualization and multi-tenancy, cloud infrastructure is becoming increasingly complex and invisible compared to traditional data centers. This ambiguity makes the management of service provision, monitoring, backup, disaster recovery and in particular security management more complicated. For this reason, there is a growing lack of confidence in cloud infrastructures.

This thesis aims to overcome this dilemma, while taking into account two data security concerns, namely data confidentiality and data integrity.

Firstly, we are focusing on the problem of data confidentiality, a fairly complicated issue due to the flexible sharing of data in a dynamic group of users. To address this concern, we have, on one hand, proposed a further improvement of a P2E method, based on the use of attribute-based cryptography, where we have studied the impact of using more general access structure on sharing performance and scalability. Thanks to the properties of the threshold access structure, this contribution has demonstrated its ability to manage fairly complex accesses, while having a negligible impact on performance. On the other hand, we define ESS-ODER, a solution built on attribute-based cryptography, which offers the delegation of the decryption task to a proxy server to deal with the power limitation of certain terminals. In fact, ESS-ODER allows secure sharing of data in a cloudy environment while maintaining flexible access control management.

Secondly, we address the problem of proof of data possession (PDP). In fact, the cloud client must have an efficient and secure method to perform periodic integrity checks remotely without storing the data locally. In order to meet this security requirement, we define SERDAS, a solution built on identity-based encryption, allowing

users to delegate the verification of their data, in a secure manner, to a third party auditor. This method provides a probabilistic guarantee of data integrity while having an unlimited number of verifications.

Résumé

Le cloud computing s'est largement développé au cours de ces dernières années. Cette technologie a acquis une grande popularité dans le domaine informatique. Ceci revient à la capacité des fournisseurs de cloud d'assurer une évolutivité et des ressources « infinies » avec un accès distribué et à la demande. L'accès à distance aux services cloud permet aux utilisateurs du cloud d'ignorer l'existence de l'infrastructure informatique qui la sous-tend. En raison de la virtualisation et de la multi-localisation, l'infrastructure cloud devient de plus en plus complexe et invisible par rapport aux centres de données traditionnels. Cette ambiguïté rend la gestion de la fourniture, la surveillance, la sauvegarde, la reprise après incident des services et en particulier la gestion de la sécurité plus compliquée. Pour cette raison, il y a un manque croissant de confiance dans les infrastructures cloud.

Cette dissertation vise à surmonter ce dilemme, tout en tenant compte de deux préoccupations de sécurité des données, à savoir ; la confidentialité des données et l'intégrité des données.

En premier lieu, nous nous focalisons sur le problème de confidentialité des données, un enjeu assez compliqué à cause du partage de données flexible dans un groupe dynamique d'utilisateurs. Pour répondre à cette préoccupation, nous avons, d'une part, proposé une nouvelle amélioration de la méthode P2E, reposant sur l'utilisation de la cryptographie basée sur les attributs, où nous avons étudié l'impact de l'utilisation d'une structure d'accès plus générale sur la performance et l'évolutivité de partage. Grâce aux propriétés de la structure d'accès à seuil, cette contribution a démontré sa capacité de gérer des accès assez complexes, tout en ayant un impact négligeable sur la performance. D'autre part, nous définissons ESS-ODER, une solution à base de cryptographie basée sur les attributs, qui propose la délégation de la tâche de déchiffrement vers un serveur proxy pour répondre aux limitations de certains terminaux à puissance limitée. En effet, ESS-ODER permet un partage sécurisé des données dans un environnement cloud tout en maintenant une gestion de contrôle d'accès flexible.

Deuxièmement, nous abordons le problème de la preuve de possession de données (PDP). En fait, le client cloud doit disposer d'une méthode efficace pour effectuer des vérifications d'intégrité périodiques à distance sans conserver les données localement. Afin de satisfaire cette exigence de sécurité, nous définissons SERDAS, une solution fondée sur le chiffrement basé sur l'identité, qui permet à l'utilisateur de déléguer, d'une manière sécurisée, la vérification de ses données à un auditeur tiers. Cette méthode assure l'intégrité des données tout en ayant un nombre illimité de vérifications.

———-

Remerciements

De nombreuses personnes ont contribué d'une manière ou d'une autre à la réalisation de ce travail. Certaines de ces personnes viennent même à l'improviste dans nos vies pour nous donner un mot de courage ou simplement pour nous écouter quand nous sommes déprimés ou quand nous ne trouvons pas de réponse à nos multiples questions. Je tiens à vous remercier tous du fond du cœur.

Je tiens à exprimer ma plus sincère gratitude et mes remerciements à Prof. **Fatiha MRABTI** et Prof. **Rachid BEN ABBOU**, mes directeurs de thèse, pour leurs soutien, confiance et conseils tout au long de ma thèse. J'ai trouvé dans mes directeurs le soutien dont un doctorant a besoin. Sans aucun doute, leur influence dans ma vie a largement contribué à ce que j'ai accompli aujourd'hui.

Je remercie ensuite l'ensemble des membres du jury, qui m'ont fait l'honneur de bien vouloir étudier avec attention mon travail : Prof. **Sabir ESSAID**, Prof. **Hussain BEN-AZZA** et Prof. **Azeddine ZAHI** pour avoir accepté d'être rapporteurs de cette thèse, Prof. **Said NAJAH** et Prof. **Khalid ZENKOUAR** pour avoir accepté d'examiner cette thèse, enfin Prof. **Arsalane ZARGHILI** pour m'avoir fait l'honneur d'accepter de présider ce jury.

Je suis éternellement redevable à mes parents aimant et à tous les membres de ma famille, en particulier mon père **Mohammed**, ma mère **Zahra** et mes charmantes sœurs. ils ont essayé avec empressement d'offrir les meilleures conditions, des soins généreux, un dévouement et un soutien pour réaliser ma thèse. Merci de toujours montrer la fierté sur leurs visages tout en faisant référence à moi et à mes réalisations.

Je voudrais exprimer ma plus profonde gratitude à ma femme **Fatima zahra** pour ses encouragements et son sacrifice sans lesquels je ne pourrais pas avoir les forces nécessaires pour surmonter toutes les difficultés.

Je souhaite également remercier vivement mes grands-parents, mes beaux-parents et le reste de ma famille pour leur soutien toute au long de mon trajet.

J'adresse un grand merci à mes chers amis **Yahya**, **Ali**, pour leurs encouragements, leurs compagnies précieuses et surtout pour leurs soutiens tout le long de mes années de thèse.

Merci à mes collègues pour tout le temps passé ensemble, les réunions d'équipe et les discussions. Ce fut une grande expérience de vous avoir tous autour de moi.

Table des matières

Table des figures

Liste des tableaux

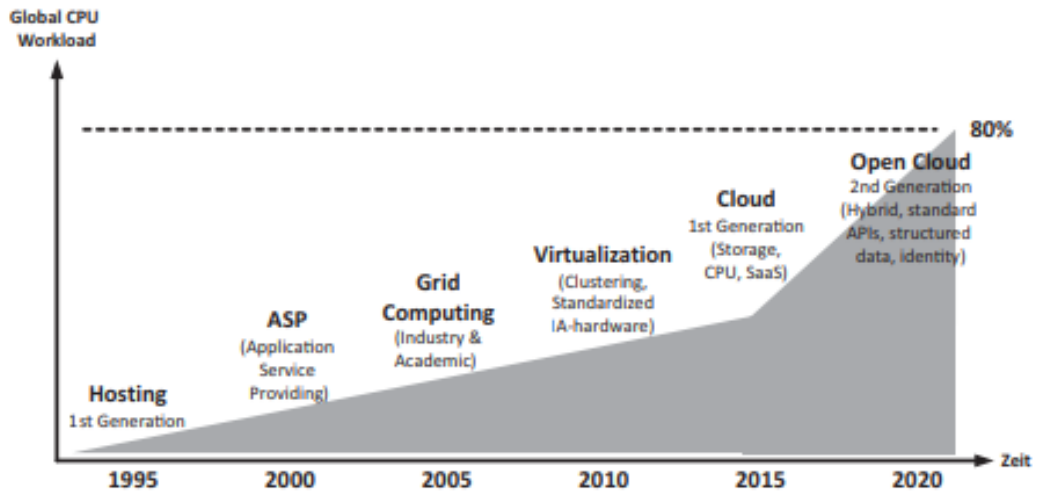
Liste des Acronymes

3DES	Triple Data Encryption Standard.
AA	Autorité d'attributs.
ABE	Chiffrement basé sur les attributs.
AES	Advanced Encryption Standard.
CP-ABE	Ciphertext-policy attribute based encryption.
CSP	Fournisseur de services.
DES	Data Encryption Standard.
EC	Courbes Elliptiques.
ECC	Cryptographie basé sur les courbes elliptiques.
IAAS	Infrastructure en tant que Service (Infrastracture as a Service).
IBC	Cryptographie basé sur l'identité.
ID	Identité.
IDE	Chiffrement basé sur l'identité.
KP-ABE	Key-policy attribute based encryption (KP-ABE).
PAAS	Platforme en tant que Service (Platforme as a Service).
PDP	possession de données prouvable.
PKG	Générateur privé de clés.
POP	Preuve de possession.
POR	Preuve de récupérabilité.
RDPC	la vérification des possessions de données à distance (Remote Data possession checking).
SAAS	Logiciel en tant que Service (Software as a Service).
TPA	Auditeur externe (Third party auditor).

Introduction Générale

Les services de Cloud Computing sont apparus, pour la première fois, au début des années 2000, en particulier pour les grandes entreprises. Depuis, cette technologie a connu une popularité croissante au niveau des petites et moyennes entreprises. Cette popularité est liée à la capacité de cette technologie de répondre, avec un coût minime, aux besoins croissants de la digitalisation des données et de l'utilisation des systèmes d'information qui nécessitent une capacité de stockage de plus en plus importante (figure 1) [50].

Le terme "Cloud Computing" , proposé par RAMNATH K. CHELLAPPA, lors d'une conférence à Dallas en 1997 [19], fait référence à un nouveau concept, relativement parlant, qui englobe plusieurs services promettant d'être la solution à tous les problèmes de capacité et de performance [81, 24].



La croissance de l'utilisation du Cloud Computing [50]

Actuellement, les organisations adoptent, de plus en plus, des services de cloud publique pour stocker (Ex : Microsoft Skydrive et Dropbox) et traiter leurs données (Ex : Amazon EC2 et MapReduce Framework). En fait, selon GARTNER le marché des services cloud continue toujours de se développer (une croissance de 17% de revenus dans le marché mondial est projeté en 2020) . L'utilisation de la technologie cloud est

également déterminante pour favoriser le partage de données entre plusieurs organisations (par exemple, les organisations gouvernementales) ainsi que le travail collaboratif distant Inter-entreprise.

Cependant, des problèmes critiques de la confidentialité des données [87, 38, 51, 82] limitent l'adoption généralisée du Cloud, en particulier les Clouds publiques. Il est primordial que les données sensibles stockées doivent être partagées de manière sélective entre différents utilisateurs que ce soit en inter ou intra organisation. En effet, le partage de ce type de données doit être basé sur une politique de contrôle d'accès définie par l'organisation propriétaire des données partagées.

Contexte

Les données, stockées dans le Cloud, peuvent être transférées entre des centres de données situés dans différentes zones géographiques (ou même pays). Les utilisateurs du Cloud n'ont pas beaucoup d'informations sur l'endroit où leurs données sont stockées et traitées, ce qui rend la tâche de garantir la conformité de la confidentialité et de l'intégrité des données, lors des opérations quotidiennes, très difficile pour les fournisseurs de services Cloud (CSP) et leurs clients. D'où la nécessité de développer des méthodes permettant de répondre aux risques suivants :

Confidentialité des données : L'externalisation des données vers des serveurs distants rend la confidentialité des données un problème difficile et conflictuel. Les serveurs distants sont contrôlés et gérés par d'éventuels fournisseurs de services Cloud (CSP) semi-fiables. L'utilisation du chiffrement des données peut être une solution à la problématique de confidentialité. Néanmoins, cette approche soulève plusieurs problèmes de gestion, tels que le stockage et la gestion des clés de chiffrement/déchiffrement du côté client.
De plus, la préservation de la confidentialité devient plus compliquée avec un partage flexible des données au sein d'un groupe d'utilisateurs. Ce partage nécessite une gestion efficace des clés de déchiffrement entre différents utilisateurs autorisés.
Le défi est de définir une méthode de révocation qui ne nécessite pas de mettre à jour les clés secrètes des utilisateurs non révoqués. En plus, les politiques de contrôle d'accès devraient être flexibles et capables de distinguer parmi les utilisateurs ceux qui ont le droit d'accès aux données. En d'autres termes, les données peuvent être partagées par différents utilisateurs ou groupes, et les utilisateurs peuvent appartenir à plusieurs groupes.

Intégrité des données : Pour augmenter la robustesse des services de stockage, les fournisseurs de services optent pour la répartition des données sur plusieurs serveurs de stockage. Une telle distribution offre une résilience contre les problèmes liés au matériel. Néanmoins, afin de réduire les coûts d'exploitation et d'économiser les capacités de stockage, les fournisseurs malhonnêtes pourraient tenter d'alléger les procédures de réplication, cela peut entraîner des erreurs de données irrécupérables ou même des pertes totales de données. En plus, les clients n'ont aucun moyen technique pour vérifier que le stockage est fait de manière correcte et intègre.

Objectifs

Pour relever les défis susmentionnés, nous nous sommes fixés les objectifs suivants :

— **Objectif 1** Améliorer la confidentialité et la gestion de contrôle d'accès aux données dans les environnements de stockage cloud tout en optimisant la flexibilité du partage dynamique entre les utilisateurs. En effet, les mécanismes de sécurité proposés devraient garantir à la fois la robustesse et l'efficacité. En plus, il est nécessaire d'avoir un processus de révocation d'accès des utilisateurs avec un coût de calcul minimal pour tous les acteurs du système.

— **Objectif 2** Minimiser le coût de stockage et de partage des données chez les utilisateurs avec accès à des terminaux ayant des puissances limitées.

— **Objectif 3** Faire face au problème de vérification d'intégrité des données dans les environnements de stockage Cloud, en se focalisant sur trois aspects importants : sécurité, possibilité de délégation de la vérification d'intégrité et performance, tout en maintenant un coût de stockage et de traitement optimal pour les utilisateurs dotés de terminaux ayant des puissances limitées.

Contributions

Lors de la définition des solutions de confidentialité et d'intégrité des données externalisées, les aspects suivants sont pris en compte : la flexibilité des méthodes, la robustesse de la sécurité et la réduction du coût de calcul chez les différents acteurs.

En considérant les objectifs cernés dans la section précédente, nous avons proposé les contributions majeures ci-dessous :

- **Contribution 1 :** Nous Proposons une méthode que nous avons abrégé FP2E [33]. la méthode est une amélioration de P2E [30], qui opte à maximiser la flexibilité

des politiques d'accès. Dans notre approche nous utilisons les structures d'accès monotones pour permettre aux utilisateurs de définir un niveau complexe de contrôle d'accès avec un nombre minimal d'attributs. Nous proposons aussi une méthode de révocation des droits qui permet de gérer (attribuer/révoquer) les droits d'accès des utilisations avec un impact minimal sur les différents acteurs du système.

- **Contribution 2 :** Nous proposons ESS-ODER [39], une nouvelle approche à clé publique permettant d'assurer la confidentialité des données tout en maintenant le partage dynamique entre les utilisateurs dans les environnements distribués avec un fardeau faible pour les utilisateurs du service de stockage. Dans cette approche, nous proposons d'externaliser le calcul de l'opération de déchiffrement vers un serveur de calcul proxy ce qui permet de minimiser le coût de calcul de l'opération de déchiffrement chez les clients. En plus, nous étudions l'impact de l'utilisation du concept de parallélisme sur l'optimisation du temps de calcul et de gestion des ressources de tous les acteurs du système.
- **Contribution 3 :** Proposition d'une approche d'audit de données efficace et sécurisée [31]. Dans notre approche, nous avons donné la possibilité au propriétaire des données de déléguer la vérification de l'intégrité de ses données chez un auditeur tiers tout en maintenant la sécurité des données auditées. Nous avons aussi donné la possibilité à l'auditeur tiers de faire une vérification groupée des données de différents utilisateurs afin de minimiser le coût et de maximiser la performance et l'efficacité du système.

Organisation du mémoire

Ce mémoire est composé de quatre chapitres principaux :

- **Chapitre 1 -Généralités du Cloud Computing :** Ce chapitre introduit le Cloud Computing y compris la définition, l'architecture, les modèles de déploiement et les services.
- **Chapitre 2 -État De L'art :** Dans ce chapitre, nous discutons les concepts généraux de la cryptographie. D'une part, nous décrivons les concepts de base de chiffrement : la cryptographie symétrique, cryptographie à clé publique, et particulièrement le chiffrement basés sur les attributs. D'une autre part nous discutons le concept de la vérification probabiliste de l'intégrité à fin d'introduire les concepts de bases utilisée pour la conception des méthodes discutées tout au long de ce rapport.

- **Chapitre 3 -Confidentialité et Gestion de contrôle d'accès :** Dans ce chapitre, nous explorons l'impact de l'utilisation d'un type de structure d'accès plus flexible sur la performance et la flexibilité des méthodes basées sur les attributs (Contribution 1), puis nous proposons une méthode permettant le partage flexible et confidentiel des données tout en maintenant un fardeau de calcul minimal chez les clients du service de stockage (Contribution 2).
- **Chapitre 4 -Vérification des données a distance :** Ce chapitre décrit notre méthode d'audit proposée (contribution 3). Au cours de ce chapitre nous présentons une méthode qui permet aux utilisateurs des services de stockage de garantir l'intégrité de leurs données en sollicitant un service d'audit tiers sans divulguer aucune information sur les données stockées.
- **Conclusion :** Dans la conclusion nous présentons une synthèse de nos travaux avec une description de nos contributions et les futures perspectives.

Chapitre I

Généralités du Cloud Computing

Sommaire

Le paradigme du cloud computing n'est pas nouveau et peut être considéré comme une extension de la façon dont nous utilisons l'internet. Dans ce chapitre, nous allons présenter les différentes caractéristiques du cloud computing et les différents problèmes de sécurité liés a ce type de plateforme de services.

I.1 Cloud Computing

I.1.1 Définition

Le Cloud Computing peut être défini comme un nouveau paradigme dans lequel les fournisseurs de services offrent des ressources évolutives de manière dynamique et souvent virtualisée. Cette Technologie est devenue une tendance dans laquelle les utilisateurs utilisent divers appareils, notamment des ordinateurs personnels, des ordinateurs portables, des smart-phones et des assistants numériques personnels, pour accéder aux services offerts par un fournisseur de services que ce soit des services applicatifs, services de stockage ou plates-formes de développement d'applications via internet. Les avantages de cette technologie sont les suivants : réduction des coûts, haute disponibilité et évolutivité.

Sur un plan économique, le Cloud offre aux utilisateurs la possibilité de réserver, utiliser les ressources dont ils ont besoin et d'être facturés seulement sur l'utilisation réelle de services sans avoir un souci de maintenance ni de gestion. Ces ressources sont disponibles et accessibles via l'internet à tout moment.

Le Cloud Computing est formellement défini par l'institut national des normes et de la technologie (NIST) [75] comme un modèle permettant un accès réseau pratique et omniprésent à la demande à un pool partagé de ressources informatiques configurables (réseaux, serveurs, stockage, applications et services). Ces ressources pouvant être rapidement réservées, utilisées et libérées avec un effort minimal de gestion et d'interaction du fournisseur de services.

I.1.2 Caractéristiques

Le Cloud Computing peut être vu comme un système d'exploitation distribué sur des milliers de machines assurant l'abstraction de l'infrastructure et permettant l'hébergement et l'exécution des applications sous forme de services tels que : le stockage, calcul, traitement de données etc.

Généralement le Cloud Computing est défini par cinq caractéristiques essentielles [75] :

Ressources partagées : le fournisseur met en commun les ressources informatiques (stockages, mémoire, CPU, bande passante, machine virtuelle, etc.) afin de satisfaire plusieurs utilisateurs. Grâce à la technologie de virtualisation, les ressources physiques du fournisseur de services sont virtualisées et partagées entre plusieurs utilisateurs. En effet, ces ressources peuvent être affectées et réaffectées en fonction de la demande des utilisateurs. Ceci induit une méconnaissance de la localisation exacte des ressources utilisées.

Accès large via le réseau : les différents services d'un fournisseur de Cloud ainsi que les ressources réservées par un utilisateur sont disponibles et accessibles à travers une large gamme de périphériques connectés au réseau.

Dimensionnement rapide : les ressources peuvent être dimensionnées rapidement à la hausse ou à la baisse selon la demande de l'utilisateur.

Services à la demande : l'utilisateur peut réserver sans aucune interaction humaine avec le fournisseur des ressources de calcul et de stockage selon ses besoins.

Un service mesuré et facturé à l'usage : le Cloud mesure automatiquement l'utilisation des ressources faite par un utilisateur afin de lui permettre de calculer automatiquement sa facture.

I.1.3 Modèles de déploiements

L'institut national des normes et de la technologie (NIST) [3] distingue quatre modèles de déploiement pour le Cloud :

— Cloud Publique : le fournisseur offre des services facturés à l'utilisation. Le fournisseur est responsable de la sécurité, ce qui donne aux utilisateurs un faible degré de contrôle et de surveillance sur les deux aspects physique et logique du Cloud.

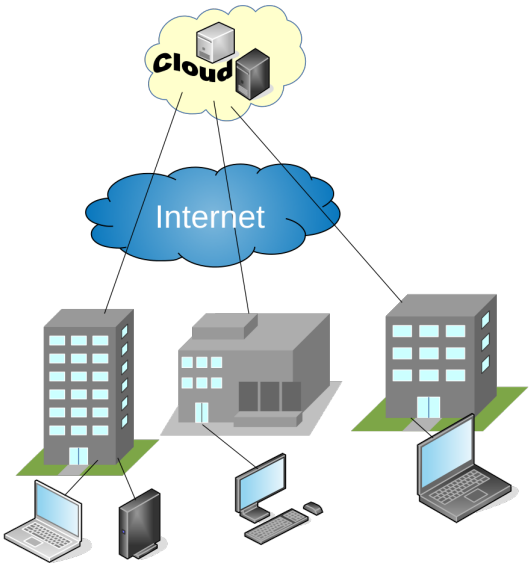


FIGURE I.1 – Cloud publique [25]

— Cloud Privé : dans ce type de déploiement les services ne sont accessibles que par l'organisation pour laquelle il est dédié, cela veut dire que toutes les infrastructures (réseaux, applications, stockage...) associées à ce modèle sont réservées et gérées par l'organisation.

— Cloud Hybride : un environnement Cloud avec des fournisseurs internes et externes. Avec cette architecture, une organisation peut déployer des applications

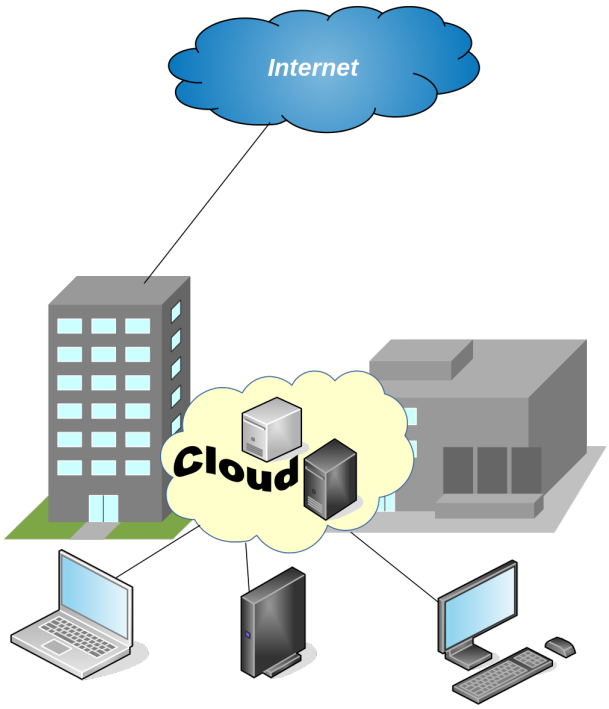


FIGURE I.2 – Cloud privé [25]

sur un Cloud publique tout en conservant la confidentialité de ses données et ses applications dans le Cloud privé.

— Cloud communautaire : dans ce type de déploiement, un ensemble d'utilisateurs avec des intérêts communs peuvent mettre en place un Cloud privé à utilisation partagée par les membres de la communauté. Par exemple, un groupe d'universités peuvent décider d'opérer et d'interconnecter leurs infrastructures informatique et de créer un Cloud communautaire auquel chacun de ses membres peut accéder [86, 75].

I.1.4 Modèles de services

Les services du Cloud Computing sont classés sous trois principaux modèles de services : Logiciel en tant que service (Software-as-a-Service(SaaS)), Plateforme en tant que service (Platform as-a-Service (PaaS)) et Infrastructure en tant que service (Infrastructure-as-a-Service (IaaS)) [74]

— **Logiciel en tant que service (SaaS)** : l'utilisateur contrôle partiellement la configuration d'applications fonctionnant sur une infrastructure du Cloud, comme Google Apps. Ces applications sont accessibles à travers un grand nombre de périphériques client (téléphone portable, ordinateur, etc.) par le biais d'un interface Web.

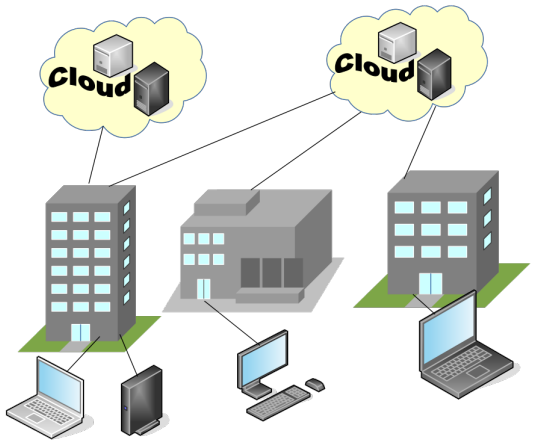


FIGURE I.3 – Cloud Hybride [25]

— **Plateforme en tant que service (PaaS) :** le fournisseur de service met à la disposition des utilisateurs une plateforme spécifique qui leur permet d'exécuter des applications propriétaires. Ils ont aussi un contrôle total sur leurs applications déployées et la configuration associée.

— **Infrastructure en tant que service (IaaS) :** pour ce modèle, les ressources sont gérées, agrégées et louées aux utilisateurs, comme étant des capacités de stockage (par exemple, service Amazon Simple Storage (S3) [23]), des ressources de réseau, ou des capacités de calcul (par exemple, Amazon Elastic Compute Cloud (EC2) [111]). L'utilisateur peut alors exécuter le système d'exploitation de son choix et des logiciels qui répondent au mieux à ses besoins, sans être en mesure de gérer ou contrôler l'infrastructure Cloud sous-jacente.

I.2 Problèmes de la sécurité dans le Cloud

Avec le développement rapide des différentes technologies de virtualisations, calcul et transmission, de nouvelles variantes de modèles de déploiement émergent pour répondre aux diverses demandes et exigences d'utilisateurs. Un exemple de ces

variantes est celui d'un cloud privé virtuel[54].

Avec l'émergence de technologies d'accès réseau haut débit (comme la 2G, la 3G, le Wi-Fi, le Wi-Max, etc.) et les smartphones, un nouveau dérivé de l'informatique en cloud est apparu. Ceci est communément appelé « Mobile Cloud Computing (MCC) ».

Le MCC peut être défini comme une composition de la technologie mobile et de l'infrastructure du Cloud Computing, où les différents traitements de l'information sont exécutés dans le Cloud et accessibles via un appareil mobile [85]. Ce type de Cloud devient de plus en plus populaire chez de nombreuses entreprises qui souhaitent faciliter l'accessibilité des données aux employés.

Le Cloud Computing se distingue des autres paradigmes tels que « Grid Computing » dans les divers aspects comme l'approvisionnement de services à la demande, les interfaces centrées sur l'utilisateur, la garantie de qualité de service, le système autonome, etc. [36]. Ces aspects sont dûs à l'utilisation des différentes technologies comme :

Virtualisation : L'une des technologies primordiales qui ont contribué énormément au développement rapide et à la popularité du Cloud Computing. Ce concept fait référence à la technologie qui permet de partager tout type de ressources physiques avec plusieurs utilisateurs sous format de ressources virtuelles, la variation de ces ressources partagées a donné naissance à plusieurs types de virtualisation (Virtualisation de stockage, Virtualisation du matériel, Virtualisation du réseau, ...)

Web Service et SOA (Service-Oriented Architecture) : Ces technologies ont permis d'offrir différents types de services via le web (en utilisant des technologies comme XML, Web Services Description Langage (WSDL), Simple Object Access Protocol (SOAP), ...). L'organisation des services dans un Cloud est gérée sous la forme d'une architecture SOA (Service Oriented Architecture). SOA peut être défini donc comme une entité utilisant plusieurs services pour effectuer une tâche spécifique [54].

Application Programming Interface (API) : les API Permettent le développement des applications et services utilisés pour réserver et utiliser les ressources matérielles et logicielles offertes par le Cloud, ces APIs représentent un portail qui offre un accès direct ou indirect à l'infrastructure du Cloud aux différents utilisateurs.

Le développement rapide du Cloud Computing a créé un environnement dans lequel un ensemble de ressources très important est disponible aux utilisateurs pour

un coût minime. Cela peut être visualisé de deux manière différentes. D'une part, cet environnement représente un espace dans lequel différents types d'utilisateurs peuvent utiliser des ressources avec une fraction du coût d'une infrastructure réel. D'une autre part, la disponibilité de ces ressources a un grand nombre d'utilisateurs augmentent le risque de cyber-attaques à cause de plusieurs facteurs.

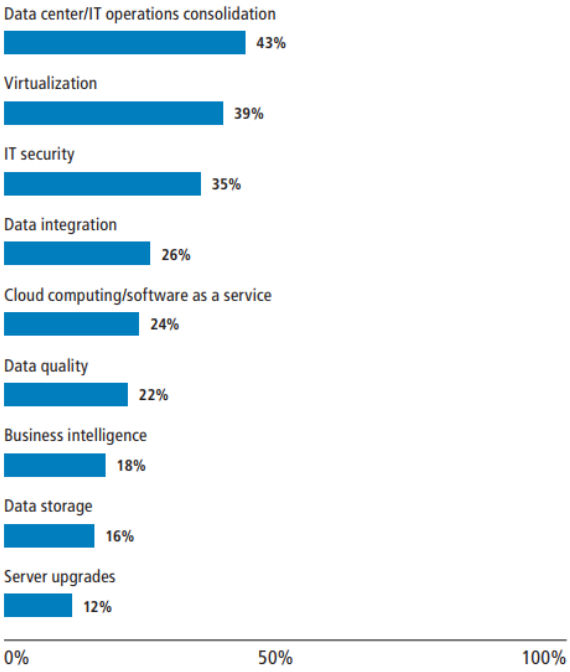


FIGURE I.4 – : Sources de méfiances des organisations envers le Cloud publique [49]

I.2.1 Problèmes de la sécurité dans les modèles de services

Le Cloud est né de la combinaison de plusieurs technologies d'informations, ceci signifie qu'il hérite non seulement du potentiel mais aussi des risques que ces technologies soulèvent. La nature du cloud a créé un ensemble de problèmes de sécurité que les fournisseurs de services doivent prendre en considération. Ces problèmes de sécurité créent des barrières qui découragent les différentes entreprises qui visent à migrer vers une solution Cloud (figures I.4-I.5).

Les modèles de services dans le Cloud peuvent être vus dans un modèle en couches inter-connectées dans lequel le SAAS et le PAAS sont hébergés en dessous de l'IAAS (figure I.6). cette hiérarchie signifie que les vulnérabilités de sécurité dans une couche en bas (IAAS par exemple) vont affecter non seulement les services offerts dans cette couche mais aussi la sécurité des services offerts dans les couches plus hautes.

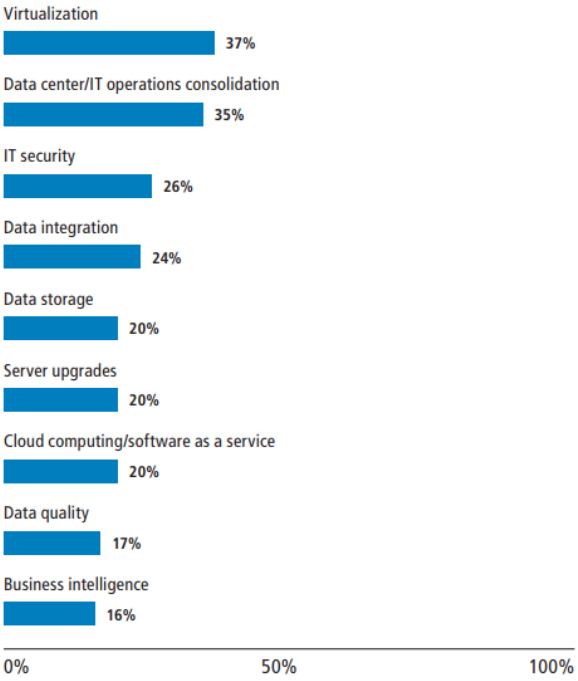


FIGURE I.5 – sources de méfiances des organisations envers le Cloud privés [49]

I.2.1.1 Problèmes de sécurités dans SAAS

: SAAS offre des services sous forme d'applications sur demande (email, ERM, CMR, SCM...) [49] via internet à un grand nombre d'utilisateurs. Les services offerts par un fournisseur sont affectés par les vulnérabilités des applications web avec un niveau de risque plus élevé vu que les données d'un grand nombre d'utilisateurs sont stockées d'une manière groupée dans des infrastructures partagées.

La sécurisation des SAAS est sous la responsabilité du fournisseur de service qui doit s'assurer d'une part de la sécurité applicatif de ces services, et d'autre part de sécuriser l'application contre les différentes failles de sécurité reliées à la plateforme d'accès. En plus, en utilisant les services offerts par un fournisseur, l'utilisateur va compter sur ce dernier pour garantir la confidentialité de ses données [74, 56].

Généralement les données des utilisateurs sont traitées en clair chez le fournisseur de service puis stockées dans ses serveurs de stockage. Le fournisseur de service peut même sous-traiter le stockage de ces données chez un fournisseur tiers ce qui relève d'autres soucis de confidentialité.

I.2.1.2 Problèmes de sécurités dans PAAS

:

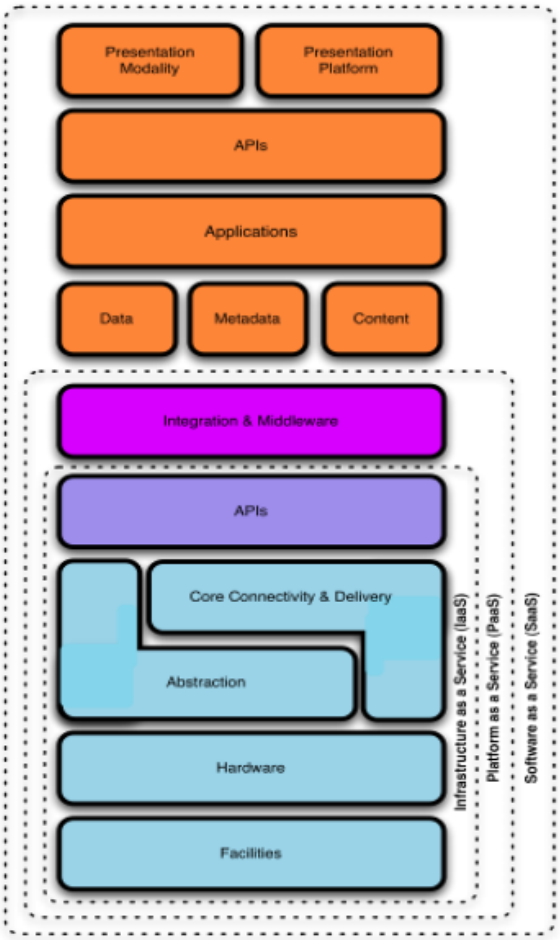


FIGURE I.6 – Modèle de services du Cloud

Dans ce type de modèle le fournisseur de service offre aux clients un environnement qui facilite le développement et le déploiement des services SAAS sans prévoir d'acheter et de maintenir les couches matérielles et logicielles [74].

Le fournisseur de services offre aux utilisateurs du PAAS plus de contrôle sur les différents types d'applications, cela signifie que la responsabilité de sécurité à ce niveau sera partagée entre l'utilisateur et le fournisseur des services cloud (Cloud Services Provider *CSP*).

I.2.1.3 Problèmes de sécurités dans IAAS :

Le fournisseur de services offre des ressources matérielles virtualisées (centre de données, machines virtuelles, ...) sous formes de services aux clients, ces ressources sont partagées dans un environnement multi-tenant entre les différents utilisateurs ce qui relève plusieurs risques de sécurités comme :

a. **Machines virtuelles :** Chaque utilisateur peut définir ses propres contrôles de sécurités qui correspondent à ses besoins, ces machines virtuelles peuvent souffrir des mêmes vulnérabilités des machines réelles (logiciels malveillants, virus ...).

b. **Images de machines virtuelles :** Les machines virtuelles sont stockées sous format d'images dans chez du fournisseur de service. Ces images sont en risque d'être volées ou d'avoir un code malicieux injecté directement. En plus, vu que les machines virtuelles sont dupliquées à partir d'un modèle, si ceci contient des informations liées à l'utilisateur originale elles seront en risque d'être utilisés par un nouvel utilisateur [22].

c. **Sécurité du réseau virtuel :** Le partage d'une infrastructure réseau entre différents utilisateurs dans le même serveur augmente le risque des attaques réseaux, de l'exploitation des vulnérabilités des serveurs (DNS, DHCP et les différents protocoles réseaux) et celles des programmes de virtualisation du réseau. Cela peut d'une part, affecter l'utilisation des différents services fournis par le fournisseur de services (ressources partagées) et d'une autre part, être utilisé comme vecteur d'attaques contre les différentes machines virtuelles stockées dans les serveurs partagés.

d. **Séparations des machines virtuelles :** Les machines virtuelles offertes par le fournisseur de services utilisent les ressources des serveurs réels (CPU, RAM, I/O ...) d'une manière partagée entre elles. Le fournisseur de services doit assurer que les ressources utilisées par les différentes machines sont séparées afin d'assurer la sécurité de données et de ressources utilisées par les différents utilisateurs.

e. **Sécurité des Hyperviseurs :** l'hyperviseur est l'agent de virtualisation qui permet de faire la liaison entre les ressources virtuelles et les ressources réelles et vice versa (Figure I.7). En effet, vu le rôle important joué par l'hyperviseur, une faille de sécurité de cet agent peut affecter la sécurité de toutes les machines virtuelles liées à ce dernier (les informations transmise par l'hyperviseur sont généralement en clair).

I.2.2 Problèmes de la sécurité des données stockées dans le Cloud

Le Cloud offre la possibilité à ses utilisateurs d'externaliser le stockage de leurs données vers un fournisseur de service avec un coût minimisé. Malgré ces avantages, les utilisateurs hésitent d'adopter ce type de services à cause des préoccupations des sécurités des données. Ces préoccupations sont dues principalement a la

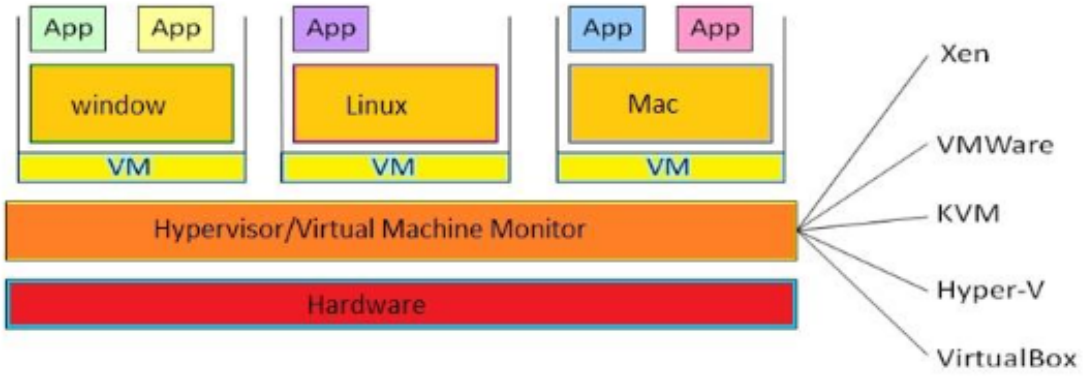


FIGURE I.7 – Hyperviseur

perte de contrôle physique et sont accentuées par les différentes révélations médiatisées. En effet, le service de stockage devient une cible de grande valeur vu que les données de plusieurs utilisateurs de services sont stockées dans le même environnement Cloud[88].

Les services de stockage de données dans le Cloud posent de nombreux défis de conception du à la perte de contrôle physique. Ces défis ont une influence significative sur la sécurité des données et les performances des systèmes. Autrement dit, les données externalisées dans un serveur Cloud sont souvent soumises à un grand nombre de vecteurs d'attaques.

La confidentialité des données, dans l'environnement du Cloud, devient plus difficile et conflictuelle, cela est dû en grande partie au fait que les utilisateurs externalisent leurs données sur des serveurs distants, qui sont contrôlés et gérés par des fournisseurs de services de Cloud (CSP) non toujours fiables. Cela signifie que quand le fournisseur de service devient malicieux, il peut opter activement pour récupérer le maximum d'informations possibles pour maximiser ses gains (par exemple le CSP peux essayer de vendre des informations confidentielles de ses utilisateurs à des sociétés intéressées), ou même supprimer les données en stockage passif (utilisées ou consultées rarement par les utilisateurs) pour libérer de l'espace afin de maximiser les revenus de ses services. En effet, même quand le fournisseur de services est fiable, il peut en cas d'incident qui cause un endommagement ou perte totale ou partielle des données ne pas notifier les utilisateurs impliqués.

Le problème de la sécurité des données peut aussi être causé par la nature de l'architecture du Cloud Computing ; les données sont généralement stockées dans des serveurs de stockages ou centre de données partagées ; cela signifie qu'une vulnérabilité de sécurité dans ce centre de données (que ce soit au niveau de virtualisation,

application ou autre) pourrait affecter la confidentialité et l'intégrité de toutes les données stockées sur ce serveur. Tout cela signifie que l'utilisateur de ce type de services doit prendre suffisamment de précaution afin d'assurer la confidentialité et l'intégrité de ses données.

Le client peut faire face au problème de confidentialité en chiffrant les données avant d'externaliser, ceci va lui permettre de garantir la confidentialité vu que la clé de chiffrement est gardée hors de porté du fournisseur de service [74, 56]. Cette solution peut résoudre potentiellement le problème de confidentialité des données mais vu que les données en stockage sont souvent partagées ou utilisées par plusieurs utilisateurs, l'utilisation de chiffrement peut poser des problèmes de gestion et de stockage des clés de chiffrement pour le client de ce service.

Le problème de gestion des clés deviennent de plus en plus clair lors du partage des données chiffrées. Le partage des données dans un environnement cloud nécessite soit le partage des clés ou le chiffrement des données avec les clés publiques des récepteurs de ces données. En plus, le partage des données exige d'avoir une politique d'accès qui définit des privilèges d'accès granulaire pour chaque utilisateur et permettant une révocation des droits d'accès simple et efficace.

L'utilisation d'une approche cryptographique traditionnelle ne peut pas satisfaire les exigences du Cloud, vu que la complexité de gestion des clés et de la révocation des privilèges augmente d'une façon exponentielle avec le nombre des utilisateurs avec lesquels les données sont partagées. L'utilisation d'une approche cryptographique traditionnelles n'est pas valable dans une environnement Cloud.

Le stockage des données dans les serveurs distribués relève aussi les problèmes de vérifications d'intégrité, vu la non fiabilité des fournisseurs de services Cloud. Même dans les cas où les fournisseurs de services implémentent des politiques de tolérance aux pannes, les clients restent incapables de vérifier l'intégrité de leurs données vu le manque de contrôle physique. En plus, le volume des données distribuées devient de plus en plus énorme, ce qui signifie que la vérification de l'intégrité des données doit être faite d'une manière qui peut garantir l'intégrité des données sans avoir recours à la récupération totale des fichiers stockés. Cette vérification peut être faite en utilisant des algorithmes de hachage traditionnels, mais l'utilisation de ce type d'algorithme va nécessiter la prévision de plusieurs versions de hachage pour chaque fichier stocké ; après chaque vérification le haché utilisé devient inutilisable vu que le fournisseur de service peut garder le haché pour passer les prochaines vérifications sans garder les données stockées.

Conclusion

L'externalisation des données chez un CSP relève des exigences de confidentialité et d'intégrités que les méthodes traditionnelles sont incapable d'y répondre ,ceci nécessite des approches adaptés à l'environnement distribué du Cloud. Dans le chapitre suivant nous allons présenter les bases cryptographiques, puis nous allons discuter quelques approches proposées dans la littérature pour répondre au exigences du Cloud.

Chapitre II

État de l'art

Sommaire

Pendant plusieurs années, la cryptographie s'utilisait exclusivement pour sécuriser les informations dans les domaines militaires, diplomatiques et gouvernementaux. L'utilisation des méthodes cryptographiques permettaient d'offrir des mesures de sécurité telles que l'authentification, la confidentialité et l'intégrité des données. Avec le développement des systèmes d'informations et l'évolution des dispositifs et des méthodes d'échange d'informations, l'utilisation de la cryptographie s'élargie vers la sécurisation des informations dans le domaine publique, cela a poussé les chercheurs à proposer de nouveaux systèmes cryptographiques.

Dans cette section, nous allons présenter les concepts généraux de la cryptographie en décrivant, la cryptographie symétrique et à clé publique puis nous allons décrire les types de chiffrement basés sur les attributs et la vérification probabiliste d'intégrité qui représentent les bases utilisées pour la conception des méthodes discutées tout au long de ce rapport.

II.1 Confidentialité

La confidentialité des données concerne la protection des informations contre l'accès par des personnes non autorisées. En d'autres termes, seules les personnes autorisées à le faire peuvent accéder aux données sensibles. Cela est possible en utilisant une méthode de chiffrement et de déchiffrement qui permet à « Alice » d'envoyer un message à « Bob » de manière à ce qu'un adversaire « Eve » ne reçoive aucune information significative sur le contenu du message. Il est généralement considéré dans l'un des deux modes :

— Dans le mode symétrique (clé privée), le chiffrement et le déchiffrement sont effectués sous une clé partagée par l'expéditeur et le destinataire.

— Dans le mode asymétrique (clé publique), l'expéditeur a certaines informations publiques et le destinataire détient certaines informations secrètes correspondantes.

II.1.1 Chiffrement Symétrique

Le chiffrement symétrique est un type de chiffrement dans lequel une seule clé (une clé secrète) est utilisée pour chiffrer et déchiffrer les données. Les entités qui utilisent le chiffrement symétrique pour sécuriser leur communication doivent échanger, préalablement, la clé afin qu'elle puisse être utilisée dans le processus de déchiffrement. Un algorithme de chiffrement symétrique peut être défini comme suit :

Soient C l'espace de message en texte chiffré, M l'espace de texte en clair et K la clé. Nous désignons l'algorithme de chiffrement par E et l'algorithme de déchiffrement par D :

— **L'algorithme de chiffrement** $E : M \times K \to C$, prend comme entrée le message clair m et la clé de chiffrement k et retourne comme résultat le message chiffré c.

— **L'algorithme de déchiffrement** $D : C \times K \to M$, prend comme entrée le message chiffré c et la clé de chiffrement k et retourne comme résultat le message clair m.

On peut dire qu'un algorithme de chiffrement symétrique est correct si et seulement si $\forall m \in M, k \in K, D(E(m,k),k) = m$.

Généralement les algorithmes de chiffrement symétrique sont classifiés sous deux catégories : algorithmes de flux et algorithmes de bloc.

Chiffrement de Flux

Un chiffrement de flux est un chiffrement à clé symétrique dans lequel le texte est combiné à une clé de chiffrement pseudo-aléatoire. Dans un chiffrement de flux,

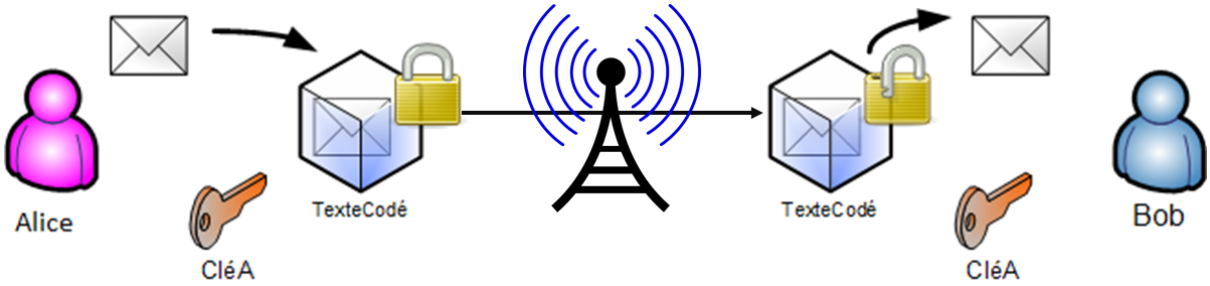


FIGURE II.1 – Chiffrement Symétrique.

chaque caractère en texte brut est chiffré un à un avec le caractère correspondant du flux de clés afin de générer le texte chiffré. En pratique, un chiffre est typiquement un bit et l'opération de combinaison est un ou-exclusif (XOR).

Chiffrement de César L'algorithme de César est l'un des plus anciens et des plus simples méthodes de chiffrement [73], cette méthode a été utilisée par Jules César, qui l'a apparemment utilisée dans ses correspondances secrètes avec ses généraux. Il s'agit d'un type de chiffrement par substitution dans lequel chaque lettre du texte en clair est « décalée » d'un certain nombre de places dans l'alphabet.

Cette méthode utilise la notion de permutation afin de chiffrer les données, ce qui signifie que la clé utilisée est le pas de permutation (par exemple avec un pas de 3 et une permutation à gauche, D devient A, E devient B ainsi de suite).Actuellement, cette méthode ne fournit aucun niveau de sécurité vu sa faiblesse aux attaques à force brute.

Carré de Polybe Proposé par Polybe [113], cette méthode consiste à remplacer chaque lettre par la paire de nombres correspondante dans un tableau de correspondance créé à partir de l'alphabet (par exemple les lettres de l'alphabet latin (26) sont disposées dans un carré de taille 5X5) [16]. Le chiffre de substitution de Polybe a trouvé une grande acceptation chez les cryptographes qui l'ont utilisé comme base pour de nombreux chiffrements [78].

Vu que cette méthode utilise un tableau de correspondance créé à partir des lettres d'alphabet comme base pour chiffrer les messages, cela signifie que cette méthode est susceptible aux attaques par analyse de fréquences [99].

Chiffrement de Vernam Un code à flux symétrique dans lequel le texte en clair est combiné à un flux de données aléatoire ou pseudo-aléatoire (le "flux de clés") de même longueur, afin de générer le texte crypté, à l'aide de la fonction booléenne "exclusif ou" (XOR) [63].

L'algorithme de Vernam est appelé chiffrement à masque jetable, cela signifie que pour chaque opération, la clé de chiffrement doit être renouvelée. Cela pose un grand problème dans le cas de partage d'un grand nombre de messages avec un grand nombre de récepteurs. Ce type d'algorithme pose un grand fardeau de stockage et de gestion vu que la clé (qui est de même taille que le message) doit être stockée par l'expéditeur et le destinataire.

Rivest Cipher 4 (RC4) Proposé par Rivest en 1987, RC4 génère un flux pseudo-aléatoire de bits (un flux de clés) qui est combiné avec le message afin de le chiffrer ou déchiffrer. L'algorithme utilise une clé de longueur variable de 1 à 256 octets pour initialiser une table d'état de 256 octets. Cette table d'états est ensuite utilisée pour la génération ultérieure d'octets pseudo-aléatoires, puis pour générer un flux pseudo-aléatoire qui est combiné avec le texte claire en utilisant une opération XOR afin de générer le texte chiffré. Chaque élément de la table d'état est échangé au moins une fois.

Grâce à sa performance, cet algorithme a été largement utilisé dans les application commerciales et les protocoles de sécurisation d'échanges TLS/SSL. Cet algorithme est considéré comme non sécurisé et son utilisation dans TLS a été interdite par L'Internet Engineering Task Force (IETF) depuis 2015 [83].

Chiffrement par Blocs

Dans ce type de chiffrement, l'algorithme chiffre des blocs de données avec des clés de tailles préfixées. Actuellement, il existe un grand nombre de méthodes de chiffrement par blocs, certaines parmi elles sont publiquement connues, comme :

— Data Encryption Standard (Data Encryption Standard (DES))[102] : Le premier algorithme de chiffrement certifié par NSA, DES exécute le chiffrement de blocs 16 fois sur un bloc de taille 64 bits utilisant une clé de taille 56 bits. Cet algorithme est considéré, depuis 1998 [1], non sécurisé vu qu'avec la puissance des matériels informatiques, la totalité de l'espace de la clé utilisée par DES peut être fouillé dans un temps acceptable (recherche exhaustive).

— Triple Data Encryption Standard (Triple Data Encryption Standard (3DES)) : Une amélioration de DES, dans cette méthode le message est chiffré 3 fois par DES ; $c = 3DES(m) = DES_{k_1}(DES_{k_2}(DES_{k_3}(m)))$. Cet algorithme a une longueur de

1. DES a été cracké "DES cracker (Deep Crack)" dans 56 hours [37]

clé effective de 112 bits qui se trouve en dehors de la capacité actuelle des attaques à force brute. Mais il existe de nouvelles normes de chiffrement plus efficaces que DES recommandées pour l'utilisation.

— Advanced Encryption Standard (Advanced Encryption Standard (AES)) [88] : Considéré comme l'algorithme de chiffrement le plus populaire et le plus utilisé actuellement, il a été proposé comme remplacement du DES à cause de la vulnérabilité de ce dernier aux attaques à force brute. Malgré que triple DES peut être considéré comme un remplacement robuste de DES, son temps d'exécution le rend moins populaire que AES (le temps d'exécution de AES est 3 fois plus rapide que celui de 3DES). Cet algorithme consiste en une série d'opérations de substitutions et de permutations liées.

Le chiffrement Symétrique permet d'effectuer un échange sécurisé d'informations entre deux entités en utilisant une clé de chiffrement/déchiffrement, ce qui est efficace dans un ensemble d'entités limité, mais devient très problématique quand l'interaction est faite entre un grand nombre d'entités ou si les entités sont incapables de faire un partage sécurisé de clé avant de commencer l'échange de données. La nature de ce type de chiffrement impose un échange préalable de clés de chiffrement et une gestion d'un grand nombre de clés. Pour résoudre ce type de problème, une nouvelle forme cryptographique (chiffrement asymétrique) a été proposée dans les années 1970 [91].

Protocole d'échange de clé de Diffie Hellman

En 1976 Whitfield Diffie et Martin Hellman [29] ont proposé l'algorithme de Diffie hellman (DH) d'échange de clés sur un canal non sécurisé. cette algorithme joue un rôle très actif jusqu'à nos jours dans les protocoles des échanges internet. L'algorithme de Diffie Hellman est utilisé pour sécuriser en temps réel les échanges entre deux entités Alice et Bob sur un réseau non sécurisé. Les valeurs publiques échangées entre les deux entités sont un nombre premier p et un générateur non nul $g \in Z_p^*$.

Dans cet algorithme, Alice et Bob commencent par choisir deux clés secrètes $s_A, s_B \in Z_p^*$ et calculer leur clés secrètes respectives : $Pub_A = g^{s_A} mod p$, $Pub_B = g^{s_B} mod p$.

Après avoir échangé leurs clés publiques Alice et Bob peuvent utiliser $g^{A \cdot B} mod p$ comme clé de chiffrement vu que ces clés peuvent être facilement trouvées par les deux acteurs en utilisant leurs clés privées :
$g^{A \cdot B} mod p = Pub_B^{s_A} mod p = Pub_A^{s_B} mod p$.

Le problème principal est que cet algorithme est susceptible aux attaques d'homme du milieu ; une personne (Eve) peut se comporter comme Bob à Alice et comme Alice

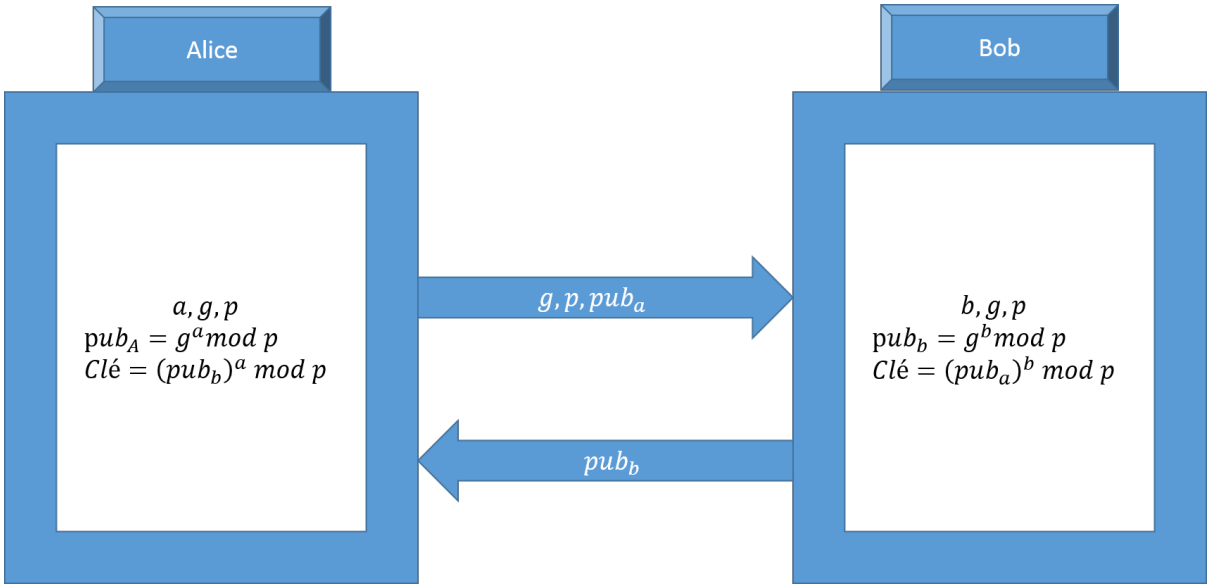


FIGURE II.2 – Échange de clés Diffie-Hellman.

à Bob en remplaçant la clé privée lors de l'échange des clés. Ce problème peut être évité quand Alice et Bob signe leurs propre clés publiques avant l'échange.

II.1.2 Chiffrement Asymétrique

La cryptographie asymétrique ou à clé publique considère deux clés de chiffrement, dites « clés asymétriques ». Ces deux clés sont générées simultanément et sont complémentaires car le chiffrement avec l'une de ces clés nécessite le déchiffrement avec l'autre clé. Chaque clé joue un rôle spécifique. La clé privée est une clé secrète qui n'est connue que par une seule entité. Il est préférable que la clé publique soit largement diffusée, pour que l'entité concernée puisse être identifiée par un grand nombre d'entités. Bien entendu, la connaissance de la clé publique ne doit en aucun cas permettre de retrouver la clé privé.

Ce type de méthode se base sur un protocole d'échange de clé (ex : protocole de Diffie Hellman [29]) dans lequel les différentes entités peuvent partager un secret sans interaction préalable.

Méthodes de chiffrement Asymétrique

Différemment au chiffrement Symétrique, la cryptographie à clé publique utilise des paires de clés; une publique et une privée, qui est gardée secrète. Ce système permet à quiconque de chiffrer un message à l'aide de la clé publique du destinataire. Ce message ne peut alors être déchiffré qu'avec la clé privée du destinataire.

Algorithme de RSA (Rivest–Shamir–Adleman)

RSA est l'un des premiers systèmes cryptographiques à clé publique, cet algorithme est largement utilisé dans le domaine de la transmission sécurisée de données. Dans ce type d'algorithme, chaque entité (individu ou entreprise) a deux types de clé, une clé publique utilisée pour le chiffrement des données et une clé privée utilisée pour le déchiffrement des données. Cet algorithme consiste en quatre étapes principales :

— Génération des clés : dans la phase de génération des clés, l'algorithme choisit deux nombres premiers p et q très grands et calcule $n = pq$. Ensuite, l'algorithme calcule le plus petit multiple commun entre $p-1$ et $q-1$: $\lambda(n) = ppcm(p-1, q-1)$ [2] et choisit un nombre $1 < e < \lambda(n)$ [55], tel que e et q sont premiers entre eux.

L'algorithme calcule un nombre d tel que d est l'inverse modulaire de e. la clé publique consiste de l'exposant e et le nombre du modulo n, la clé privée consiste en la clé de déchiffrement d.

— Distribution (partage/échange) de clés : les deux entités A et B échangent leurs clés publiques entre eux afin de pouvoir chiffrer et transmettre les données d'une manière secrète entre eux.

— Chiffrement : quand une entité A désire envoyer un message M vers une entité B, A commence par chiffrer le message en utilisant la clé publique de B : $c \equiv M^e \mod n$.

— Déchiffrement : suite à la réception du message chiffré c, B récupère le message claire en utilisant sa clé privée d : $M \equiv c^d \mod n$.

La sécurité de l'algorithme de RSA est généralement liée à la difficulté de résolution du problème de factorisation [101] (la tâche de factoriser n en ses deux facteurs premier p et q) ce qui devient de plus en plus facile avec l'évolution des algorithmes de factorisations et la puissance de calcul. Actuellement, RSA est l'un des algorithmes de chiffrement asymétriques les plus utilisés pour le chiffrement et les signatures numériques, cela signifie qu'avec la longueur croissante de la taille de la clé nécessaire pour une utilisation sécurisée du RSA, le coût de traitement devient de plus en plus important. Cette charge a des conséquences, en particulier pour les sites de commerce électronique qui effectuent un grand nombre de transactions sécurisées.

2. changé de $\Phi(x) = (p-1)(q-1)$ dans RFC2437 en 1998

Décimaux dans la clé	Bits dans la clé	Date de réalisation
100	332	Avril 1991
110	365	Avril 1992
120	398	juin 1993
129	428	Avril 1994
130	431	Avril 1996
140	465	Février 1999
155	512	août 1999
160	530	Avril 2003
174	576	Décembre 2003
200	663	Mai 2005
193	640	Novembre 2005
232	768	Décembre 2009
232	768	Juillet 2012
240	795	Décembre 2019
250	768	Février 2020

TABLE II.1 – Progrès en factorisation RSA entre 1991 et 2009 [101]

Cryptographie basée sur les courbes elliptiques

La cryptographie à courbe elliptique a été introduite en 1985 [114] par Victor Miller et Neal Koblitz [76, 59], qui ont développé indépendamment l'idée d'utiliser les courbes elliptiques comme base d'un groupe censé offrir plus de sécurité que d'autres groupes avec des clés de taille beaucoup plus petites. Les courbes elliptiques ont rapidement gagné l'intérêt. Au début des années 2000, la NSA a fait des courbes elliptiques son algorithme standard pour le cryptage et la signature.

Les courbes elliptiques (EC) sont des formes cubiques définies sur des champs finis, généralement un champ premier ou un champ binaire noté F_p ou F_{2p}, où p et $2p$ représentent respectivement l'ordre du champ (le nombre d'éléments du corps fini). Dans cette thèse, nous ne considérons que les courbes elliptiques définies sur des corps premiers finis, répondant à l'équation de Weistrass [43].

L'ensemble des points d'une courbe elliptique $E(F_p)$, avec l'opération binaire : $E(F_p) \times E(F_p) \rightarrow E\ (F_p)$ forme un groupe additif abélien $(E(F_p),)$. C'est-à-dire que l'opération binaire du groupe est l'ajout de deux points de la courbe. La courbe elliptique $E(F_p)$ est dite bien définie (lisse) si son discriminant Δ est différent de 0. Ce dernier garantit que la courbe elliptique ne contient pas de points singuliers pour lesquels l'addition ne peut pas être définie. Ce groupe abélien doit remplir quatre propriétés,

à savoir l'associativité, la commutativité, l'existence d'inverse et l'élément d'identité. Ce dernier est un élément rationnel appelé le point à l'infini P^{∞} [43, 54].

L'utilisation des courbes elliptiques sur des périphériques à faible consommation s'est révélée beaucoup plus évolutive vu qu'elle offre la possibilité d'avoir le même niveau de sécurité offert par RSA en utilisant une clé de taille relativement petite. (voir table II.2).

RSA (Taille de clé en bits)	ECC (Taille du modulo en bits)
1024	160-223
2048	224-255
3072	256-383
7680	384-511
15300	512+

TABLE II.2 – Comparaison de taille de clé pour avoir le même niveau de sécurité [6]

Le tableau II.2 montre que l'utilisation des clés Courbes Elliptiques (EC) est plus intéressante que les clés RSA dans la cryptographie à clé publique. En d'autres termes, pour le même niveau de sécurité, la recommandation de taille de clé actuelle dans RSA est de 2 048 bits, le même niveau de sécurité peut être atteint en utilisant une clé EC relativement très petite de 224 bits [54], cet avantage augmente considérablement avec le niveau de sécurité. Par conséquent, l'utilisation des méthodes cryptographiques basées sur les courbes elliptiques devient intéressante vu que la mise en oeuvre de courbes elliptiques nécessite moins de capacités de stockage et de calcul tout en conservant un haut niveau de sécurité.

Infrastructure a clés publique (PKI)

Une infrastructure à clé publique (PKI) est un système pour la création, le stockage et la distribution de certificats numériques qui sont utilisés pour vérifier qu'une clé publique particulière appartient à une certaine entité. un tiers de confiance (autorité de certification) crée des certificats numériques qui mappent les clés publiques aux entités, stocke en toute sécurité ces certificats dans un référentiel central et les révoque si nécessaire.

Ce type d'Infrastructure dépend largement sur la sécurité de l'autorité de certification, ce qui rend l'impact de sa vulnérabilité très critique.

Chiffrement basé sur l'identité

En 1984, Shamir [95] a introduit la cryptographie à base d'Identité (ID) (IBC) avec l'idée originale de fournir des paires de clés publiques et privées qui ne nécessitent aucun certificat ni déploiement de CA. Shamir suppose que chaque entité utilise l'un de ses identifiants comme clé publique, il attribue la fonction de génération de clé privée à une entité spéciale appelée générateur de clé privée (Private Key Generator, Générateur privé de clés (PKG)). C'est-à-dire qu'avant d'accéder au réseau, chaque entité doit contacter le PKG pour obtenir sa clé privée. L'**((IBC)** a été amélioré par l'utilisation de la cryptographie à courbe elliptique **(ECC)**. En conséquence, de nouvelles méthodes de chiffrement et de signature basées sur des identités ont été proposées.

En 2001, Boneh et Franklin [14] ont proposé le premier algorithme de chiffrement basé sur l'ID, reposant sur l'utilisation de fonctions d'appariement bilinéaire pour lier des points de courbe elliptiques à un certain nombre de groupes multiplicatifs.

Le Chiffrement basé sur l'identité [14] est un type de chiffrement asymétrique dans lequel on a plus besoin d'avoir un échange préalable des clés publiques. Ce type de chiffrement consiste en trois acteurs principaux (figure II.3) :

— Générateur de clé (PKG) : cette entité est responsable de la génération des clés privées de tous les acteurs du système. Chaque acteur enregistre son identité chez le *PKG* et reçoit la clé privée générée en se basant sur son identité.

— Expéditeur : l'expéditeur utilise l'identité du destinataire en combinaison avec la clé publique maitre (la clé publique du *PKG*) pour chiffrer les données avant de les envoyer au destinataire.

— Destinataire : après la réception des données chiffrées de la part de l'expéditeur, le destinataire utilise sa clé privée (récupérée de la part du *PKG*) pour déchiffrer les données.

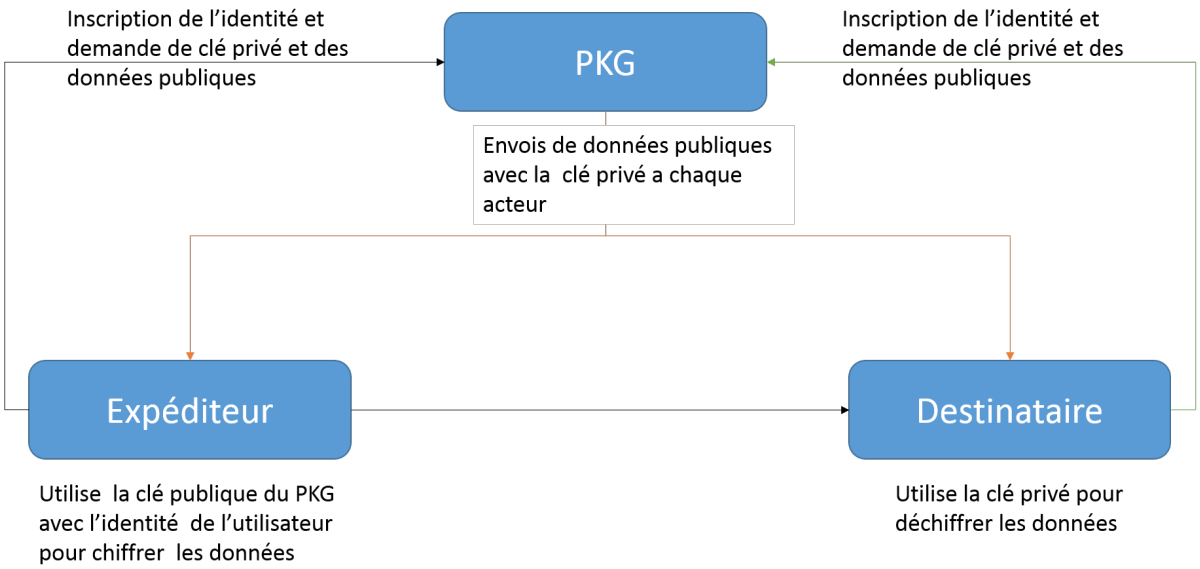


FIGURE II.3 – Chiffrement basé sur l'identité.

II.1.3 Chiffrement basé sur les attributs

Chiffrement basé sur les attributs (Attribute based encryptions (Chiffrement basé sur les attributs (ABE))) a été proposé par Sehai et Waters [90] en 2006. Cette méthode cryptographique permet l'externalisation des données chez un fournisseur de service tout en maintenant la confidentialité des données et une gestion de contrôle d'accès flexible [65]. Dans cette approche, le propriétaire des données utilise un ensemble d'attributs pour définir les privilèges des différentes utilisations et la politique d'accès aux données partagées pour les données chiffrées.

Suite à la proposition de chiffrement basé sur les attributs [90], plusieurs variantes de cette méthode ont été proposées (figure II.4) pour répondre aux besoins de différents modèles de systèmes [61]. Parmi ces variantes $CP-ABE$ et $KP-ABE$ se focalisent sur le partage de secret dans un système mono-autoritaire (une seule autorité est responsable de génération des clés, chiffrement et partage des données) :

— Key-policy attribute based encryption[120] (**(KP-ABE)**) :Les privilèges de chaque utilisateur sont décrits par une structure d'accès intégré dans sa clé privée. Cette politique d'accès est composée d'un ensemble d'attributs liés par des opérateurs logiques « ET » et « OU ». Chaque message chiffré est associé à un ensemble d'attributs, seul les utilisateurs qui trouvent, parmi les attributs associé au texte chiffre, suffisamment d'attributs pour satisfaire leurs structures d'accès peuvent récupérer la clé de déchiffrement.

— Ciphertext-policy attribute based encryption [10] **(CP-ABE)** : Déferrement a KP-ABE, la politique d'accès dans CP-ABE est associé au texte chiffré. Chaque texte

chiffré est associé à une structure d'accès décrivant les privilèges nécessaires pour récupérer la clé de déchiffrement. Un utilisateur peut décrypter si et seulement s'il trouve, parmi ses attributs, un sous-ensemble capable de satisfaire la structure d'accès du texte chiffré.

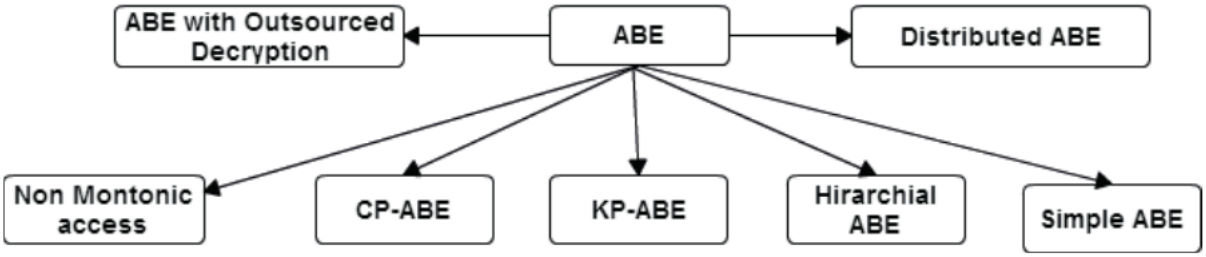


FIGURE II.4 – Variantes de chiffrement basé sur les attributs [61].

Structure d'accès

BIEMEL et al [7]. définissent les méthodes linaires de partages de secret comme des méthodes de partage qui ont une fonction linéaire de reconstruction ; le secret peut être récupéré en utilisant une combinaison linéaire des différents éléments de l'ensemble sur lequel la fonction est définie. Cette fonction linéaire est liée à une structure de contrôle d'accès qui sert à définir un niveau d'abstraction à la politique d'accès.

Définition : Soit $U = U_1, U_2, \ldots, U_n$ un ensemble d'utilisateurs, $A \in 2^{U_1,U_2,\ldots,U_n}$ une collection monotone. une structure d'accès est une collection A de sous-ensembles $U_1, U_2, \ldots, U_n$ qui décrivent les ensembles autorisés. Le rôle d'un utilisateur est défini par l'ensemble des d'attributs qui lui ont été attribués[71].

Une structure de contrôle d'accès peut être définie comme un arbre avec les attributs comme feuilles et des nœuds $(AND, OR...)$ comme portes seuillées (figure II.5).

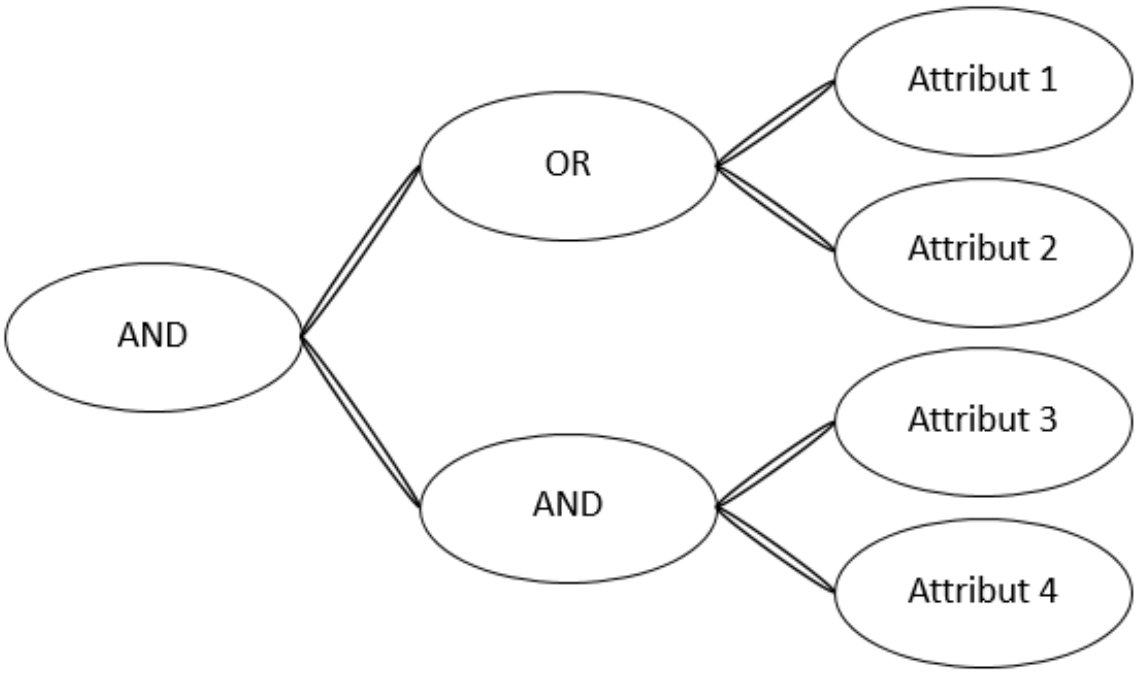


FIGURE II.5 – Structure d'accès binaire.

Types de Structure d'accès

Structure d'accès à forme minimale (minimal form access structures) : Une structure d'accès monotone A peut être décrite par un ensemble A^{-1} composé d'un ensemble de sous-ensembles minimal dont chaque sous ensemble S satisfait la structure d'accès A.
par exemple la structure d'accès décrite dans la figure II.5 peut être représentée comme $\{\{attribut1, attribut3, attribut4\}, \{attribut2, attribut3, attribut4\}\}$.

Formules booléennes monotones (Montone Boolean Formulas) : Une formule booléenne monotone est généralement une manière plus fine pour décrire une structure d'accès minimale, cela signifie que chaque structure d'accès minimale $A = A_1, A_2, \cdots, A_n$ peut être représentée par une formule booléenne simplifiée d'une taille plus fine. Par exemple $\{A, B\}, \{A, C\}, \{A, B, C\}$ peut être présentée sous la forme $(A \wedge B) \vee (A \vee C)$ qui peut être simplifiée comme $A \wedge (B \vee C)$

Arbre d'accès monotone (Monotone Access Trees) : Dans une structure d'accès monotone chaque nœud représente une porte à seuil et chaque feuille représente un attribut. On peut distinguer deux types principaux d'arbre d'accès monotone :

— Structure d'accès à seuil($t-n$ threshold-gate Access structure) : La structure d'accès à seuil est la forme générale d'une structure d'accès monotone. Dans ce type de structure d'accès chaque nœud est représenté par un seuil d'activation $t-n$ (dont t est le seuil et n est le nombre de feuille liées à ce nœud) pour lequel l'utilisateur doit avoir au moins t attributs (parmi les n attributs liés au nœud à question) pour l'activer et calculer la valeur correspondante.

— Structure d'accès booléenne ($(AND-OR)$-gate access structure) : Structure d'accès booléenne est une forme spéciale de la structure d'accès à seuil dans lequel $n=2$ et $t=1$ pour le cas de OR ou $t=2$ pour le cas de AND (figure II.5).

Une structure d'accès à seuil peut être transformée en une structure d'accès booléenne en transformant chaque porte $t-n$ en multiples portes $AND-OR$ mais cela va (dans la plupart des cas) augmenter le nombre de feuilles dans la structure d'accès ce qui affectera la performance de la méthode (voir chapitre 3).

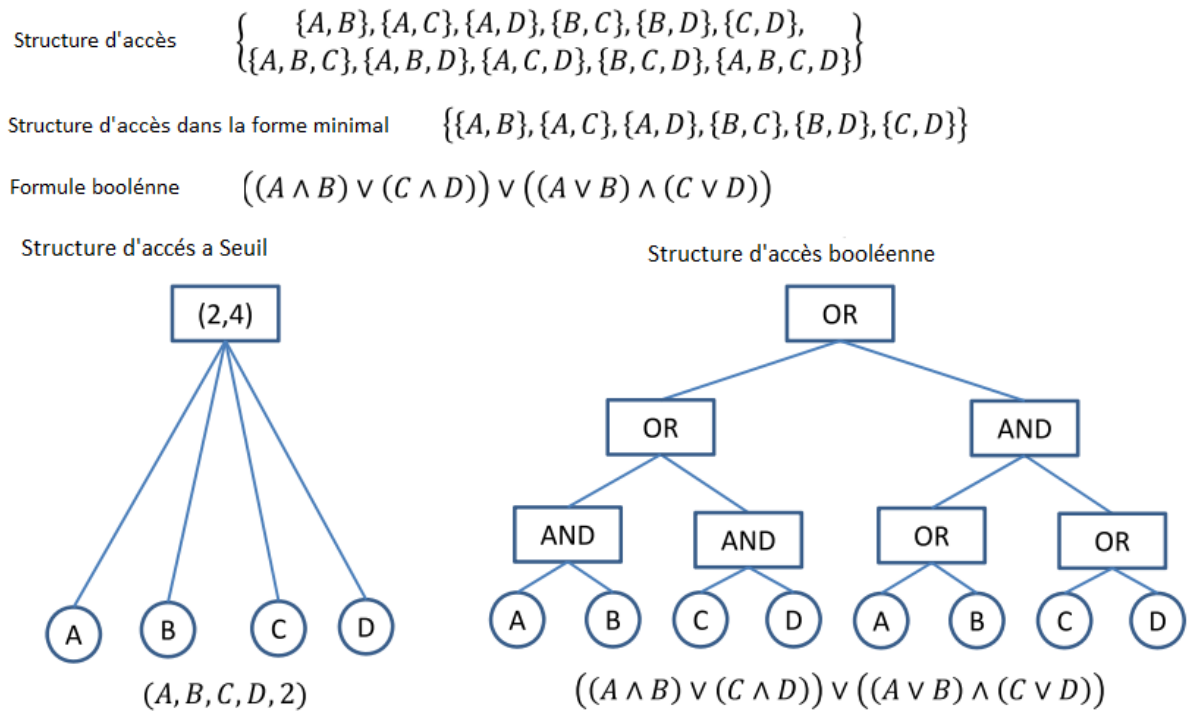


FIGURE II.6 – Exemple de structure d'accès [71].

Key-Policy Attribute based encryption (Key-policy attribute based encryption (KP-ABE) (KP-ABE))

Cet algorithme (décrit dans la figure II.8) utilise les structures d'accès pour définir les droits d'accès de ses utilisateurs. La clé de Chaque utilisateur dans KP-ABE est liée à une structure d'accès qui sert à décrire son rôle dans l'organisation. Chaque texte

chiffré est lié a un ensemble d'attributs. Un utilisateur peut accéder aux données partagées si et seulement s'il trouve suffisamment d'attributs parmi les attributs associés aux fichiers pour satisfaire la structure d'accès associée à sa clé.

Cet algorithme se compose de 4 sous algorithmes principaux [90] :

— Initialisation ($setup(DA) -> (MK, PK)$) : la fonction d'initialisation prend comme paramètre l'univers des attributs DA et retourne une clé privée maîtresse MK et une clé publique PK partagée avec l'ensemble des membres du système.

L'algorithme commence par choisir un groupe Bilinéaire G_1 d'ordre premier p. Soit g le générateur de G_1, $e : G_1 \times G_1 -> G_2$ une Application bilinéaire. L'algorithme génère MK et PK de la manière suivante :

- **Clé maitresse** MK : le système commence par choisir un nombre $\alpha \in Z_p$. Pour chaque attribut $att_i \in DA$ l'algorithme génère un nombre $z_i \in Z_p$ et définit la clé maitresse comme : $MK = \{\alpha, [z_i]_{i \in UA}\}$

- **Clé Public** PK : le système calcule $Y = e(g,g)^{\alpha}$ puis définit le clé publique comme $PK = \{Y, [g^{z_i}]_{i \in UA}\}$

— Génération de clés $Keygen(MK, A)$: cet algorithme prend comme paramètre la clé maitresse MK et une structure d'accès A_u décrivant le rôle de l'utilisateur u. L'algorithme génère la clé privée de la manière suivante :

- L'algorithme procède comme suit. Commencez par choisir un polynôme $Q(i)$ pour chaque nœud x (y compris les feuilles) de la structure d'accès A. Ces polynômes sont choisis de la manière descendante, à partir du nœud racine r (exemple de structure dans la figure II.7 avec $Q(i)$ le polynôme choisi et 1 la valeur associée à la racine).

- Après la construction de la structure d'accès et la définition du polynôme associé à la structure d'accès, l'algorithme génère la clé secrète en calculant $D_i = g^{Q_i(0)/t_i}$ pour chaque attribut att_i la clé secrète est définie comme suit :
$D = \{D_i, i \in A\}$

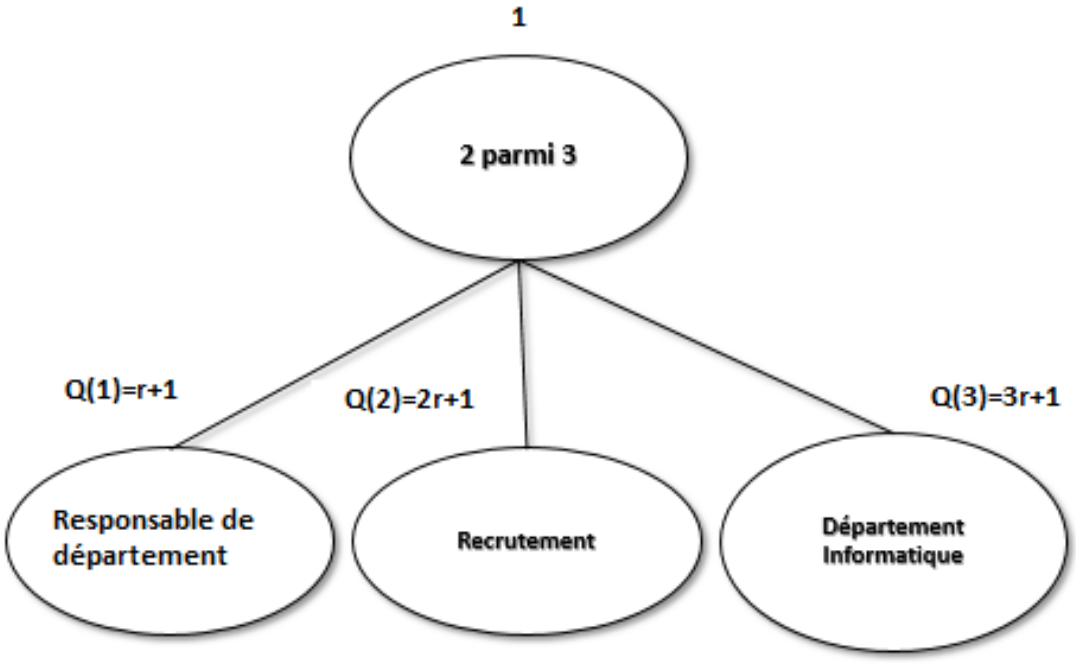


FIGURE II.7 – Exemple d'utilisation de polynôme dans une structure d'accès pour décrire les droits d'un utilisateur.

— Chiffrement $Enc(MK, s, M, A)$: pour chiffrer un message M l'algorithme commence par choisir une clé de chiffrement $s \in Z_p$ puis appelle l'algorithme de chiffrement en passant la clé maitresse MK, la clé secrète s, le message M et l'ensemble d'attributs A. L'algorithme génère le message chiffré E comme suit : $E = \{M_c = MY^s, \{E = T_i^s\}\}$.

— Déchiffrement $Dec(D, E, PK)$: un utilisateur peut récupérer le message M (s'il satisfait les conditions de déchiffrement) en utilisant sa clé privée de manière suivante :

L'utilisateur commence par choisir l'ensemble d'attributs S, parmi les attributs associés au fichier chiffré, satisfaisant sa structure d'accès puis récupère la clé de déchiffrement :

$$\begin{aligned} e(g,g)^s &= \prod_{x \in S} e(D_x, t_x)^{\Delta_x} \\ &= e(g^{Q_x(0)/t_x}, T_i^s)^{\Delta_x} \\ &= e(g^{Q_x(0)}, g^s)^{\Delta_x} \end{aligned} \tag{II.1}$$

L'utilisateur peut par la suite récupérer le message clair en calculant $M = M_c \cdot e(g,g)^s$

Par example, on suppose qu'un utilisateur a comme attributs Responsable de département et Recrutement. En utilisant ces attributs il sera capable de calculer $Q(0) = Q(1)\Delta_1 + Q(2)\Delta_2 = 2(r+1) - 2r - 1 = 1$. Donc en appliquant l'algorithme de déchiffrement ci-dessus il peut retrouver $e(g^{Q_x(0)}, g^s) = e(g^1, g^s) = e(g,g)^s$

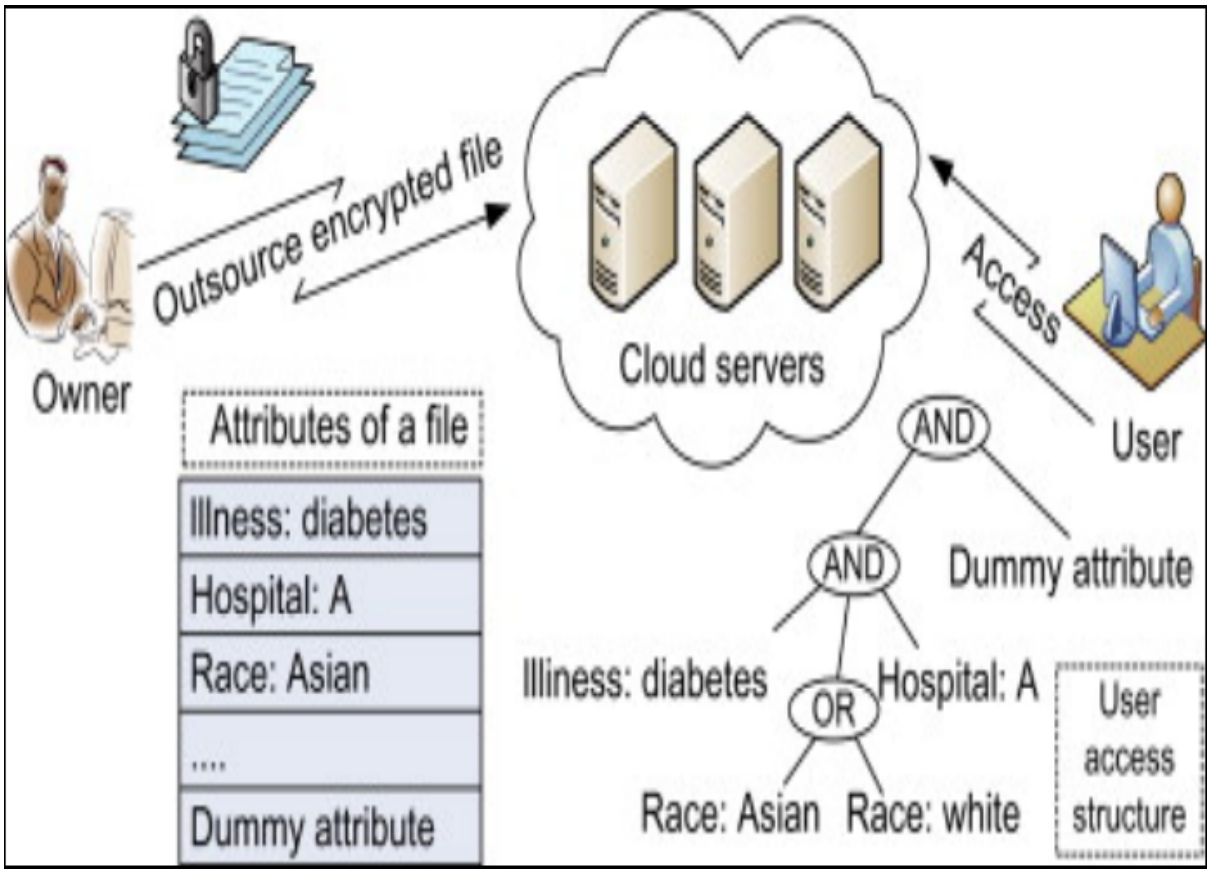


FIGURE II.8 – Exemple d'un système KP-ABE [60].

CP-Attribute based encryption

Dans cet algorithme (décrit dans la figure II.10) la clé de chaque utilisateur est composée d'un ensemble d'attributs décrivant son rôle dans l'organisation, le propriétaire des données chiffre les données en attribuant une structure d'accès qui sert à décrire la politique d'accès liée au fichier. Un utilisateur peut accéder aux données partagées si et seulement s'il trouve suffisamment d'attributs parmi ceux qui décrivent son rôle pour satisfaire la structure d'accès associée au fichier chiffré.

CP-ABE se compose de 5 parties principales [10] :

— initialisation ($setup(DA) -> (MK, PK)$) : la fonction d'initialisation prend comme paramètre l'univers des attributs DA et retourne une clé privée maitresse MK et une clé publique PK partagée avec l'ensemble des membres du système.

L'algorithme commence par choisir un groupe Bilinéaire G_1 d'ordre premier p. Soit g le générateur de G_1, $e : G_1 \times G_1 \rightarrow G_2$ une Application bilinéaire, soit $H : 0,1^* \rightarrow G_1$ une fonction de hachage qui lie un mot binaire à un élément de l'ensemble G_1. L'algorithme génère MK et PK de la manière suivante :

- - Clé maitre MK : le système commence par choisir $\alpha, \beta \in Z_p$, et calcule la clé maitre $MK = \alpha, g^{\beta}$
- - Clé Public PK : le système calcule $Y = e(g,g)^{\alpha}$, $f = g^{1/\beta}$ et publie la clé publique :

$$PK = \{G_1, Y, f, g\}$$

— Génération de clés $Keygen(MK, A)$: cet algorithme prend comme paramètre la clé maitre MK et un ensemble d'attributs E_u décrivant le rôle de l'utilisateur u. L'algorithme génère la clé privée de la manière suivante :

- - le système commence par choisir $r \in Z_p$, puis pour chaque attribut $i \in E_u$ le système choisit $r_i \in Z_p$ et calcule la clé secrète sous la forme :
$SK = \{D = g^{(\alpha+r)/\beta}, \forall i \in E_u : D_i = g^r H(i)^{r_i}, D'_i = g^{r_i}\}$

— Délégation des droits d'accès $delegate(SK, E_{u_y})$: la fonction de délégation permet à un utilisateur x de définir les droits d'accès de l'utilisateur y en un sous ensemble d'attributs $E_{u_y} \in E_{u_x}$. L'utilisateur x peut par la suit calculer la clé déléguée de l'utilisateur y de la manière suivante :

L'utilisateur commence par choisir un élément secret $r' \in Z_p$, puis pour chaque attribut $i \in E_{u_y}$ il choisit un élément $r'_i \in Z_p$ et calcule la clé :
$SK_y = \{\bar{D} = D^{r'}, \forall i \in E_u : \bar{D}_i = Dg^r H(i)^{r'_i}, \bar{D}'_i = Dg^{r_i}\}$

- - après la construction de la structure d'accès et la définition du polynôme associé, l'algorithme génère la clé secrète en calculant $D_i = g^{Q_i(0)/t_i}$ pour chaque attribut att_i. La clé secrète est définie comme suit : $D = \{D_i, i \in A\}$

— Chiffrement $Enc(MK, s, M, A)$: pour chiffrer un message M l'algorithme commence par choisir une clé de chiffrement $s \in Z_p$ puis appelle l'algorithme de chiffrement en passant la clé maitre MK, la clé secrète s, le message M, L'ensemble d'attributs S et la structure d'accès A définissant la politique associée au fichier (Figure II.9). L'algorithme calcule par la suite le message chiffré E comme suit :

Soit Y l'ensemble des feuilles de la structure d'accès A, le message chiffré peut être décrit comme :
$CT = \{A, \bar{C} = MY^s, C = h^s, \forall x \in Y : C_x = g^{Q_x(0)}, C'_x = H(att(x))^{Q_x(0)}\}$.

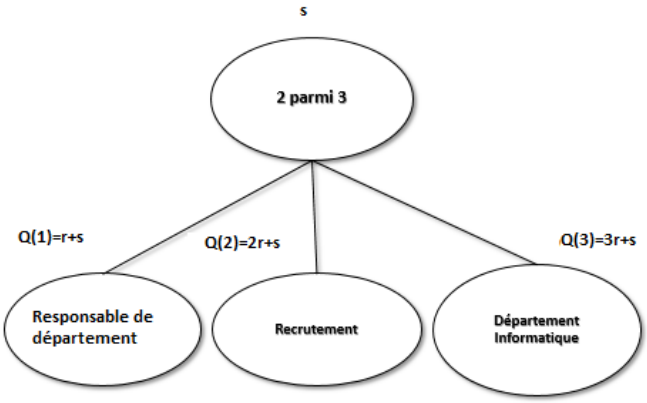


FIGURE II.9 – Exemple d'utilisation de polynôme pour partager un secret dans une structure d'accès.

— Déchiffrement $Dec(CT, SK, PK)$: un utilisateur peut récupérer le message M (si ses attributs peuvent satisfaire la politique d'accès du fichier) en utilisant sa clé privée de la manière suivante :

L'utilisateur commence par choisir l'ensemble des attributs S satisfaisant la structure d'accès du fichier puis récupère le message de la manière suivante : soit $i = att(x)$:

$$\begin{aligned} T &= \prod_{x \in S} \left(\frac{e(D_i, C_x)}{e(D'_i, C'_x)}\right)^{\Delta}_{x} \\ &= e(g,g)^{rs} \\ M &= \bar{C} / e(C, D) / T \end{aligned} \tag{II.2}$$

Prenant le cas décrit par la structure d'accés (figure II.9). soit un utilisateur qui a comme attributs "Responsable de département" et "Recrutement". En utilisant ces attributs il sera capable de calculer $Q(0) = Q(1)\Delta_1 + Q(2)\Delta_2 = 2(r+s) - 2r - s = s$. Donc en appliquant l'algorithme de déchiffrement ci-dessus il peut retrouver $T = e(g,g)^{rs}$

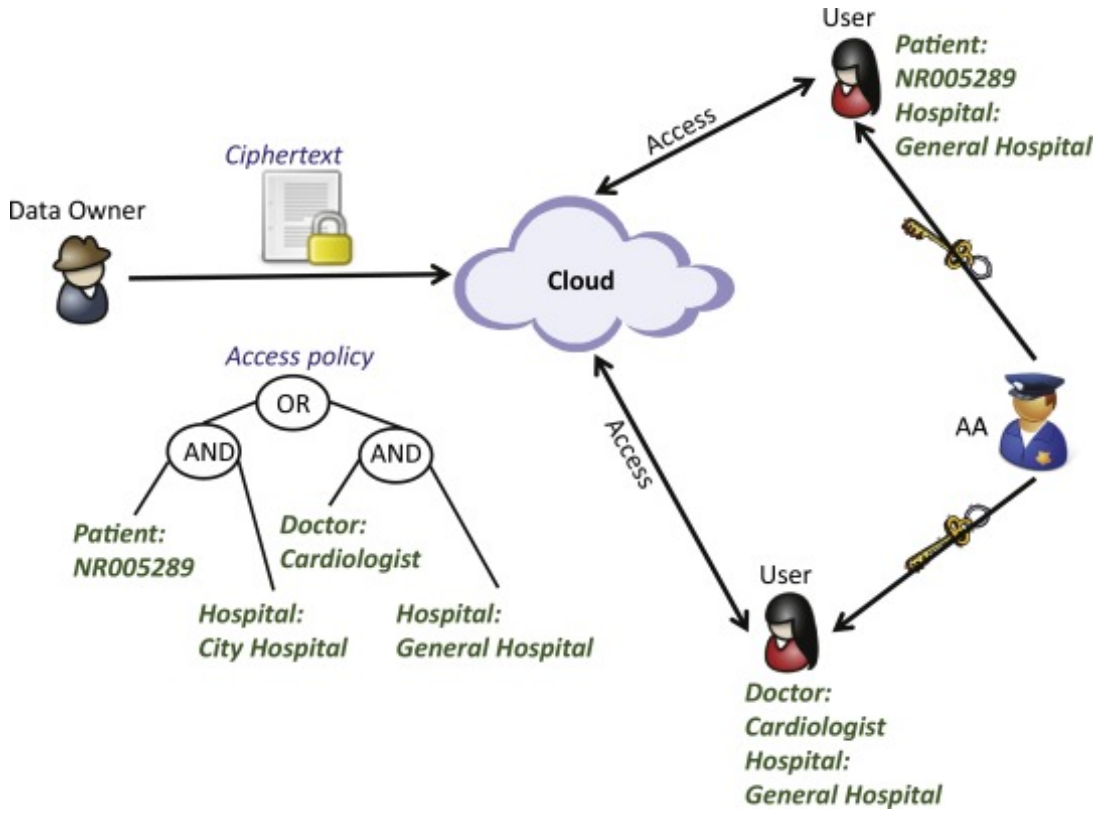


FIGURE II.10 – Exemple d'un système CP-ABE [27].

P2E

X. DONG et al. a proposé une variante de CP-ABE abrégé P2E [30], cette approche est capable de fournir un partage sécurisé des données tout en maintenant la possibilité de gestion des droits des utilisateurs (accord, révocation...).

Dans cette méthode, l'algorithme simplifie tout type de structures d'accès sous forme d'une formule booléenne avant de la transformer en matrice en utilisant la méthode de BEIMEL [7].

P2E se compose de 5 parties principales :

Initialisation

Lors de l'initialisation de l'univers d'attributs, Le propriétaire des données commence par choisir un grand nombre q, deux groupes G_1 et G_2, une application bilinéaire $e : G_1 \times G_1 \rightarrow G_2$ et une fonction de hachage H qui mappe un utilisateur ID_u à un élément de G_1. Ensuite, définit un ensemble d'attributs W qui seront utilisés pour partager des données. Pour chaque attribut $i \in W$, le propriétaire génère deux nombres aléatoires α_i, β_i, dans Z_p^*, de sorte que la clé secrète est :

$$Sk = \{\alpha_i, \beta_i, i \in W\} \tag{II.3}$$

et publie la clé publique :

$$Pk = \{e(g_1, g_1)^{\alpha_i}, g_i^{\beta_i}, i \in W\}. \tag{II.4}$$

Chiffrement

Pour chiffrer un message M_a, le propriétaire commence par définir un ensemble d'attributs I. Sur la base de I, le propriétaire définit une formule booléenne décrivant la politique d'accés du fichier, puis l'utilise pour générer la matrice de partage M et $\rho(x)$ la fonction qui mappe l'attribut x à la ligne correspondant dans la matrice. à l'aide de la méthode de conversion de BEIMEL [7]. Ensuite, Le propriétaire choisit deux vecteurs aléatoires v et ω, tels que la première entrée de v est s et la première entrée de ω est 0, puis calcule $\lambda_x = v \times M_x$ et $\omega_i = \omega \times M_i$. (avec i l'indice la ligne correspondant a l'attributs x). Pour chaque attribut x, le propriétaire choisit un nombre aléatoire $r_x \in Z_p$, puis calcule :

$$\begin{aligned} C_{\rho(x),1} &= e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \\ C_{2,\rho(x)} &= g_1^{r_{\rho(x)}} \\ C_{\rho(x),3} &= g_1^{\beta_{\rho(x)} r_{\rho(x)}} g_1^{\omega_{\rho(x)}} \end{aligned} \tag{II.5}$$

Le message M_a est chiffré en utilisant un algorithme de chiffrement symétrique avec $e(g,g)^s$ comme clé : $C_0 = Enc^{sym}_{e(g,g)^s}(M)$

Enfin, le propriétaire des données envoie le fichier chiffré suivant au CSP $D = \{\forall x, C_{\rho(x),1}, C_{\rho(x),2}, C_{\rho(x),3}; (M, \rho), C_0\}$.

Génération de clés

le propriétaire des données reçoit l'identité de l'utilisateur ID_u de la part du PKG et choisit un ensemble d'attributs I_u décrivant ses droits d'accès. La clé secrète de l'utilisateur est calculée par-suite comme suit :

$$Sk_u = \{g_1^{\alpha_i} H(ID_u)^{\beta_i}, i \in I_u\}. \tag{II.6}$$

La clé générée Sk_u est chiffrée en utilisant la clé publique de l'utilisateur avant de la livrer ,soit directement ou via le CSP , de cette façon, seul l'utilisateur peut récupérer sa clé secrète en utilisant sa clé privé.

Déchiffrement

L'utilisateur récupère le message chiffré D depuis le serveur puis il commence le processus de déchiffrement. l'utilisateur choisi l'ensemble d'attributs qui satisfait la politique d'accès a partir de ses attributs I_u, puis calcule :

$$\begin{aligned}
&\prod_x \left(\frac{C_{\rho(x),1} \cdot e(H(ID), C_{\rho(x),3})}{e(sk_{\rho(x),u}, C_{\rho(x),2})}\right) \\
&= \prod_x \left(\frac{e(g_1,g_1)^{\lambda_{\rho(x)}} \cdot e(g_1,g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1^{\beta_{\rho(x)} r_{\rho(x)}} g_1^{\omega_{\rho(x)}})}{e(g_1^{\alpha_{\rho(x)}} H(ID_u)^{\beta_{\rho(x)}}, g_1^{r_{\rho(x)}})}\right) \\
&= \prod_x \left(\frac{e(g_1,g_1)^{\lambda_{\rho(x)}} \cdot e(g_1,g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1)^{\beta_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1)^{\omega_{\rho(x)}}}{e(g_1,g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID_u), g_1)^{\beta_{\rho(x)} r_{\rho(x)}}}\right) \\
&= (e(g_1,g_1)^{\Sigma\lambda_{\rho(x)}} \cdot e(H(ID), g_1)^{\Sigma\omega_{\rho(x)}} \\
&= e(g_1,g_1)^s \cdot e(H(ID), g_1)^0) \\
&= e(g_1,g_1)^s
\end{aligned} \tag{II.7}$$

l'utilisateur récupéré par-suite la message claire en utilisant la clé $e(g_1,g_1)^s$ avec l'algorithme de chiffrement symétrique :
$Ma = Dec^{sym}_{e(g_1,g_1)^s}(C_0)$.

Révocation

Révocation des droits à un fichier : Lors de la révocation des droits, le propriétaire reconstruit la politique d'accès avec une nouvelle clé de chiffrement puis chiffre les données et met à jour une partie des méta-données correspondante chez le CSP $(C_{0,new}, C_{\rho,1})$. Ensuite, le propriétaire fait confiance au CSP pour ne pas divulguer les nouvelles valeurs aux utilisateurs révoques.

Révocation des attributs : Cette opération utilise la même logique suivit par le processus de révocation des droits à un fichier ; le propriétaire réadapte les fichiers chiffrés et fait confiance au fournisseur de service pour contrôler l'accès. En plus, il accorde des nouveaux attributs aux utilisateurs non révoquées afin de les utiliser dans le chiffrement des nouveaux fichiers.

Malgré l'efficacité de cette méthode, l'utilisation des formules booléennes comme structures d'accès impacte le nombre d'attributs nécessaire pour décrire des structures d'accès limites. En plus, le processus de révocation repose sur le fournisseur de service pour contrôler l'accès aux fichiers révoqués ce qui ne prend pas en considération la collusion entre les utilisateurs et le fournisseur de services.

II.2 Intégrité

L'intégrité a pour but d'assurer l'exactitude et la cohérence des données tout au long de leurs cycles de vie [17]. Elle constitue un aspect primordial de la conception, de la mise en œuvre et de l'utilisation de tout système de stockage, traitement et transmission des données. Le but général des techniques de vérification d'intégrité est de s'assurer que les données sont enregistrées correctement sans avoir subies aucune modification au cours de la durée de stockage. Les données externalisées chez un fournisseur de services subissent plusieurs risques d'intégrités comme :

— les données peuvent être endommagés chez le fournisseur de services à cause des pannes matérielles.

— les données peuvent être modifiées suite à des attaques internes ou à un mal fonctionnement du système interne (hyperviseur, antivirus . . .)

— le fournisseur de service peut opter pour supprimer les données inactives afin de maximiser ses gains (il peut relouer un espace utilisé par un utilisateur.)

La vérification d'intégrité des données stockées localement peut être effectuée en pré-calculant des signatures/hachées des fichiers en utilisant des algorithmes tels que (MD5, MD4, SHA-0 , SHA-1, SHA-2...) [57, 11, 58, 8]. Le fichier est dit intègre tant que la signature pré-calculée est égale à celle calculée au moment de vérification. Ce type de méthode n'est plus valable dans un environnement distribué vu la nécessité de récupération des données pour chaque vérification [12], ce qui a donné naissance à de nouvelles méthodes de vérification distribuée.

II.2.1 Vérifications Traditionnelles

Vérification par signature : Signature calculée par le fournisseur de service

Cette vérification consiste à préparer un ensemble de signatures pour les données avant de les externaliser vers un fournisseur Cloud (Figure II.11). En effet, ce type d'approche imposera un grand fardeau de stockage chez le propriétaire des données.

FIGURE II.11 – Vérification d'intégrité en utilisant des signatures précalculées.

Le problème dans ce type d'approche c'est que les signatures sont utilisables une seule fois, ceci signifie que le propriétaire des données doit prévoir un nombre important de signatures avant l'externalisation des données.

Vérification par signature : Signature recalculée par l'utilisateur

Le propriétaire des données dans cette méthode externalise ses données vers le fournisseur de service et les récupère à chaque instant de vérification d'intégrité (Figure II.12).

FIGURE II.12 – Vérification d'intégrité en recalculent la signature en local.

Malgré que cette approche permet la vérification répétitive des données sans prévoir un grand nombre de signatures ou d'hachées, le propriétaire est amené à récupérer ses données pour chaque vérification ce qui imposera un grand fardeau non seulement de transmission, mais aussi de calcul.

Les problèmes illustrés dans les approches traditionnelles (exemple approche 1 et 2) deviennent de plus en plus inutilisable à cause de la croissance exponentielle de la taille des données externalisées (Figure II.13). Cette croissance a créé un besoin d'une approche qui permet une vérification continue des données avec des exigences minimes de calcul et transmission.

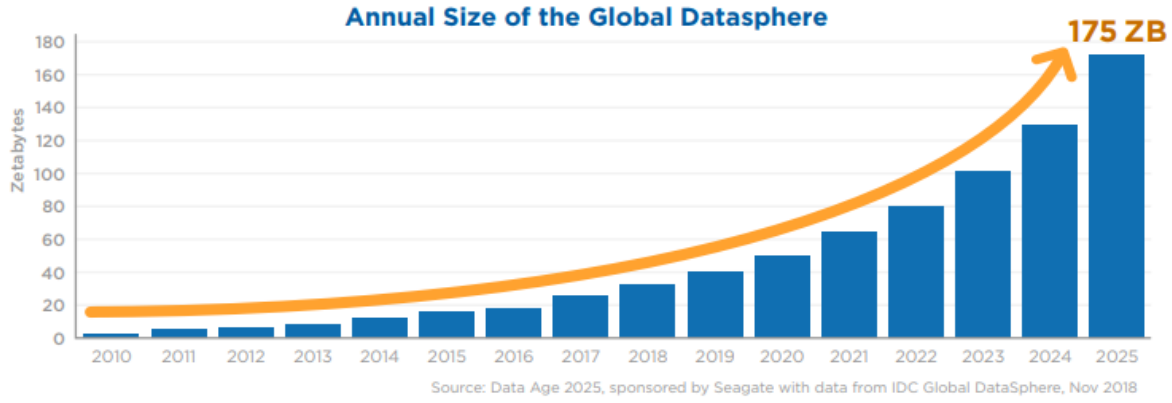


FIGURE II.13 – La taille globale des données dans le monde [28].

II.2.2 Preuve de possession (Proof of data Possession(POP))

Le concept des preuves probabilistes d'intégrité a été proposé par ATENIESE en 2007 [4], dans ce type d'approche, il suffit de vérifier l'intégrité d'un nombre limité de blocs dans un fichier pour prouver d'une manière probabiliste l'intégrité de la totalité des données (Équation II.8) avec un taux de vérification qui augmente avec le nombre de blocs vérifiés.

$$P_x = 1 - \prod_{x=0}^{c-1} \frac{n-t-x}{n-c} \tag{II.8}$$

Soit n le nombre de blocs de données supprimées par le fournisseur de service, ATENIESE et al [4] a prouvé que pour un nombre de blocs $c = 460$, le propriétaire des données peut avoir une garantie de 99% de détecter si le serveur supprime $t = 1\%$ de ses données.

La méthode probabiliste proposée par ATENIESE se compose de deux phases principales :

— Pré-traitement : le propriétaire des données subdivise ses données en n blocs et calcule un tag pour chaque bloc de données avant d'externaliser ses données chez le fournisseur de services (figure II.14).

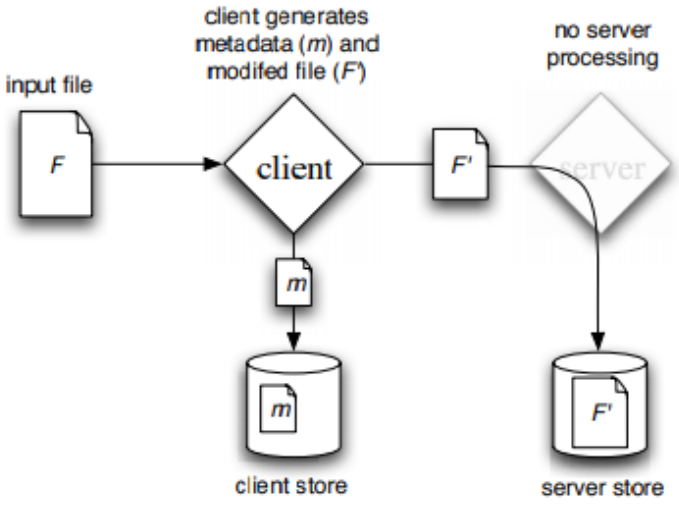


FIGURE II.14 – Pré-traitement des données [4].

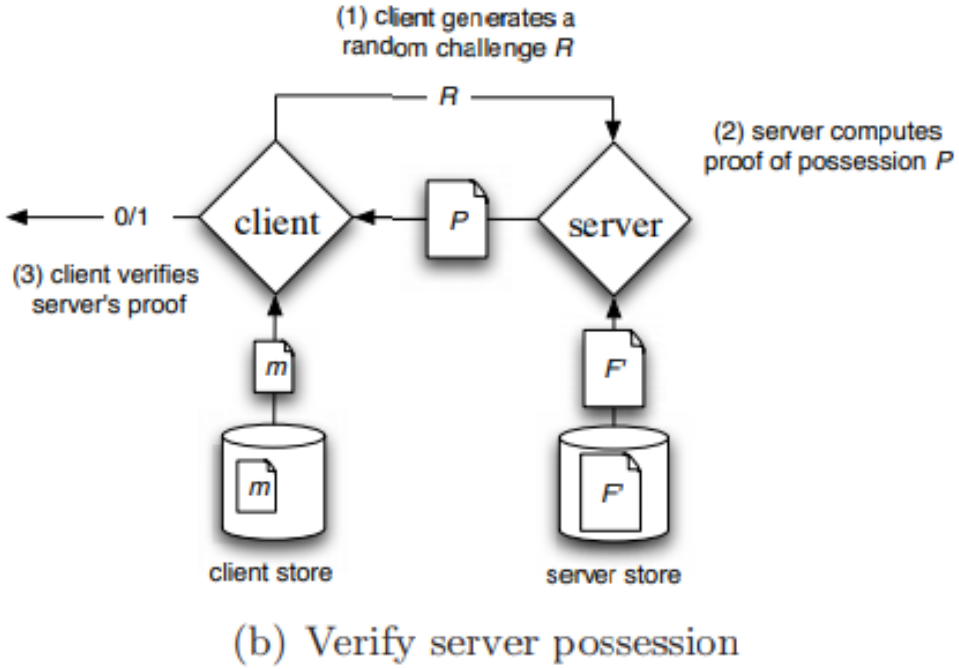


FIGURE II.15 – Vérification de possession des données [4].

— Vérification : le propriétaire choisit un nombre c de blocs aléatoires à vérifier et envoie une demande de vérification au serveur. Ce dernier calcule une preuve et renvoie les tags avec la preuve au propriétaire des données pour les vérifier (figure II.15).

Malgré que cette méthode offre la possibilité d'avoir une vérification illimitée et simple des données stockées chez un fournisseur de services, la vérification doit être toujours effectuée par le propriétaire des données d'une manière cyclique. Afin de réduire le coût de vérification chez l'utilisateur de service de stockage, plusieurs méthodes de vérification publiques d'intégrité ont été proposées. Dans ce type de méthodes un auditeur tiers peut effectuer la vérification sans intervention directe du propriétaire des données [106, 110, 108, 123, 124, 122].

II.2.3 Preuve de récupérabilité (Proof of Retrievability(POR))

La notion de preuve de récupérabilité Preuve de récupérabilité (POR) est proposée par JUELS et KALISKI [53]. Le concept de POR est basé sur celui de Proof of Knowledge POR [9], c'est-à-dire que s'il parvient à réussir le contrôle d'intégrité des données, le vérificateur peut récupérer le fichier non modifié auprès du fournisseur de service de stockage [35]. Le système de POR de Juels et de Kaliski utilise des codes de vérification ponctuels et de correction d'erreurs pour assurer, d'une part, la possession des données (le CSP stocke toujours les données dans ses serveurs), d'autre part, la "possibilité de récupération" des fichiers de données sur des systèmes de stockage distants. Cependant, leur approche est limitée par le fait que la vérification de l'intégrité doit être effectuée par le propriétaire des données, c'est-à-dire que les données ne sont pas publiquement vérifiables.

SHACHAM et WATERS [93] ont par la suite fourni deux nouvelles méthodes de POR efficaces avec une sécurité prouvée. Reposant sur une idée totalement différente de celles de JUELS et de KALISKI. Ils ont présenté deux systèmes de POR compacts, l'un pour l'audit privé et l'autre pour l'audit public. Les deux systèmes sont très efficaces et sécurisés. Leur première méthode est basée sur l'algorithme de signature BLS [15] et peut être appliqué à la vérification publique, mais la sécurité totale a été prouvée dans le modèle oracle aléatoire. Leur deuxième schéma est construit sur une fonction pseudo-aléatoire ne permettant qu'un audit privé, mais sa sécurité est prouvée dans le modèle standard. Le système d'audit publique de SHACHAM et WATERS sert de base à plusieurs contrôles d'audit tiers effectués dans [106] et à d'autres travaux connexes.

II.2.4 Vérifications menées par un auditeur externe

Le concept de l'audit public dans la vérification à distance de l'intégrité des données est d'abord proposé par ATENIESE et al. [4] en 2007, qui introduit la notion d'auditabilité publique dans leur modèle Preuve de possession (POP) pour garantir la possession de fichiers chez des fournisseur de service de stockages non fiables. Cependant, cette méthode utilise la combinaison linéaire de blocs échantillonnés lors de la vérification par un auditeur externe, cela signifie que le protocole ne préserve pas la confidentialité et risque donc de transmettre des informations de données utilisateur à l'auditeur externe [106]. SHAH et al. [94] ont proposé d'utiliser un auditeur externe (TPA) pour maintenir le stockage en ligne honnête en chiffrant d'abord les données, puis en envoyant un certain nombre de hachages pré-calculés à clé symétrique sur les données cryptées à l'auditeur. L'auditeur vérifie ensuite l'intégrité du fichier de données et la possession par le serveur d'une clé de déchiffrement préalablement validée. Par contre cette méthode souffre d'une limitation de nombre de vérifications que l'auditeur peut effectuer.

Le premier système d'audit par auditeur externe préservant la confidentialité a été proposé par WANG et al. [112, 105]. Cette méthode a offert un moyen de préserver la confidentialité en masquant la combinaison linéaire de blocs de données échantillonnées, cela assure que le TPA ne pourrait pas être en mesure de récupérer les données à partir des messages de vérifications. Malgré les avantages de la méthode proposée, le coût de vérification reste élevé chez le vérificateur de données [100] (que se soit externe ou le propriétaire des données.).

Conclusion

L'externalisation des données chez des fournisseurs de services a crée de nouvelles exigences pour la sécurité et l'intégrité des données telles que :

Sécurité et contrôle d'accés : l'utilisation d'une méthode de chiffrement par attributs ($(KP-ABE, CP-ABE...)$) permet au propriétaire de chiffrer ses données et de les partager d'une manière simple avec plusieurs utilisateurs en définissant une politique d'accès associée. Le choix de cette politique d'accès, dans le cas de $CP-ABE$ par exemple, donne un certain niveau d'intelligence au fichier chiffré en lui permettant de choisir les utilisateurs autorisés en se basant sur leurs privilèges (les attributs associés à leurs clés). Le choix du type de structure d'accès impacte non seulement la politique d'accès, mais aussi la flexibilité des structures d'accès et la performance de la méthode de chiffrement.

Intégrité et détection d'anomalies : l'utilisation de la vérification probabiliste proposée par [4] permet de faire une vérification efficace des données stockées chez le fournisseur de service. Ce type de vérification nécessite l'interaction directe du propriétaire de données avec le fournisseur de services, ceci va imposer un fardeau important de transmission et calcul chez le propriétaire des données.

Dans les deux prochains chapitres nous allons :

— Étudier l'impact d'utilisation d'une structure d'accès à seuil (au lieu d'une structure binaire) sur une variante de $CP-ABE$.

— Proposer une approche de gestion de contrôle d'accès avec un calcul sous-traité chez un serveur proxy.

— Proposer une méthode de vérification publique d'intégrité avec un fardeau de calcul et de stockage minimal.

Chapitre III

Confidentialité et gestion de contrôle d'accès

Sommaire

Introduction

Avec le développement rapide des services de stockage Cloud, les problèmes de sécurité des données deviennent de plus en plus critiques à cause de l'externalisation des données sous le contrôle du fournisseur de services.

Cela signifie qu'on ne peut pas garantir la sécurité des données contre les attaques internes et externes [103].

Pour résoudre le problème de confidentialité, le propriétaire des données peut choisir d'utiliser l'infrastructure *PKI*. La solution la plus pratique consiste à chiffrer les données pour renforcer la confidentialité des données avant de les télécharger sur les serveurs Cloud. Malgré que cela garantisse la confidentialité des données, une telle implémentation nécessiterait une version chiffrée des données avec la clé publique de chaque utilisateur avec lequel les données sont partagées. Cela augmenterait à la fois la surcharge de calcul et celle de stockage pour le propriétaire des données. Cela signifie que le propriétaire des données a besoin d'une solution qui garantit la confidentialité des données et la gestion de contrôle d'accès.

De nombreuses solutions ABE [67, 21, 48, 97, 30, 89, 119, 67, 96, 125, 52] ont été proposées, telles solutions peuvent fournir un contrôle d'accès flexible et granulaire sans l'intervention du fournisseur de services (CSP). En effet, les attributs de l'utilisateur n'appartiennent pas à une seule autorité qui contrôle les clés secrètes des attributs et génère les clés de l'utilisateur (par exemple, dans le cas de dossiers de santé en ligne, l'enregistrement peut être partagé avec un médecin appartenant à un autre hôpital). C'est pour cette raison que des solutions multi-autorités ont été proposées[18, 97, 79, 117, 70, 98, 66, 64, 84, 20].

L'efficacité des différentes opérations de la méthode est importante, car les utilisateurs de services Cloud n'utilisent pas seulement des appareils puissants pour accéder aux données. Cela signifie que toute méthode adoptée doit prendre en compte la puissance de calcul des utilisateurs, en particulier l'opération de déchiffrement pouvant être utilisées à plusieurs reprises pour accéder à différents fichiers stockés sur le serveur Cloud. De plus, étant donné que les données peuvent être consultées par un grand nombre d'utilisateurs, un processus de révocation est nécessaire pour assurer un contrôle d'accès souple et affiné [70]. Le processus de révocation doit permettre au propriétaire de mettre à jour facilement les droits d'accès pour un fichier de données. En outre, les autorités d'attributs doivent pouvoir mettre à jour les attributs des utilisateurs tout en maintenant la sécurité en amont et en aval des données stockées.

Dans ce chapitre nous proposons :

— Une méthode qui étend la flexibilité de P2E [30] ,en utilisant des structures d'accès à seuil plutôt que des structures d'accès binaire, avec un processus de révocation capable de maintenir une sécurité en avant et en arrière[1] sans entraîner de surcharge importante sur les différents acteurs du système.

1. voir la section 1.2.2 pour plus de détails.

— Une méthode capable de réaliser un contrôle d'accès granulaire tout en maintenant la confidentialité des données avec un processus efficace de révocation d'attribut capable de maintenir à la fois la sécurité en avant et en arrière. De plus, nous augmentons l'efficacité de notre processus de déchiffrement en utilisant un serveur de déchiffrement proxy qui effectuera la plupart des calculs pour les utilisateurs. Nous utilisons le calcul parallèle afin de réduire la surcharge des opérations de chiffrement, génération de clés et déchiffrement proxy.

III.1 Généralités Mathématiques

III.1.1 Rappels Mathématiques

III.1.1.1 Groupes

Définition Un groupe $(G, *)$ est un ensemble d'éléments associé à une opération binaire $(*)$ qui prend deux éléments quelconques dans le groupe, et les combine pour former un troisième élément dans ce groupe [77]. Un ensemble est un groupe si l'ensemble et l'opération satisfont les quatre propriétés suivantes[47] :

— Loi de composition interne : $\forall a, b \in G, a * b \in G$

— Associativité : $\forall a, b, c \in G, a * (b * c) = (a * b) * c$

— Existence d'élément neutre : il existe un élément $e \in G$ tel que $\forall a \in G, a * e \in G$

— Inverse : $\forall x \in G, \exists y \in G$, tel que $x * y = e$

Groupe abélien un groupe abélien est un groupe fini d'ordre p (qui a un nombre d'éléments fini p) qui satisfait la propriété suivante [26] : $\forall a, b \in G, a * b = b * a$.

Un groupe abélien est appelé cyclique s'il existe un élément g tel que toute élément appartenant à G peut être obtenu en appliquant l'opération du groupe sur g. g est appelé générateur du groupe G [2].

Corps Un corps est un ensemble F, contenant au moins deux éléments, sur lesquels deux opérations $+$ et $\cdot$ (appelées respectivement addition et multiplication) sont définies de sorte que pour chaque paire d'éléments $x, y \in F$ il y a des éléments uniques $x + y$ et $x \cdot y$ dans F. Ces opérations doivent satisfaire les conditions suivantes :

— Associativité : $a + (b + c) = (a + b) + c$, et $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.

— Commutativité : $a + b = b + a$, et $a \cdot b = b \cdot a$.

— L'existence de l'identité additive et multiplicative : il existe deux éléments $0 \in F$ et $1 \in F$ tels que $a + 0 = a$ et $a \cdot 1 = a$.

— L'existence d'inverses additifs : pour tout a dans F, il existe un élément $-a \in F$, appelé l'inverse additif de a, tel que $a + (a) = 0$.

— L'existence de l'inverse multiplicatif : pour chaque $a \neq 0$ il existe $a^{-1} \in F$ tel que $a \cdot a^{-1} = 1$

— Distributivité : $(a + b) \cdot c = a \cdot c + b \cdot c$ et $a \cdot (b + c) = a \cdot b + a \cdot c$

Corps finis Un corps fini (F) ou champ de Galois est un corps contenant un nombre fini d'éléments. Comme avec n'importe quel champ, un corps fini est un ensemble sur lequel les opérations de multiplication, addition, soustraction et division sont définies et satisfont à certaines règles de base[2] :

1. $(F, +)$ est un groupe abélien avec une identité additive égal a 0.
2. $(F\{0\}, .)$ est un groupe abélien avec une identité multiplicative égal a 1.
3. $\forall a, b, c \in F, a * (b + c) = a * b + a * c$

III.1.1.2 Problème du logarithme discret

Soit G un groupe cycle d'ordre p et g un générateur de groupe G, le problème du logarithme discret ou LDP consiste à trouver, pour une valeur $h \in G$, un entier $x \in Z$ tel que $g^x \mod p = h$. Ceci revient à trouver le logarithme de h à base g, $log_g(h) = x$.

Lorsqu'on calcule un logarithme de h en base g où g est d'ordre fini égal à N, le nombre d'étapes de calcul est majoré par N fois le temps de calcul d'une opération de groupe. Ce qui signifie que pour des groupes d'ordre assez grand, le temps de calcul devient prohibitif.

III.1.1.3 Hypothèse décisionnelle de Diffie-Hellman (DDH)

Soit G un groupe cyclique d'ordre p, g le générateur de G et $a, b \in Z_p$, le problème de Diffie-Hellman revient à calculer g^{ab} en connaissant g^b, g^a, g, p. En effet ce problème est plus facile que celui du logarithme discret car il est résolu automatiquement si on peut résoudre le LDP, vu qu'on peut directement calculer g^{ab} si on peut trouver a ou b.

III.1.2 Génération d'une matrice de partage linéaire

Définition Une méthode de partage secret est linéaire si les propriétés suivantes sont satisfaites [7, 62, 72] :

1. Les éléments d'un secret de chaque acteur forment un vecteur sur un corps fini Zp

2. La méthode comprend une matrice $W(l \times n)$ avec l lignes et n colonnes. Pour chaque ligne i de la matrice $W(1 \leq i \leq l)$, il existe une fonction $\rho(i)$ mappant cette ligne à l'élément correspondant. Un vecteur v est défini par $v = (s, r_2, \ldots, r_n)$ avec s est le secret à partager, $s \in Z_p$ et $r_2, \ldots, r_n \in Z_p$ sont choisis au hasard pour masquer le secret. Le résultat de $W(i).v = (\lambda_1, \ldots, \lambda_l)$ est le vecteur d'action où chaque ligne (i) possède un élément.

Dans chaque méthode linéaire de partage de secret ($LSSS$), pour tout ensemble autorisé dans une structure d'accès, $I \subseteq 1, \ldots, l$ est défini comme étant $I = \{i : \rho(i) \in S\}$. Il existe un ensemble de constantes, $\{\omega_i \in Z_p\}_{i \in I}$, qui peuvent être utilisées avec des partages valides pour reconstruire le secret $\Sigma \omega_i \lambda_i = s$, qui peut être calculé dans un temps polynomial satisfaisant.

III.2 Assurer un contrôle d'accès sécurisé, flexible et efficace tout en préservant la confidentialité dans le Cloud Computing (FP2E)

Le service de stockage cloud est l'un des services les plus populaires proposés par les fournisseurs de services Cloud. Dans ce service, les utilisateurs peuvent externaliser leur stockage de données dans un espace de stockage tiers tout en maintenant un accès facile via plusieurs appareils à un coût minimal. Les utilisateurs du service de stockage peuvent partager leurs données avec plusieurs utilisateurs. Cependant, l'externalisation des données dans les serveurs d'un fournisseur de services élimine tout contrôle physique dont le propriétaire des données dispose sur ses données, ce qui expose ses données au risque d'attaques malveillantes. En effet, pour des données plus sensibles, le fournisseur de services Cloud peut essayer d'accéder aux données s'il a des avantages à en retirer. Ainsi, les propriétaires de données ont besoin d'un moyen pour assurer la confidentialité de leurs données tout en conservant tous les avantages procurés par l'utilisation du service.

X. DONG et al. [30] ont proposé une méthode efficace qui préserve la confidentialité (P2E). Dans cette méthode, la structure d'accès est transformée en formules booléennes qui sont ensuite converties en matrice LSSS (Linear Secret Sharing Scheme). Cependant, étant donné que les formules booléennes sont équivalentes aux structures d'accès binaires, l'utilisation de formules booléennes limite la flexibilité de leurs structures d'accès dans leur schéma, en plus cette transformation entraînera une augmentation importante du nombre d'attributs.

Dans cette section nous améliorons la méthode proposée par X. DONG en utilisant les structures d'accès monotones pour permettre aux utilisateurs de définir un niveau complexe de contrôle d'accès avec un nombre minimal d'attributs. Nous proposons aussi une méthode de révocation des droits qui permet de gérer (attribuer/révoquer) les droits d'accès des utilisations avec un impact minimal sur les différents acteurs du système.

III.2.1 Démarche de FP2E

Le propriétaire de données choisit un ensemble d'attributs à utiliser pour construire la structure d'accès. Cette structure d'accès est, ensuite, utilisée pour chiffrer le fichier de données. Nous utilisons des structures d'accès à porte seuil sans qu'il soit nécessaire de les convertir en formule booléenne équivalente, cette structure d'accès est ensuite convertie en matrice *LSSS* selon la méthode décrite par Liu [71] et al.

Les privilèges d'accès d'un utilisateur dans notre système sont décrits à l'aide d'un ensemble d'attributs, choisis par le propriétaire des données, et de son identifiant unique. Une partie de la clé est ensuite générée pour chaque attribut de l'ensemble d'attributs de l'utilisateur. Les utilisateurs ne peuvent accéder aux données cryptées que s'ils peuvent trouver, dans leur ensemble d'attributs, un sous-ensemble pouvant satisfaire à la politique d'accès du fichier de données.

La méthode proposée est caractérisée par cinq algorithmes :

— **Initialisation :** Le propriétaire des données définit un univers d'attributs à utiliser. Pour chaque attribut de l'univers défini, le propriétaire des données génère une clé publique et une clé secrète, puis publie les clés publiques de l'attribut.

— **Chiffrement :** Au cours de cette phase, le propriétaire des données choisit un ensemble d'attributs pour construire une structure d'accès à seuil pour le fichier de données. Cette structure d'accès est ensuite convertie en une matrice LSSS et utilisée pour chiffrer le fichier de données. Le propriétaire sous-traite ensuite le texte chiffré au serveur de Cloud.

— **Génération de clés :** Pour générer la clé de l'utilisateur, le propriétaire des données choisit un ensemble d'attributs définissant les privilèges de l'utilisateur, puis, pour chaque attribut, le propriétaire des données génère une clé d'attribut. Les clés d'attributs sont ensuite transférées à l'utilisateur.

— **Déchiffrement :** Lors de la récupération du fichier de données à partir du Cloud, l'utilisateur autorisé trouve un ensemble d'attributs dans son univers d'attributs pour satisfaire la politique d'accès, puis l'utilise pour déchiffrer les données.

— **Révocation :** Dans la phase de révocation, nous considérons deux cas : dans le premier cas, le propriétaire doit révoquer un ensemble d'attributs d'un utilisateur. Il génère donc les nouvelles clés d'attribut pour les utilisateurs non révoqués et une clé de mise à jour de texte chiffré, puis les transfère au CSP . Après cela, le serveur Cloud mettra à jour les textes chiffrés et remettra les clés à leurs utilisateurs respectifs. Dans le second cas, le propriétaire doit mettre à jour la structure d'accès d'un fichier spécifique. Il chiffrera donc les données à l'aide de la nouvelle politique d'accès et transmettra le nouveau texte chiffré au serveur Cloud.

III.2.2 Modélisation de FP2E

Description du Modèle du Système

Le modèle de système dans notre méthode comprend quatre entités principales :

— **Fournisseur de services (CSP) :** offre un espace de stockage et une puissance de calcul sous forme de services dont les utilisateurs peuvent bénéficier. Il est également responsable de la maintenance des différents services fournis.

— **Propriétaire des données :** utilisateur du service bénéficiant de l'espace de stockage offert par le CSP, le propriétaire des données sous-traite ses données aux serveurs de Cloud et les partage avec d'autres utilisateurs du service.

— **Générateur de clés privées (PKG) :** un tiers de confiance, génère et fournit les clés privées aux utilisateurs du système. Le PKG est également responsable de la transmission des identifiants ID et des clés publiques des utilisateurs aux propriétaires.

— **Utilisateur de données :** utilise le service Cloud pour télécharger les données partagées par le propriétaire des données et les déchiffre à l'aide de sa clé secrète.

Modèle de l'adversaire

Dans le modèle de l'adversaire, nous considérons deux types de menaces :

— **Menaces internes :** on considère le fournisseur de services comme semi-fiable dans notre système, cela veux dire que le CSP se comportera normalement la plupart du temps. Cependant, il peut essayer d'accéder aux données des utilisateurs s'il peut en tirer certains avantages. Les utilisateurs du service représentent aussi un risque de sécurité, car ils peuvent tenter d'accéder à des données non autorisées soit en collaborant entre eux ou avec le CSP.

— **Menaces extérieures :** menaces des adversaires au-delà du système.

Nous supposons que toutes les communications entre les acteurs de notre modèle sont sécurisées sous un protocole existant tel que Secure Sockets Layer (SSL).

III.2.3 Démarche de FP2E

Initialisation

Le propriétaire des données commence par choisir un grand nombre q, deux groupes G_1 et G_2, une application bilinéaire $e : G_1 \times G_1 \rightarrow G_2$ et une fonction de hachage H qui mappe un utilisateur ID_u à un élément de G_1. Ensuite, définit un ensemble d'attributs W qui sera utilisé pour partager des données. Pour chaque attribut $i \in W$, le propriétaire génère deux nombres aléatoires α_i, β_i, dans Z_p^*, de sorte que la clé secrète soit :

$$Sk = \{\alpha_i, \beta_i, i \in W\} \tag{III.1}$$

et publie la clé publique :

$$Pk = \{e(g_1, g_1)^{\alpha_i}, g_i^{\beta_i}, i \in W\}. \tag{III.2}$$

Chiffrement

Pour chiffrer un message M_a, le propriétaire commence par définir un ensemble d'attributs I. Sur la base de I, le propriétaire définit une structure d'accès, puis l'utilise pour générer la matrice de partage M et $\rho(x)$ la fonction qui mappe l'attribut x à la ligne correspondante dans la matrice à l'aide de la méthode de conversion [71]. Ensuite, Le propriétaire choisit deux vecteurs aléatoires v et ω, tels que la première entrée de v est s et la première entrée de ω est 0, puis calcule $\lambda_x = v \times M_x$ et $\omega_i = \omega \times M_i$. (avec i l'indice de la ligne correspondante à l'attribut x). Pour chaque attribut x, le propriétaire choisit un nombre aléatoire $r_x \in Z_p$, puis calcule :

$$\begin{aligned} D_{\rho(x),1} &= e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \\ D_{2,\rho(x)} &= g_1^{r_{\rho(x)}} \\ D_{\rho(x),3} &= g_1^{\beta_{\rho(x)} r_{\rho(x)}} g_1^{\omega_{\rho(x)}} \end{aligned} \tag{III.3}$$

Le message M_a est chiffré en utilisant un algorithme de chiffrement symétrique avec $e(g, g)^s$ comme clé : $M_e = Enc^{sym}_{e(g,g)^s}(M)$

Enfin, le propriétaire des données envoie le fichier chiffré au CSP
$D = \{\forall x, D_{\rho(x),1}, D_{\rho(x),2}, D_{\rho(x),3}; (M, \rho), Me\}$.

Génération de clés

le propriétaire des données reçoit l'identité de l'utilisateur ID_u de la part du PKG et choisit un ensemble d'attributs I_u décrivant ses droits d'accès. La clé secrète de l'utilisateur est calculée par-suite comme suit :

$$Sk_u = \{g_1^{\alpha_i} H(ID_u)^{\beta_i}, i \in I_u\}. \tag{III.4}$$

La clé générée Sk_u est chiffrée en utilisant la clé publique de l'utilisateur avant de la livrer, soit directement ou via le CSP, de cette façon seul l'utilisateur peut récupérer sa clé secrète en utilisant des couples classiques de clés privé/publique (a titre d'exemple RSA).

Déchiffrement

L'utilisateur récupère le message chiffré D depuis le serveur puis il commence le processus de déchiffrement. L'utilisateur commence par choisir l'ensemble d'attributs qui satisfait la politique d'accès à partir de ses attributs I_u, puis calcule :

$$\begin{aligned}
&\prod_x \left(\frac{D_{\rho(x),1} \cdot e(H(ID), D_{\rho(x),3})}{e(sk_{\rho(x),u}, D_{\rho(x),2})}\right)^{\Delta_\rho(x)} \\
&= \prod_x \left(\frac{e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1^{\beta_{\rho(x)} r_{\rho(x)}} g_1^{\omega_{\rho(x)}})}{e(g_1^{\alpha_{\rho(x)}} H(ID_u)^{\beta_{\rho(x)}}, g_1^{r_{\rho(x)}})}\right)^{\Delta_{\rho(x)}} \\
&= \prod_x \left(\frac{e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1)^{\beta_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID), g_1)^{\omega_{\rho(x)}}}{e(g_1, g_1)^{\alpha_{\rho(x)} r_{\rho(x)}} \cdot e(H(ID_u), g_1)^{\beta_{\rho(x)} r_{\rho(x)}}}\right)^{\Delta_{\rho(x)}} \\
&= \prod_x \left(e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(H(ID), g_1)^{\omega_{\rho(x)}}\right)^{\Delta_{\rho(x)}} \\
&= e(g_1, g_1)^s \cdot e(H(ID), g_1)^0 \\
&= e(g_1, g_1)^s
\end{aligned} \tag{III.5}$$

l'utilisateur récupère par-suite la message claire en utilisant la clé $e(g_1, g_1)^s$ avec l'algorithme de chiffrement symétrique :
$Ma = Dec^{sym}_{e(g_1,g_1)^s}(Me)$.

Révocation

Le processus de révocation dans notre méthode met à jour uniquement les clés des utilisateurs non révoqués associées aux attributs révoqués cela nous permet de minimiser l'impact sur l'utilisabilité de la méthode. On considère deux scénarios pour l'opération de révocation.

— **Scénario 1** : nous avons un utilisateur ou un groupe d'utilisateurs dont les privilèges d'accès sont révoqués ou modifiés. Cette opération nécessitera la mise à jour des fichiers chiffrés et des clés des utilisateurs.

— **Scénario 2** : le cas de la révocation des droits d'accès d'un utilisateur ou d'un groupe d'utilisateurs à un fichier chiffré spécifique. Ce processus ne nécessite que la mise à jour de la stratégie d'accès du fichier chiffré.

Révocation d'attributs : la révocation d'attributs dans notre système suit deux étapes principales :

Mise a jours des clés des utilisateurs : Pour chaque attribut révoqué i, le propriétaire génère un nouvel $\alpha_i, \beta_i \in Z_p$ et met à jour ses clés (privé et publique).Ensuite, pour chaque utilisateur U non révoqué, le propriétaire génère une nouvelle clé de l'attribut mis à jour :

$$sk_{u,i} = g_1^{\alpha_i} H(ID_u)^{\beta_i} \tag{III.6}$$

Ensuite le propriétaire chiffre $sk_{u,i}$ à l'aide de la clé publique de l'utilisateur, puis envoi les clés au CSP qui les remettra aux utilisateurs respectifs.

Mise à jour du message chiffré : pour mettre à jour le message chiffré, le propriétaire commence par créer les modificateurs associés aux attributs modifiés $CUK = cukc_1, c_2$, avec $c_1 = \alpha_{i,old} - \alpha_{i,new}$ et $c_2 = \beta i, old - \beta i, new$.

Le propriétaire des données envoi les modificateurs au CSP qui met à jour les messages chiffrés en calculant :

$$\begin{aligned} D_{1,new} &= \frac{D_1}{e(D_2, g_1)^{c_1}}, \\ &= e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\frac{\alpha_{old} \cdot r_x}{(\alpha_{old} - \alpha_{new}) \cdot r_x}}, \\ &= e(g_1, g_1)^{\lambda_{\rho(x)}} \cdot e(g_1, g_1)^{\alpha_{new} \cdot r_x}. \end{aligned} \tag{III.7}$$

et

$$\begin{aligned} D_{3,new} &= \frac{D_3}{(D_2)^{c_2}}, \\ &= g_1^{\omega_{\rho(x)}} \cdot g_1^{\frac{\beta_{old}}{\beta_{old}-\beta_{new}} \cdot r_x}, \\ &= g_1^{\omega_{\rho(x)}} \cdot g_1^{\beta_{new} \cdot r_x}. \end{aligned} \tag{III.8}$$

Après l'opération de mise à jour du texte chiffré, les données révoquées sont dans l'un des états suivants :

— **État 1** : Jamais acquises par l'utilisateur révoqué ; par conséquent, la clé de chiffrement n'a pas besoin d'être mise à jour, car l'utilisateur révoqué ne l'a pas précédemment acquise et ne pourra pas la récupérer après la mise à jour du texte chiffré.

— **État 2** : Auparavant acquises par l'utilisateur révoqué, le propriétaire n'a donc qu'à re-chiffrer les données lorsque le contenu est mis à jour, car l'utilisateur révoqué n'acquerra aucune nouvelle information même s'il déchiffre les données avant l'opération de mise à jour.

Dans les deux cas, le propriétaire n'aura pas besoin de re-chiffrer les données jusqu'à la prochaine mise à jour.

Changement de politique d'accès : Le propriétaire des données peut modifier les droits d'accès d'un ensemble d'utilisateurs (octroi ou révocation de droits) au fichier de données comme suit :

il choisit d'abord une nouvelle clé de chiffrement s_{new}. il définit, ensuite, la nouvelle politique d'accès A_{new} et génère la Matrice de partage linéaire correspondante (M_{new}, ρ_{new})), puis calcule le nouveau fichier chiffré à l'aide du processus décrit dans la section III.2.3. Le propriétaire envoie ensuite le nouveau fichier D_{new} au CSP.

Après avoir reçu le message de mise à jour du texte chiffré du propriétaire, le serveur Cloud remplace l'ancien fichier chiffré par le nouveau.

III.2.4 Analyse et évaluation de FP2E

Sécurité : l'objectif principal de notre construction est d'améliorer la flexibilité des structures d'accès utilisé dans la méthode [30]. Cela signifie que notre méthode diffère de celle de DONG et al. en terme de structure d'accès et méthode de révocation. De plus, cette méthode est sécurisée dans le modèle générique de groupe bilinéaire par la même preuve de sécurité que la méthode de DONG et al.[30].

Contrôle d'accès granulaire : le propriétaire des données doit pouvoir gérer facilement le contrôle d'accès des différents fichiers stockés sans affecter la confidentialité des données. En outre, les utilisateurs ne devraient pas pouvoir accéder aux données pour lesquelles ils ne sont pas autorisés.

Preuve : Chaque utilisateur U reçoit un ensemble d'attributs décrivant les privilèges qui lui sont attribués par le propriétaire des données. Seul un utilisateur possédant les attributs requis peut trouver $\Sigma_x \lambda_{\rho(x)} \cdot \delta_{\rho(x)} = s$ et $\Sigma_x \omega_{\rho(x)} \cdot \delta_{rho(x)} = 0$ et récupérer la clé $e(g,g)^s$, cela signifie que seul les utilisateurs autorisés peuvent acquérir la clé de déchiffrement, ni le CSP ni l'utilisateur non autorisé ne peuvent acquérir la clé de déchiffrement. En plus, en utilisant les structure d'accès à seuil, notre système permet au propriétaire des données d'obtenir un contrôle d'accès plus fin sur les données externalisées.

Résistance à la collusion : une collusion entre utilisateurs ou entre utilisateurs et le CSP ne devrait pas permettre aux utilisateurs ou au CSP d'obtenir des données pour lesquelles ils ne sont pas autorisés.

Preuve : La clé de chaque utilisateur dans notre méthode est liée à sa propre identité $l'ID_u$, cela signifie que Même si plusieurs utilisateurs combinent leurs clés, ils seront incapables de récupérer la clé $e(g_1,g_1)^s$, car ils doivent annuler la composante $e(H(ID),g_1)^{\omega_{\rho(x)}}$ de l'équation, ceci n'est pas possible vu que chaque utilisateur a un identifiant différent. Par conséquent, notre méthode est robuste contre la collusion.

La confidentialité des informations des utilisateurs : Le CSP ne devrait pas pouvoir accéder aux informations confidentielles des utilisateurs.

Preuve : La distribution des clés, que se soit lors de la génération ou de la révocation des droits d'accès, est faite en utilisant la clé publique des différents utilisateurs, cela signifie que notre méthode ne divulgue aucune information sur la clé secrète des utilisateurs ce qui permet d'assurer la confidentialité des informations vis à vis les différents acteurs du système.

La sécurité en avant : Après le processus de révocation, les utilisateurs révoqués ne devraient plus avoir accès au fichiers autorisés avant la révocation.

Preuve : Les données stockées dans les serveurs du CSP sont liées à une politique d'accès qui garantis que seul les utilisateurs autorisés peuvent y accéder. Lors de

la révocation des droits ou du changement des droits d'accès, la politique des fichiers est mise à jour de tel façon que les attributs restent homogènes entre les différents utilisateurs (existant et nouveaux). Cela signifie que les nouveaux utilisateurs peuvent accéder aux données précédemment externalisées quand ils ont les attributs nécessaires pour satisfaire la politique d'accès au texte chiffré. Par conséquent, la méthode respecte l'exigence du sécurité en avant.

La sécurité en arrière : lorsque de nouveaux utilisateurs rejoignent le système, ils devraient pouvoir accéder aux données précédemment partagées, dans la mesure où ils répondent aux critères définis par les règles d'accès.

Preuve : Notre système de révocation (décrit dans la section III.2.3) met à jour les attributs des différents utilisateurs et le texte chiffré, cela signifie que tant que le CSP exécute complètement le processus de révocation, même si l'utilisateur révoqué a accès aux fichiers chiffrés à l'aide des attributs révoqués, il ne pourra pas récupérer la nouvelle clé de chiffrement. Seul l'utilisateur disposant de l'attribut mis à jour de clés peut déchiffrer le texte chiffré. D'où la méthode de révocation satisfait l'exigence de la sécurité en arrière.

Performance

L'analyse de performance de la méthode se focalise sur les opération de chiffrement, déchiffrement et révocations :

Performance de chiffrement et déchiffrement : La surcharge de calcul du processus de cryptage dans notre méthode est similaire à P2E, l'opération de couplage est l'opération la plus coûteuse pour chaque fichier de données différent. Cependant, le propriétaire calcule $e(g_1, g_1)$ une seule fois.

Dans le processus de chiffrement, le propriétaire des données doit effectuer deux multiplications scalaires pour calculer $D_{(x,1)}$, une multiplication scalaire pour $D_{(x,2)}$ et une autre pour $D_{(x,3)}$ pour chaque feuille de la structure d'accès. Par conséquent, le propriétaire des données a besoin d'au plus $4|I|$ multiplications scalaires, la complexité de chiffrement est donc $O(|I|)$, avec $|I|$ le nombre de nœuds d'extrémité dans la structure d'accès. Pour récupérer les données, l'utilisateur doit effectuer au plus une multiplication scalaire pour chaque nœud feuille plus une multiplication scalaire pour chaque nœud non feuille de la structure d'accès pour faire les calcules décrit dans la section III.2.3 , la complexité de déchiffrement est donc $O(max(|I|, N))$.

Par-contre, L'utilisation de la structure d'accès à seuil dans notre méthode (FP2E) réduit le coût du processus de chiffrement par rapport à P2E, car la transformation d'une structure d'accès à seuil en une structure d'accès binaire entraînera une augmentation importante du nombre de feuilles (figure III.1 illustre un exemple de transformation de la structure d'accès). Cette augmentation du nombre d'attributs entrainera une augmentation de la surcharge de chiffrement, cela signifie que la transformation d'une structure d'accès complexe en une structure d'accès binaire entraînera une augmentation importante de la charge de chiffrement chez le propriétaire des données, ce qui rend notre méthode plus efficace et plus flexible [2].

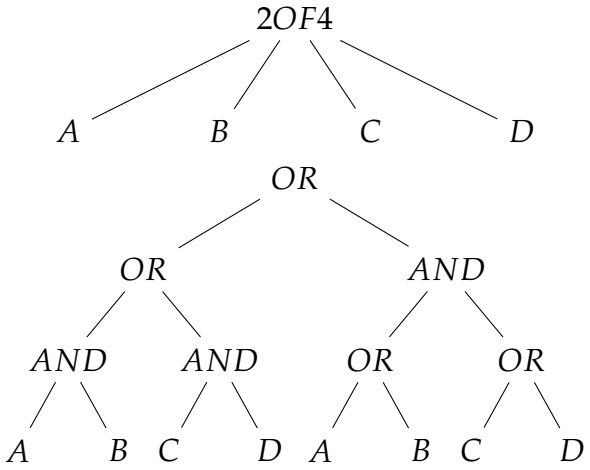


FIGURE III.1 – Transformation d'une structure d'accès a seuil (2 of 4) vers une structure d'accès binaire

Performance de révocation Pour un attribut révoqué x, soit nb_x le nombre d'utilisateurs non révoqués qui détiennent les attributs révoqués et $nb_(c,x)$ le nombre de textes chiffrés contenant l'attribut révoqué.

Les droits d'accès d'un utilisateur dans les méthodes basés sur **CP-ABE** sont définis avec un ensemble d'attributs lors de la génération de clé. *DACC* [89] a proposé un système de révocation d'accès dans lequel le propriétaire re-chiffre les données et met à jour $C_{(1,x)}$. Cette méthode garantit la sécurité en avant en transférant le nouveau $C_{(1,x)}$ aux utilisateurs non révoqués. *P2E* a suivi la même ligne de pensée, la seule différence est qu'au lieu de transférer le $C_{(1,x)}$ mis à jour aux utilisateurs non révoqués, P2E met à jour le texte chiffré stocké chez le CSP et fait confiance en lui pour ne pas transférer $C_{(1,x)}$ aux utilisateurs révoqués.

En fait, les deux processus de révocation proposés par les systèmes *DACC* et *P2E* ne respectent pas nos exigences de sécurité. La méthode de révocation de *DACC*

2. voir section III.2.4 pour plus de détails

Méthode	Message de révocation	Entité résponsable de mise a jour du texte chiffré
FP2E	$\mathcal{O}(nb_x)$	CSP
DACC	$\mathcal{O}(nb_x \cdot nb_{c,x})$	Propriétaire des données
P2E	$\mathcal{O}(nb_{c,x})$	Propriétaire des données

TABLE III.1 – Comparaison de coûts de révocation entre notre méthode, DACC et P2E

impose aux utilisateurs de conserver une partie du texte chiffré associé. Cette méthode de révocation ajoute une surcharge de stockage de révocation aux utilisateurs et ne garantit pas la sécurité en arrière, car les nouveaux utilisateurs ne peuvent pas accéder aux données précédemment stockées. Bien que la méthode *P2E* assure la sécurité en arrière en mettant à jour le texte chiffré stocké dans le Cloud, il compte sur le serveur pour appliquer la sécurité en avant, ce qui n'est pas toujours possible car le CSP est considéré comme étant semi-fiable et peut tenter de s'associer avec les utilisateurs révoqués afin de récupérer des données chiffrées.

Dans notre méthode de révocation, nous assurons la sécurité en avant et en arrière. Nous comptons uniquement sur le CSP pour mettre à jour le texte chiffré et transférer les nouvelles clés d'attributs chiffrées aux utilisateurs non révoqués. De plus, dans notre méthode, les nouveaux utilisateurs ne nécessitent aucune connaissance préalable des opérations de révocation précédentes, ce qui leur garantit d'accéder aux données préexistantes. Les utilisateurs révoqués se voient refuser l'accès aux anciens et nouveaux fichiers à l'aide des attributs révoqués. En fait, le processus de révocation de *P2E* et de *DACC* impose un coût de calcul important pour le propriétaire, car il doit mettre à jour manuellement et re-chiffrer chaque texte chiffré concerné par ce processus. Dans notre méthode, le propriétaire n'a besoin que de générer la nouvelle clé d'attribut pour chaque utilisateur révoqué et de générer une clé *CUK* que le serveur du Cloud utilisera pour mettre à jour le texte chiffré concerné.

De plus, l'opération de re-chiffrement dans notre méthode n'est effectuée que lorsque cela est nécessaire, car même après le processus de révocation, à moins que les données ne soient mises à jour sans changer l'ancienne clé de chiffrement, les utilisateurs révoqués ne gagneront aucune information nouvelle même sur la clé de cryptage précédemment acquise.

Dans le tableau III.1, nous remarquons que la taille du message de révocation dans notre méthode est linéaire par rapport à nb_x, cela signifie qu'il est inférieur à la taille du message dans *DACC* et *P2E*, car le coût de leurs méthodes de révocation est lié à $nb_{(c,x)}$, ce qui rend notre révocation plus efficace et plus sûre que les méthodes DACC et P2E [89, 30].

Expérimentations

Dans l'évaluation expérimentale, nous varions la taille des structures d'accès en augmentant le nombre d'attributs utilisés, ce qui nous permet de tester la rapidité des opérations de cryptage, de génération de clé et de déchiffrement. L'ensemble du système d'expérimentation est implémenté en langage Java sur une machine Windows 7 avec Core 2 duo fonctionnant à 2,9 GHz. Nous effectuons trois évaluations expérimentales :

1. Nous utilisons des structures d'accès à seuil pour analyser les performances de la méthode proposé et CP-ABE.
2. Nous utilisons des structure d'accès pour comparer la méthode proposée a P2E, cela nous permet de montrer que, même en utilisant des structures d'accès binaires, nous sommes toujours en mesure de maintenir une efficacité presque identique à celle de la méthode P2E.
3. Nous utilisons des structures d'accès à porte de seuil construites principalement avec des nœuds d'accès (3 sur 4) et générons la structure d'accès binaire décrivant la même politique d'accès, puis nous utilisons ces structures d'accès pour illustrer l'efficacité obtenue par ces structures d'accès à seuil avec une politique d'accès complexe.

Résultats expérimentaux 1 Afin de tester la vitesse des différents processus dans notre méthode et dans CP-ABE, nous varions le nombre d'attributs utilisés dans les structures d'accès à seuil entre 4 et 10 000. Les résultats de cette simulation (illustrés dans les figures (III.2 ,III.3)) montrent que notre méthode permet de réduire le temps de chiffrement et de génération de clé par rapport à CP-ABE. Le coût de déchiffrement reste le même pour les deux méthodes (figure III.4).

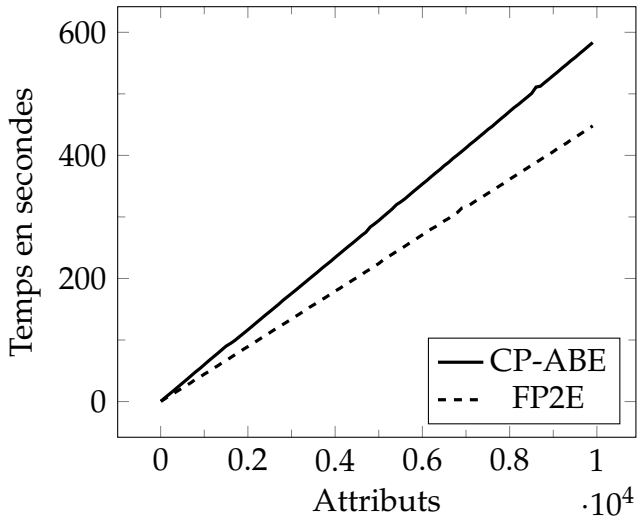


FIGURE III.2 – Coût de chiffrement de FP2E et CP-ABE

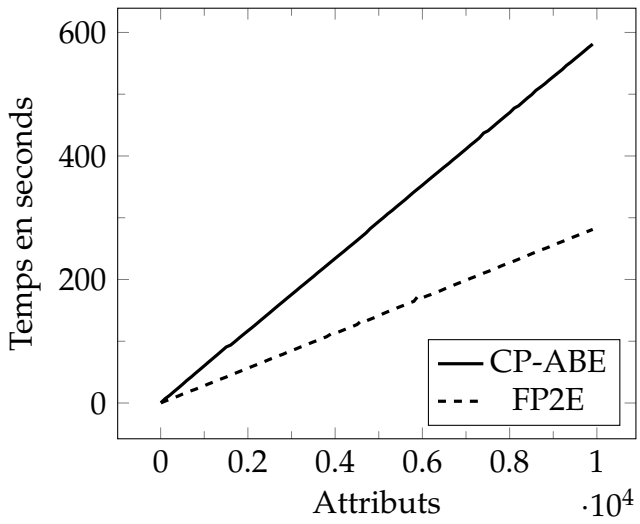


FIGURE III.3 – Coût de génération de clés de FP2E et CP-ABE

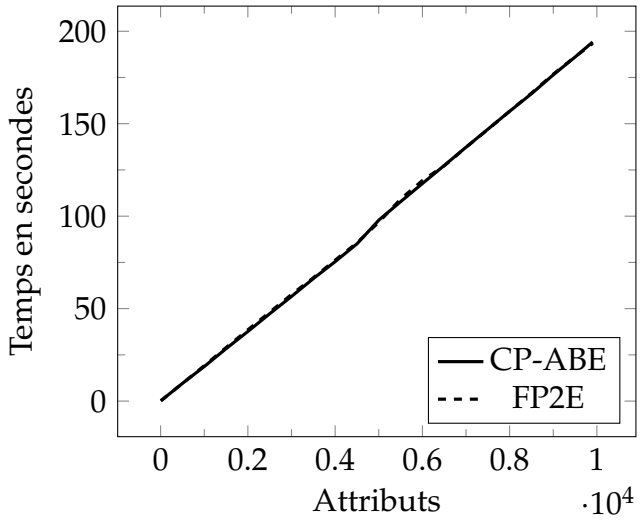


FIGURE III.4 – Coût de déchiffrement de FP2E et CP-ABE

Résultats expérimentaux 2 : Dans cette simulation, nous montrons que l'utilisation de la structure d'accès à seuil au lieu de la structure d'accès binaire ne diminue pas l'efficacité des opérations de chiffrement et de la génération de clé, même lorsque'on utilise uniquement des structures d'accès binaire. Pour cette raison, nous n'utilisons que des structures d'accès binaires à la fois dans notre méthode et dans P2E, tout en variant le nombre d'attributs utilisés entre 2 et 5 000.

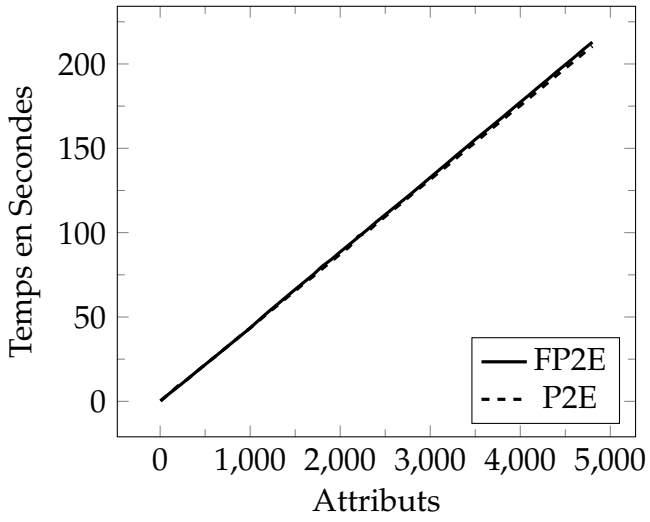


FIGURE III.5 – Comparaison de coût de chiffrement entre FP2E et P2E

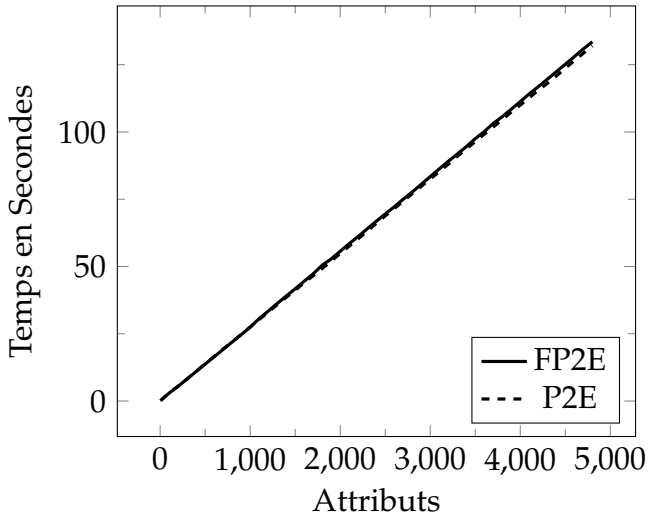


FIGURE III.6 – Comparaison de coût de génération de clés entre FP2E et P2E

Les résultats illustrés dans les figures (III.5,III.6,III.7) montrent que l'utilisation de la structure d'accès à seuil ajoute une légère augmentation de la charge de déchiffrement due à l'exponentiation du coefficient de Lagrange, tout en maintenant l'efficacité du chiffrement (figure III.5) et de la génération de clé (figure III.6). Cependant,

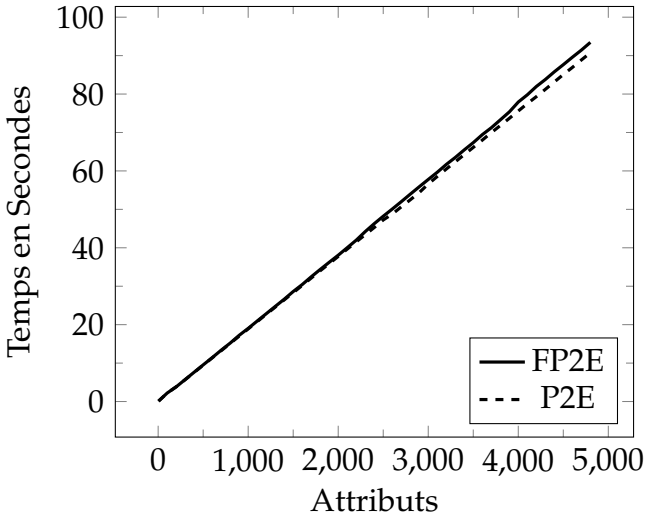


FIGURE III.7 – Comparaison de coût de déchiffrement entre FP2E et P2E

les structures d'accès binaires ne sont qu'un cas particulier des structures d'accès à seuil. Cela signifie que dans la plupart des cas, la transformation des structures d'accès à seuil en structures d'accès binaire se traduira par une augmentation importante du nombre d'attributs, ce qui confèrera à notre méthode un avantage considérable par rapport à P2E (voir les résultats expérimentaux 3 pour une illustration claire).

Résultats expérimentaux 3 : Pour illustrer l'effet de l'utilisation de la structure d'accès à seuil sur l'opération de chiffrement, nous utilisons un ensemble d'arbres d'accès à seuil utilisant uniquement le seuil (3 sur 4) dans les nœuds (figure III.8). Nous varions le nombre d'attributs en augmentant la profondeur des structures d'accès à seuil. Nous générons également les arbres binaires équivalents aux arbres à seuil (figure III.9).

	Niveau de profondeur=1	Niveau de profondeur=2	Niveau de profondeur=3	Niveau de profondeur=4	Niveau de profondeur=5
Structure d'accès a seuil	4	16	64	256	1024
Structure d'accès binaire	12	144	1728	20736	248832

TABLE III.2 – Nombre d'attributs nécessaire pour exprimer la même politique d'accès

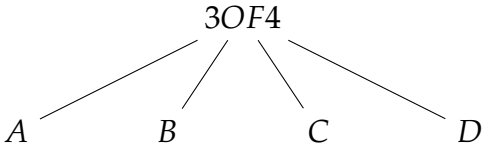


FIGURE III.8 – Exemple de structure d'accès a seuil avant la transformation

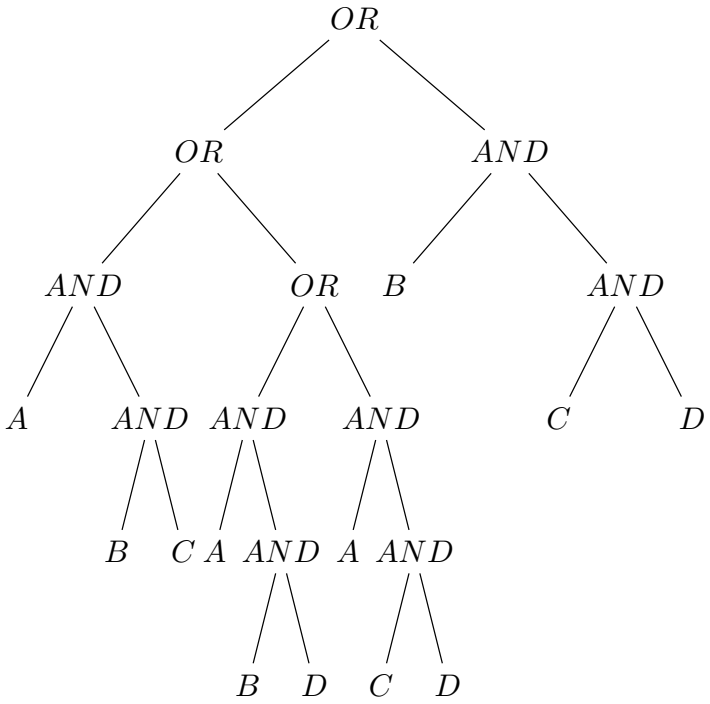


FIGURE III.9 – Exemple de structure d'accés a seuil après la transformation

Le tableau III.2 montre que pour exprimer le même niveau de droits d'accès que la structure d'accès à seuil, la structure d'accès binaire nécessite un nombre croissant d'attributs ; le nombre d'attributs n'augmente que lorsque les structures d'accès à seuil deviennent plus complexes (par exemple : pour avoir le même contrôle d'accès obtenu en utilisant 64 attributs dans un arbre d'accès à une porte de seuil, un arbre d'accès binaire nécessite 1728 attributs).

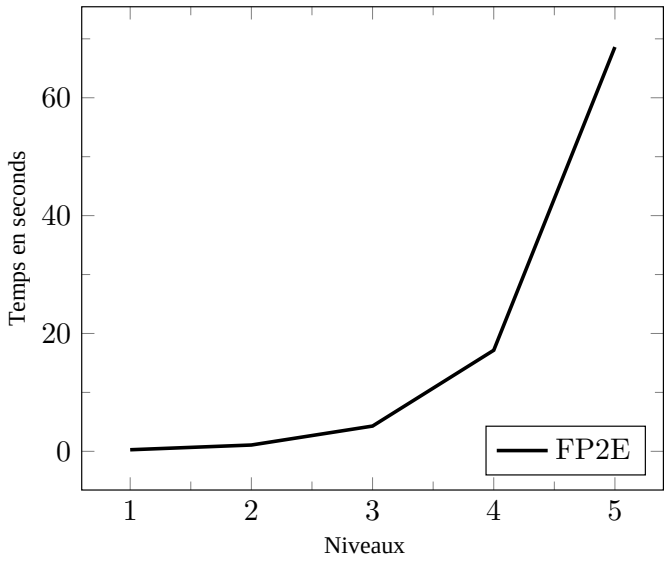


FIGURE III.10 – Coût temporel de notre méthode utilisant les structure a seuil

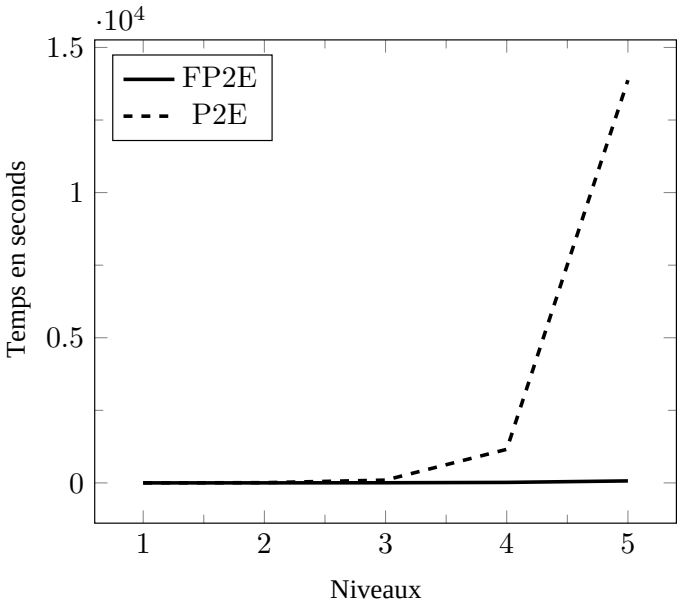


FIGURE III.11 – Comparaison du coût temporel de notre méthode avec P2E pour exprimer la même structure d'accès

De plus, étant donné que la surcharge des méthodes basées sur les attributs est linéaire par rapport au nombre d'attributs, l'augmentation du nombre d'attributs se traduit par une augmentation du coût des différentes opérations. La figure III.10 montre que le coût d'opérations de chiffrement de notre système augmente avec le nombre croissant d'attributs. La figure III.11 montre que l'augmentation du coût temporel de *FP2E* est négligeable par rapport à *P2E* qui utilise des structures d'accès binaires pour appliquer les mêmes structures d'accès.

De plus, étant donné que le nombre d'attributs dans la structure d'accès est corrélé au nombre de lignes dans la matrice *LSSS*, la transformation de la structure d'accès à porte de seuil en une structure d'accès binaire donnera une grande matrice *LSSS* qui augmentera la surcharge de stockage de données. Tout cela signifie qu'en utilisant les structures d'accès à seuil nous avons pu augmenter la performance de P2E tout en minimisant le coût de stockage lors de l'expression d'une structure d'accès compliquée.

III.3 Partage de données sécurisé avec révocation efficace et Déchiffrement externalisé pour les systèmes de stockage cloud (ESS-ODER)

Les problèmes de sécurité des données externalisées deviennent de plus en plus critiques vue fait que les données sont externalisées sous le contrôle d'un fournisseur de services cloud non-fiable. En plus, les attributs de l'utilisateur n'appartiennent, généralement, pas à une seule autorité qui contrôle les attributs des clés secrètes et génère les clés de l'utilisateur. Cela nécessite d'avoir une méthode capable de garantir la sécurité et la partageabilité des données dans un environnement multi-autoritaire sans imposer un fardeau élevé, de calcul et de stockage, sur les utilisateurs.

III.3.1 Description de la méthode

Dans un contexte multi-autoritaire, un fichier partagé peut être géré dans le domaine d'une ou plusieurs autorités, chacune de ces autorités choisit un univers d'attributs pour décrire les privilèges d'accès dans son domaine. Le propriétaire de données choisit un ensemble d'attributs, composés des attribués des différentes autorités, pour décrire la politique d'accès du fichier chiffré.

Nous utilisons des structures d'accès à porte seuil sans qu'il soit nécessaire de les convertir en formule booléenne équivalente, cette structure d'accès est ensuite convertie en matrice *LSSS* selon la méthode décrite par Liu [71] et al.

Les privilèges d'accès d'un utilisateur dans notre système sont décrits à l'aide d'un ensemble d'attributs, choisis par chaque autorité pour décrire ses droits dans son domaine, et de son identifiant unique fourni par l'autorité de certification. Une partie de la clé est ensuite générée pour chaque attribut de l'ensemble d'attributs de l'utilisateur. Les utilisateurs ne peuvent accéder aux données chiffrées que s'ils peuvent

trouver, dans leur ensemble d'attributs, un sous-ensemble pouvant satisfaire à la politique d'accès du fichier de données. La méthode proposée est caractérisée par six étapes :

1. **Inscription de l'utilisateur chez l'autorité de certification :** L'utilisateur contacte l'autorité de certification afin de récupérer son identifiant unique et ses clés publiques et privées globales.
2. **Configuration des autorités d'attributs (AA) :** Chaque autorité d'attributs choisit un univers d'attributs pour décrire les différents privilèges dans son domaine, puis génère les différentes clé publiques et privées pour chaque attribut.
3. **Chiffrement :** Au cours de cette phase, le propriétaire des données choisit un ensemble d'attributs pour construire une matrice $LSSS$, qui sera utilisé par suite pour chiffrer le fichier de données. Le propriétaire envoie ensuite le texte chiffré au CSP.
4. **Génération des clés par l'autorité des attributs :** Pour générer la clé de l'utilisateur, chaque **AA** choisit un ensemble d'attributs définissant les privilèges de l'utilisateur, puis, pour chaque attribut, l'autorité génère une clé d'attribut. Les clés d'attributs sont ensuite transférées à l'utilisateur. La consolidation des différentes clés récupérées à partir des autorités auquel l'utilisateur appartient définit sa clé dans le système.
5. **Déchiffrement :** Lors de la récupération du fichier de données à partir du CSP, l'utilisateur autorisé choisit un ensemble d'attributs dans son univers d'attributs pour satisfaire la politique d'accès, puis l'utilise pour déchiffrer les données.
6. **Révocation :** Dans la phase de révocation, nous considérons deux cas :

   Dans le premier cas, une autorité d'attributs doit révoquer un ensemble d'attributs d'un utilisateur. Elle génère donc les nouvelles clés d'attribut pour les utilisateurs non révoqués et une clé de mise à jour de texte chiffré, puis les transfère au CSP . Après cela, le serveur Cloud mettra à jour le texte chiffré et remettra les clés à leurs utilisateurs respectifs.

   Dans le second cas, le propriétaire doit mettre à jour la structure d'accès d'un fichier spécifique. Il chiffrera donc les données à l'aide de la nouvelle politique d'accès et transmettra le nouveau texte chiffré au serveur Cloud.

III.3.2 Modélisation

Modèle de système

Le modèle de système dans notre méthode comprend six entités principales (figure III.12) :

— **Fournisseur de services dans le Cloud (CSP) :** Offre un espace de stockage et une puissance de calcul sous forme de services dont les utilisateurs peuvent bénéficier. Il est également responsable de la maintenance des différents services fournis.

— **Propriétaire des données :** Utilisateur du service bénéficiant de l'espace de stockage offert par le CSP, le propriétaire des données sous-traite ses données aux serveurs de Cloud et les partage avec d'autres utilisateurs du service.

— **Utilisateurs ou utilisateurs de données :** Utilisent le service Cloud pour télécharger les données partagées par le propriétaire des données et les déchiffrent à l'aide de leurs clés secrètes.

— **Serveur Proxy** : Responsable de l'exécution de la plupart des calculs de déchiffrement. Dans notre modèle, nous supposons qu'il n'y a pas de contact entre le CSP et le serveur proxy.

— **Autorité de certification (CA)** : Un tiers de confiance qui calcule la clé secrète et la clé publique globales pour les utilisateurs. Dans notre système, l'autorité de certification est uniquement responsable des générations de clés globales des utilisateurs et de la remise des identités (ID) et des clés publiques des utilisateurs aux propriétaires et aux autorités.

— **Autorités des attributs (AA)** : Chaque autorité (Autorité d'attributs (AA)) est une entité d'attribut indépendante chargée d'émettre, de révoquer et de mettre à jour les attributs des utilisateurs en fonction de leur rôle et/ou leur identité dans leur domaine. Chaque AA peut gérer un nombre arbitraire d'attributs spécifique à elle.

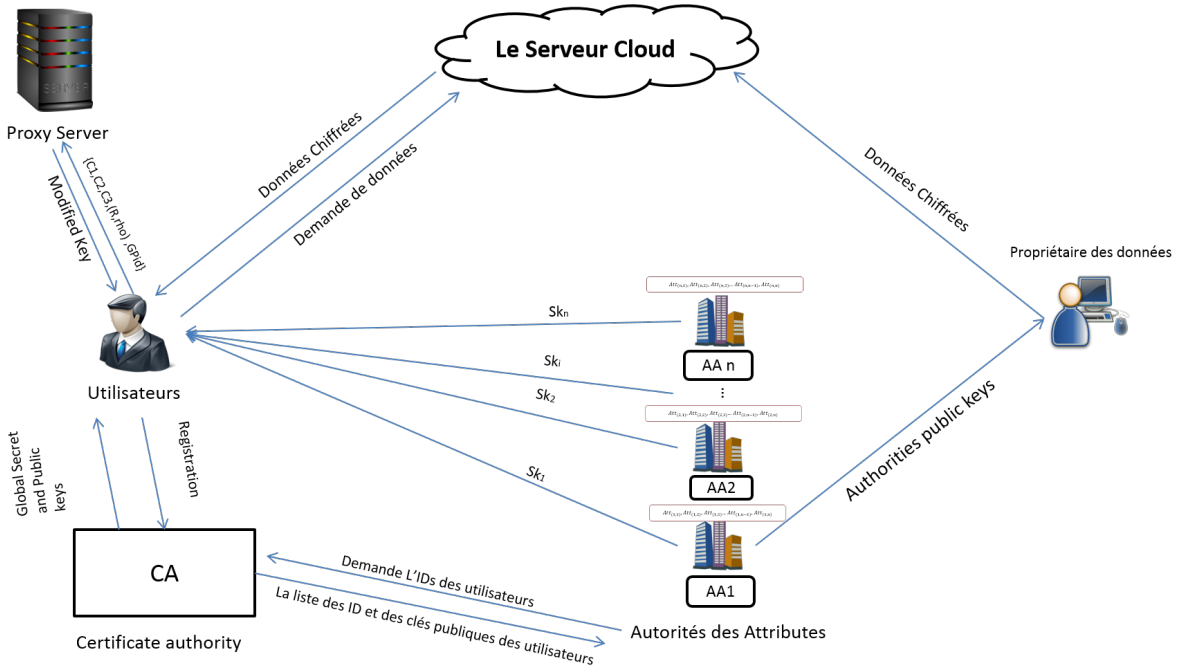


FIGURE III.12 – Modèle de système

Modèle de l'adversaire

Dans notre modèle, nous considérons un serveur Cloud semi-fiable qui suivra le protocole proposé en général ; le CSP se comporte correctement la plupart du temps, mais pour avoir certains avantages, il peut essayer de découvrir autant d'informations secrètes que possible. Ainsi, le CSP pourrait collaborer avec un certain nombre d'utilisateurs malveillants afin de récupérer le contenu des données.

Nous supposons également qu'un certain nombre d'utilisateurs malveillants pourraient essayer de combiner leurs clés afin d'accéder à des données non autorisées. D'autre Part, nous considérons le cas de collusion entre le CSP et le serveur proxy ; dans ce cas, le CSP tentera d'accéder aux données en récupérant les compositions de clés stockées par le serveur proxy. Nous considérons aussi le cas où un ou plusieurs AA sont corrompus, dans ce cas, nous devons nous assurer que même lorsqu'une collusion entre eux ou plusieurs entités du système, ils ne pourraient récupérer aucune information sur les données.

Les canaux de communication entre les différents acteurs de notre système sont supposés être sécurisés selon les protocoles de sécurité existants.

III.3.3 Détails de la méthode proposée

Soit S_A et S_U l'ensemble des autorités d'attributs et l'ensemble des utilisateurs du système respectivement. soit G et GT deux groupes multiplicatives avec le même ordre premier p , $e : G \times G \to GT$ application bilinéaire, g_1 le générateur de G et $H : \{0,1\}^* \to G$ une fonction de hachage sécurisée dans le modèle de l'oracle aléatoire.

Inscription de l'utilisateur chez l'autorité de certification

Chaque utilisateur doit s'enregistrer lui-même auprès de CA lors de l'initialisation du système. Ensuite, le CA assigne une identité d'utilisateur unique globale ID à cet utilisateur. Pour chaque utilisateur, le CA génère la clé secrète globale $G_{sk} = z_u$ et la clé publique globale $G_{pk} = g^{\frac{1}{z_u}}$, en choisissant au hasard deux nombres ID, $z_u \in \mathbb{Z}_p$. Ensuite, l'autorité de certification donne la clé publique globale G_{pk}, la clé secrète globale G_{sk} et l'identité ID à l'utilisateur.

Configuration des autorités d'attribut

Chaque autorité $k \in S_A$ exécute l'algorithme d'initialisation en utilisant $W_{A,k}$ (l'ensemble des attributs gérés par AA_k). Pour chaque attribut $i \in W_{A,k}$, le AA_k génère deux nombres aléatoires $\alpha_{i,k}, \beta_{i,k} \in \mathbb{Z}_p$, donc la clé secrète est :

$$Sk_k = \{\alpha_{i,k}, \beta_{i,k}, i \in W_{A,k}\} \tag{III.9}$$

Puis publie la clé publique :

$$Pk_k = \{g_1^{\alpha_{i,k}}, g_1^{\beta_{i,k}}, i \in W_{A,k}\} \tag{III.10}$$

Chiffrement

Pour chiffrer un message M, le propriétaire commence par définir un ensemble d'attributs I à partir des attributs appartenant aux autorités impliquées. À l'aide de ces attributs, il définit une matrice de politique d'accès R. Le propriétaire des données choisit ensuite deux vecteurs aléatoires v et w où la première entrée de v est s et la première entrée de w est 0, et calcule $\lambda = v \cdot R_x$ et $\omega = w \cdot R_x$. ensuite, pour chaque attribut x le propriétaire des données choisit un nombre aléatoire $r_x \in \mathbb{Z}_p$ et calcule :

$$\begin{aligned} C_{x,1} &= g_1^{\lambda_\rho(x)} \cdot g_1^{\alpha_\rho(x) \cdot r_x}, \\ C_{x,2} &= g^{r_x}, \\ C_{x,3} &= g_1^{\beta_{\rho(x)} \cdot r_x} \cdot g_1^{\omega_\rho(x)}. \end{aligned} \tag{III.11}$$

Le propriétaire chiffre le message M comme suit :

$M_e = M \cdot e(g_1, g_1)^s$

Enfin, le propriétaire des données envoie le fichier chiffré : $C = \{\forall x, \{C_{x,1}, C_{x,2}, C_{x,3}\}; (R, \rho), Me\}$ vers le serveur du CSP.

Génération des clés par l'autorité des attributs

Pour chaque utilisateur u dans l'ensemble des utilisateurs S_u, chaque autorité $A_k \in S_A$ choisit un ensemble d'attributs I_u à attribuer à l'utilisateur (en fonction de son rôle dans son domaine), puis récupère l'identité, la clé publique globale de l'utilisateur G_{pk} et ID à partir de CA. Enfin, elle calcule la clé secrète de l'utilisateur comme suit :

$$Sk_i = \{g_1^{\frac{\alpha_{i,k}}{z_u}} \cdot H(ID_u)^{\beta_{i,k}}\}. \tag{III.12}$$

L'utilisateur u reçoit une clé auprès de chaque autorité k à laquelle il appartient :

$Sk_{u,k} = Sk_i, \forall i \in I_u$.

Chaque $sk_{u,k}$ est ensuite chiffrée à l'aide de la clé publique de l'utilisateur et remise à l'utilisateur via le serveur de Cloud. De cette manière, seul l'utilisateur peut déchiffrer la clé à l'aide de sa clé privée.

Enfin, après avoir récupéré toutes les clés des autorités, la clé secrète de l'utilisateur devient :

$$Sk = \{G_{sk}, G_{pk}, Sk_u = \{Sk_{u,k}, \forall k \in S_A\}\}. \tag{III.13}$$

Déchiffrement

l'utilisateur récupère le fichier chiffré : $C = \{\forall x, \{C_{x,1}, C_{x,2}, C_{x,3}\}; (R, \rho), Me\}$. à partir du fournisseur de service (CSP) puis il prépare une demande de déchiffrement $CT = \{\forall x, \{C_{x,1}, C_{x,2}, C_{x,3}\}; (R, \rho), Me; Sk_u; G_{pk}\}.$, cette demande est envoyée par suite au serveur proxy CT'.

Le serveur proxy CT' reçoit la demande de déchiffrement, choisit l'élément $\Delta_x \in \mathbb{Z}q$ pour chaque attributs x tel que $\sum_x \Delta_x R_x = (1, 0, ..., 0)$ puis il calcule :

$$\begin{aligned}
MK &= \prod_x \left(\frac{e(C_{x,1}, g_1^{\frac{1}{z_u}}) \cdot e(H(ID), C_{x,3})}{e(sk_{x,u}, C_{x,2})}\right)^{\Delta_x}, \\
&= \prod_x \left(\frac{e(g_1, g_1)^{\frac{\lambda_{\rho(x)}}{z_u}} \cdot e(g_1, g_1)^{\frac{\alpha_{\rho(x)} \cdot r_x}{z_u}}}{e(g_1, g_1)^{\frac{\alpha_{\rho(x)} \cdot r_x}{z_u}}}\right., \\
&\left. \cdot \frac{e(H(ID), g_1)^{\omega_{\rho(x)}} \cdot e(H(ID), g_1)^{\beta_{\rho(x)} \cdot r_x}}{e(H(ID), g_1)^{\beta_{\rho(x)} \cdot r_x}}\right)^{\Delta_x}, \\
&= \prod_x (e(g_1, g_1)^{\frac{\lambda_{\rho(x)}}{z_u}} \cdot e(H(ID), g_1)^{\omega_{\rho(x)}})^{\Delta_x}, \\
&= e(g_1, g_1)^{\frac{\sum_x \lambda_{\rho(x)} \Delta_x}{z_u}} \cdot e(g_1, H(ID))^{\sum_x \omega_{\rho(x)} \Delta_x}, \\
&= e(g_1, g_1)^{\frac{s}{z_u}} \cdot e(H(ID), g_1)^0, \\
MK &= e(g_1, g_1)^{\frac{s}{z_u}}.
\end{aligned} \tag{III.14}$$

Le serveur Proxy renvoie la clé modifiée MK à l'utilisateur qui récupère la clé de déchiffrement en calculant :
$(MK)^{z_u} = (e(g_1, g_1)^{\frac{s}{z_u}})^{z_u} = e(g_1, g_1)^s$

Enfin, l'utilisateur utilise la clé récupère pour déchiffrer les données : $M = \frac{Me}{e(g_1, g_1)^s}$.

Processus de révocation

Révocation des attributs Pour les méthodes basées sur KP-ABE, [119], [41], La révocation des droits d'accès d'un utilisateur signifie que le propriétaire doit modifier les attributs associés à ses clés, cela nécessite de modifier les structures d'accès de tous les utilisateurs concernés. Dans notre cas, le processus de révocation d'un attribut i appartenant à une autorité k suivra trois étapes principales :

— L'autorité des attributs commence par générer une nouvelle clé secrète pour l'attribut révoqué $\alpha_{i,k}, \beta_{i,k} \in \mathbb{Z}_p$ puis met à jour ses clés publique et secrète en utilisant la nouvelle valeur.

Pour chaque utilisateur $u \in L$, avec L l'ensemble des utilisateurs non révoqués. l'autorité génère une clé mise-a-jour de l'attribut i $g_1^{\frac{\alpha_{i,k}}{z_u}} \cdot H(ID_u)^{\beta_{i,k}}$ et les renvoie à l'utilisateur $K = \{K_i \forall i \in userList\}$. Par-suite, L'autorité attributaire met à jour les fichiers chiffrée associée en générant une clé de mise à jour $CK = c\{t_1, t_2\}$, (où $t_1 = \alpha_{ancien} - \alpha_{nouveau}$ et $t_2 = \beta_{ancien} - \beta_{nouveau}$) et les envoie au serveur.

— Pour chaque fichier contenant l'attribut révoqué, le CSP calculera :

$$\begin{aligned} C_{1,new} &= \frac{C_1}{(C_2)^{t_1}}, \\ &= g_1^{\lambda_{\rho(x)}} \cdot g_1^{\frac{\alpha_{ancien} \cdot r_x}{\alpha_{ancien} - \alpha_{nouveau}} \cdot r_x}, \\ C_{1,nouveau} &= g_1^{\lambda_{\rho(x)}} \cdot g_1^{\alpha_{nouveau} \cdot r_x}. \end{aligned} \tag{III.15}$$

et

$$\begin{aligned} C_{3,new} &= \frac{C_3}{(C_2)^{t_2}}, \\ &= g_1^{\omega_{\rho(x)}} \cdot g_1^{\frac{\beta_{ancien} \cdot r_x}{\beta_{ancien} - \beta_{nouveau}} \cdot r_x}, \\ C_{3,new} &= g_1^{\omega_{\rho(x)}} \cdot g_1^{\beta_{nouveau} \cdot r_x}. \end{aligned} \tag{III.16}$$

— Enfin, à chaque demande d'accès par un utilisateur non révoqué, le CSP vérifie si l'utilisateur appartient à la liste d'utilisateurs L envoyée par l'autorité d'attribut. puis, le CSP envoie à l'utilisateur son K_i respectif.

Révocation d'accès à un fichier

Lorsque le propriétaire doit révoquer le droit d'accès d'un ensemble d'utilisateurs à un fichier de données, il choisit d'abord une nouvelle clé de chiffrement s_{new}, puis définit une nouvelle structure d'accès P et génère sa matrice $LSSS$ associée (R, ρ). Le propriétaire calcule ensuite :

$$\begin{aligned} C_{x,1} &= g_1^{\lambda_{\rho(x)}} \cdot g_1^{\alpha_{\rho(x)} \cdot r_x}, \\ C_{x,2} &= g^{r_x}, \\ C_{x,3} &= g_1^{\beta_{\rho(x)} \cdot r_x} \cdot g_1^{\omega_{\rho(x)}}. \end{aligned} \tag{III.17}$$

Finalement, le propriétaire envoie au CSP le nouveau message chiffré :

$C = \{\forall x, C_{x,1}, C_{x,2}, C_{x,3}; (R, \rho); s_{proxy} = e(g_1, g_1)^{s_{new} - s}\}$

Le CSP met ensuite à jour les informations de structure d'accès du fichier, puis re-chiffre les données comme suit :

$$\begin{aligned} Me &= Me \cdot e(g_1, g_1)^{s_{proxy}}, \\ &= M \cdot e(g_1, g_1)^{s + s_{new} - s}, \\ Me &= M \cdot e(g_1, g_1)^{s_{new}}. \end{aligned} \tag{III.18}$$

III.3.4 Exemple

Dans cet exemple, nous considérons une situation dans le secteur de la santé où un hôpital gère les données des patients. L'hôpital choisit les attributs (Cancer, Terminal, Dead Alive) parmi ses attributs et récupère les attributs (Lives in New York, American, Male, Age>20) depuis les attributs fournis par le ministère de défense (à titre d'exemple). L'hôpital procède par suite à chiffrer les données en se basant sur la structure construite (figure III.13) et les stocke dans un centre de données tiers.

Les médecins (clients de notre système) se voient attribués un ensemble d'attributs qui définissent leurs privilèges d'accès (En fonction de leur spécialité. Ancienneté...). Nous suivons deux médecins qui tentent d'accéder à un dossier médical chiffré sous la structure d'accès.

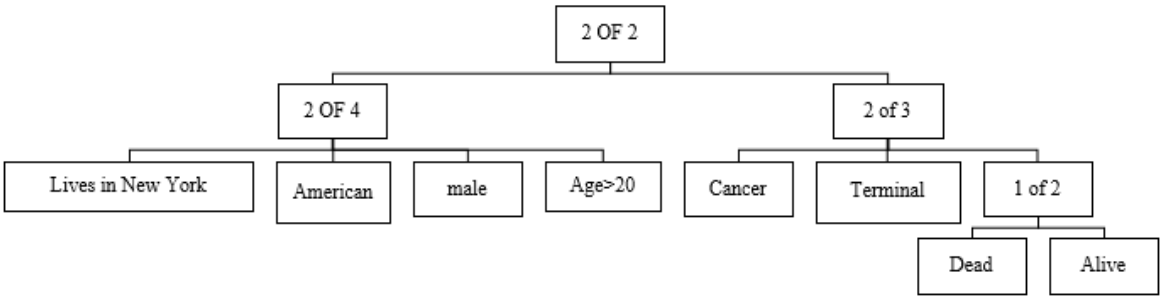


FIGURE III.13 – Structure d'accès du dossier médical

Chiffrement

le dossier médical du patient est chiffré sous une structure d'accès III.13 composé des attributs suivants : I={i1(Lives in NewYork), i2(Americain), i3(Male), i4(Age>20), i5(Cancer), i6(Terminal), i7(Dead), i8(Alive)}.

En utilisant l'algorithme dans [71], le centre de données peut générer sa matrice LSSS (équation III.19) et sa fonction de permutation (tableau III.3) à partir de la structure d'accès respective.

x	$i1$	$i2$	$i3$	$i4$	$i5$	$i6$	$i7$	$i8$
$\rho(x)$	American	Lives in New York	Male	$Age > 20$	Cancer	Terminal	Dead	Alive

TABLE III.3 – Fonction de permutation

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 2 & 0 \\ 1 & 1 & 3 & 0 \\ 1 & 1 & 4 & 0 \\ 1 & 2 & 0 & 1 \\ 1 & 2 & 0 & 2 \\ 1 & 2 & 0 & 3 \\ 1 & 2 & 0 & 3 \end{bmatrix} \tag{III.19}$$

Le centre hospitalier chiffre le dossier *Rec* avec la matrice et la fonction de permutation (R, ρ) puis envois le dossier :

$C = \{Me, C_{1,x}, C_{2,x}, C_{3,x}, x \in 1,2,3,4,5,6,7,8; (R, \rho)\}$

vers le fournisseur de service afin de le rendre accessible pour les membres du personnel.

Génération de clés :

Le centre hospitalier fournit une clé pour chaque membre du personnel décrivant ses privilèges d'accès. Dans cet exemple nous considérons deux docteurs $U1$ et $U2$ [3] :

— Docteur U1 : le centre médical accorde à ce docteur les attributs "American", "Lives in New York", "Terminal" et "Cancer" et génère sa clé en se basant sur son ensemble d'attributs $I_{U1} = i1, i2, i5, i6$ comme suit :

$$sk_{U1} = \{sk_{1,2}, sk_{2,2}, sk_{5,2}, sk_{6,2}, G_{pk} = g_1^{\frac{1}{z_{u_1}}}, G_{sk} = z_{u_1}\}$$

— Docteur U2 : le centre médical accorde ce docteur les attributs "American", "Lives in New York", "Male" et "Cancer" et génère sa clé en se basant sur son ensemble d'attributs $I_{U2} = \{i1, i2, i3, i5\}$ comme suit :

$$sk_{U2} = \{sk_{1,2}, sk_{2,2}, sk_{3,2}, sk_{5,2}, G_{pk} = g_1^{\frac{1}{z_{u_2}}}, G_{sk} = z_{u_2}\}$$

3. Veuillez noter que nous ne considérons qu'une seule autorité (le centre médical) pour des raisons de simplicité.

Déchiffrement

Nous supposons que le centre médical n'utilise que la méthode proposée pour gérer l'accès à ses dossiers médicaux, cela signifie que, suite à leurs demande, U1 et U2 reçoivent tous les deux le dossier médical :

$$C = \{Me, C_{1,x}, C_{2,x}, C_{3,x}, x \in 1,2,3,4,5,6,7,8; (R, \rho)\}$$

— U2 tente d'accéder au dossier en utilisant ses attributs. Il s'apercevra qu'il est incapable de satisfaire à la politique d'accès (figure III.13), cela signifie qu'il est incapable de récupérer la clé de déchiffrement, par conséquent son accès au dossier médical est refusé.

— U1 utilise ses attributs pour satisfaire la politique d'accès, obtenant ainsi la possibilité de récupérer l'enregistrement non chiffré. U1 récupère la clé de déchiffrement en procédant comme suit :

  — L'algorithme commence par choisir les attributs satisfaisant la politique d'accès , puis envoie la structure d'accès avec les clés nécessaires au serveur proxy :

$$\{\{C_{1,x}, C_{2,x}, C_{3,x}\} x \in 1,2,3,4,5,6,7,8; \\ (R, \rho); sk_{1,2}, sk_{2,2}, sk_{5,2}, sk_{6,2}\}.$$

  — Le serveur proxy recherche d'abord les attributs communs associés au dossier médical via la fonction de permutation ρ et obtient les attributs communs $i1, i2, i5$ et $i6$. Par conséquent, il constate que les vecteurs correspondants des attributs $i1, i2, i5, i6$ sont $(1,1,1,0)$, $(1,1,2,0)$, $(1,2,0,1)$ et $(1,2,0,2)$ respectivement dans la matrice LSSS R.
  Selon l'algorithme de déchiffrement, le serveur recherche la combinaison linéaire des lignes 1,2,5 et 6 afin d'obtenir $(1,0,0,0)$ en utilisant le coefficient de Lagrange Δ_x comme suit :

$$\begin{aligned} V_l &= (1,1,1,0) \cdot \Delta_1 + (1,1,2,0) \cdot \Delta_2 + (1,2,0,1) \cdot \Delta_5 \\ &\quad + (1,1,0,2) \cdot \Delta_6 \\ &= (1,1,1,0) \cdot 4 + (1,1,2,0) \cdot -2 + (1,2,0,1) \cdot -2 \\ &\quad + (1,2,0,2) \cdot 1 \\ &= (4,4,4,0) + (-2,-2,-4,0) \\ &\quad + (-2,-4,0,-2)(1,2,0,2) \\ &= (2,2,0,0) + (-1,-2,0,0) \\ V_l &= (1,0,0,0) \end{aligned} \tag{III.20}$$

En Fin, le serveur proxy peut utiliser l'algorithme de déchiffrement (Eq. III.14) pour calculer $MK = e(g_1, g_1)^{s/Z_u}$ puis l'envoyer à U1.

— U1 reçoit la clé modifiée du serveur proxy et récupère la clé de déchiffrement à l'aide de son G_{sk} : $e(g_1, g_1)^s = MK)^{s/Z_u{}^{Z_u}}$. Dés que $e(g_1, g_1)^s$ est calculée , le dossier médical Rec peut être récupéré comme suit : $Rec = \frac{Me}{e(g_1, g_1)^s}$.

III.3.5 Analyse de sécurité

Contrôle d'accès Granulaire : Le propriétaire des données doit pouvoir gérer facilement le contrôle d'accès à différents fichiers de données stockés sans affecter la confidentialité des données. En outre, les utilisateurs ne devraient pouvoir accéder qu'aux données pour lesquelles ils sont autorisés.

Preuve : Chaque utilisateur U_u reçoit un ensemble d'attributs décrivant les privilèges qui lui sont attribués par les autorités auxquelles il est lié. Seul un utilisateur possédant les attributs requis peut trouver $\sum \Delta_x \omega_x = 0$ et $\sum \Delta_x \lambda_x = s$ dans la plage de R_x et récupère $e(g_1, g_1)^s$ comme montré dans équation III.14.

Résistance au Collusion Une collusion entre utilisateurs ou entre utilisateurs et le CSP ne devrait pas permettre aux utilisateurs ou au le CSP d'obtenir des données pour lesquelles il n'est pas autorisé.

Preuve : La collusion désigne la coopération entre deux entités ou plus afin d'avoir accès à des données non autorisées. Dans notre analyse, nous montrons qu'ESS-ODER est totalement sécurisée contre la collusion entre les différents acteurs du système.

1. **Collusion entre utilisateurs :** Similaire à [89], nous obtenons une résistance totale à la collusion en liant le ID_u aux attributs des clés des utilisateurs. Même si plusieurs utilisateurs combinent leurs clés, ils ne pourront récupérer que $e(g_1, g_1)^{\sum \lambda_x \Delta_x} \cdot \prod_{u \in UL} e(g_1, H(ID_{\mathbf{u}}))^{\omega_x \lambda_x}$
   Puisque chaque utilisateur a un ID différent, la collusion n'autorisera pas les utilisateurs à annuler $\prod_{u \in UL} e(g_1, H(ID_u))^{\omega_x \Delta_x}$ (voir équation III.14), ils ne pourront donc pas récupérer la clé de déchiffrement.
2. **Collusion entre le serveur proxy et le fournisseur de services Cloud :** La demande de déchiffrement ne fournit aucune information sur les données au serveur proxy. Même si le serveur proxy stocke les résultats des demandes dans le but de collaborer avec le Cloud et d'accéder aux données, il ne conservera que

la clé de déchiffrement modifiée MK, ce qui est inutile sans la clé secrète globale G_{sk}.

3. **Collusion entre les utilisateurs et le serveur proxy :** Supposons que le serveur proxy conserve les informations sur les clés des utilisateurs après l'opération de déchiffrement. Lors de la collusion avec d'autres utilisateurs, le serveur proxy recevra la structure d'accès et recherchera dans la base de données un ensemble de clés qui la satisfait. Par conséquent, le serveur proxy pourra récupérer la clé de déchiffrement modifiée MK en utilisant les clés d'un utilisateur x. Toutefois, l'utilisateur malveillant et le serveur proxy ne pourront pas récupérer le déchiffrement sans la clé secrète globale G_{sk} de l'utilisateur x.

Garantie de protection de la vie privée : le CSP ne devrait pas pouvoir accéder aux informations de confidentialité des utilisateurs.

Preuve : Dans notre méthode le serveur proxy acquiert la clé de l'utilisateur lors de la demande de déchiffrement des données ce qui lui permettra récupérer la clé modifiée. Cependant, sans une connaissance préalable de la clé secrète global G_{sk}, le serveur ne pourra pas récupérer la clé de déchiffrement. Cela signifie que le serveur proxy n'aura accès à aucune information sur les données chiffrées ni sur la clé de l'utilisateur.

Sécurité en arrière et en avant : Après le processus de révocation, les utilisateurs révoqués ne devraient pas pouvoir accéder aux fichiers chiffrés à l'aide d'attributs révoqués. En plus, quand de nouveaux utilisateurs rejoignent le système, ils devraient pouvoir accéder aux données précédemment partagées, dans la mesure où ils seront capables de satisfaire les règles d'accès.

Preuve : Nous sommes en mesure de garantir la sécurité en arrière en mettant à jour les clés secrètes des utilisateurs non révoqués, ce qui empêche les utilisateurs révoqués de déchiffrer le texte chiffré auquel leur accès a été révoqué. De plus, nous sommes en mesure d'atteindre la sécurité en avant en mettant à jour le texte chiffré, qui garantit l'accès aux données précédemment stockées, par les nouveaux utilisateurs, à condition qu'ils acquièrent suffisamment d'attributs pour satisfaire la politique d'accès du fichier. D'autre part, l'utilisation du serveur de déchiffrement de proxy n'affecte pas la sécurité en avant et en arrière du système ; comme notre système est sécurisé contre la collusion, la réception des nouvelles clés ne fournira aucune nouvelle information au serveur proxy.

Sécurité sémantique

Modèle de Sécurité Nous définissons la sécurité de notre méthode dans le contexte de la sécurité sémantique, dans lequel nous donnons un texte chiffré à l'adversaire sans aucune information sur le Texte claire. L'adversaire ne devrait pas pouvoir récupérer des informations sur le texte brut tant qu'il ne possède pas tous les attributs nécessaires pour satisfaire la politique d'accès. Nous supposons également que l'adversaire ne peut corrompre les autorités que de manière statique, mais les requêtes de clés sont effectuées de manière adaptative.

Dans ce qui suit, nous définissons la sécurité d'ESS-ODER comme un jeu relevant du modèle de l'adversaire susmentionné. Dans notre modèle, nous avons deux types de menaces :

— **Menaces internes** : par le fournisseur de services Cloud «curieux», un groupe d'utilisateurs non autorisés ou des autorités d'attribut corrompues.

— **Menaces extérieures** : par des entités externes au système.

Le jeu de sécurité entre le challenger (qui joue le rôle du propriétaire des données et de toutes les autorités attributaires) et l'adversaire est défini comme suit :

— **Initialisation** :
  — L'adversaire choisit un ensemble d'autorités corrompues parmi l'ensemble des autorités $S'_A \subset S$ puis l'enverra au challenger.
  — le challenger exécutera l'algorithme de configuration afin de générer des clés publiques et privées pour chacune des autorités.
  — L'adversaire recevra la clé publique des autorités non corrompues et les clés publique et privée des autorités dans S'_A. L'adversaire choisit ensuite un identifiant et l'envoie au challenger. Ensuite, il recevra la clé secrète globale $G_{sk} = z_u$ et la clé publique globale $G_{Pk} = g^{1/z_u}$ générées par le challenger à l'aide de l'algorithme d'enregistrement de l'utilisateur.

— **Phase 1** :
  — L'adversaire effectue des requêtes clés pour le challenger en soumettant (i, ID, G_Pk) où ID est le ID de l'utilisateur et i désigne un attribut qui appartient à une bonne autorité (non corrompue).
  — Le challenger enverra les clés d'attributs calculées $g_1^{\alpha_i / Z_u} \cdot H(ID)^{\beta_i}$ à l'adversaire. Les clés demandées ne doivent pas pouvoir satisfaire à la structure d'accès spécifiée dans la phase de challenge avec les clés reçues des autorités corrompues.

— **Défi** :
  — L'adversaire choisira deux messages aléatoires M_0 et M_1 de même longueur et une matrice d'accès (R, ρ) qui sera contestée, puis les remettra au challenger.
  — Le challenger choisit une valeur aléatoire $b \in \{0, 1\}$ et chiffre le message M_b en utilisant la politique d'accès reçue.
— **Phase 2** : L'adversaire est autorisé à effectuer d'autres requêtes sur des attributs supplémentaires en utilisant le même processus que celui décrit dans la phase 1 tant que les attributs demandés ne satisfont pas à la politique d'accès (R, ρ) .
— **Déduction** : L'adversaire tentera de gagner le jeu de sécurité en devinant b'. L'adversaire ne gagne que si $b' = b$. L'avantage de l'adversaire dans ce jeu de sécurité est défini comme suit : $\epsilon = |Pr[b = b'] - 1/2|$

Lemme : ESS-ODER est sécurisée contre la corruption statique des autorités si tous les adversaires ont au plus un avantage négligeable dans le jeu de sécurité ci-dessus.

Preuve de sécurité :

Théorème 1 : Le schéma proposé est sécurisé si tous les adversaires polynomiaux [4] ne détiennent qu'un avantage négligeable dans le jeu de la sécurité.

Preuve : similaire a [30, 10, 13] nous utilisons le modèle de groupe bilinéaire générique pour nous assurer que, avec seulement un accès en boîte noire aux opérations et à la fonction de hachage H, l'adversaire ne peut pas déchiffrer le schéma proposé.

— **Initialisation** :
  — le challenger exécute l'algorithme d'installation en choisissant une clé privée de manière aléatoire $\alpha_i, \beta_i \in \mathbb{Z}_p \forall i \in W$, puis génère la clé publique : $Pk = \{g_1^{\alpha_i}, g_1^{\beta_i}, i \in W\}$ et livre g_1 et Pk à l'adversaire.
  — L'adversaire demande $H(ID)$ et G_Pk au challenger pour un certain ID.
  — Le challenger choisit deux valeurs aléatoires $h_{id}, z_{id} \in \mathbb{Z}_p$ puis calcule $H(ID) = g_1^{h_{id}}$ et $G_{pk} = g_1^{1/z_u}$.
  Enfin, le challenger livre $\{H(ID), G_{pk}, G_{ID} = z_u\}$ à l'adversaire.
— **Phase 1** : Après avoir reçu la réponse du challenger, l'adversaire adresse un certain nombre de requêtes de clés au challenger. Chaque fois que l'adversaire demande une clé pour un attribut i, le challenger calcule la clé $Sk_i = g_1^{\alpha_i / z_u} \cdot H(ID_u)^{\beta_i}$ et la remet à l'adversaire.

4. utilise un algorithme de temps polynomiale

— **Défi** : L'adversaire choisit une matrice d'accès (R, ρ) qui sera utilisée dans le défi puis la transmet au challenger.
Le challenger utilisera ensuite la matrice d'accès reçue pour générer le texte chiffré. Nous considérons un jeu modifié dans lequel, au lieu de faire la distinction entre $C_0 = M_0\ e(g_1, g_1)^s$ et $C_1 = M_1 e(g_1, g_1)^s$, l'adversaire fera la distinction entre $C_0 = e(g_1, g_1)^{s_1}$ et $C_1 = e(g_1, g_1)^{s_2}$. Comme mentionné dans [30], ce jeu modifié est justifié par l'argument hybride de [10]. Le texte chiffré sera généré comme suit :

$$C = \{\forall x, \{C_{x,1}, C_{x,2}, C_{x,3}\}; (R, \rho), e(g_1, g_1)^{s_2}\},$$

où s_2 est une valeur aléatoire dans $\mathbb{Z}_p$. La vue de l'adversaire dans le jeu modifié est distribuée de manière identique si C_0 a été défini sur $e(g_1, g_1)^{s_1}$ au lieu de $e(g_1, g_1)^{s_2}$. Cela montre que l'adversaire obtient un avantage négligeable dans le jeu modifié.

— **Phase 2** :L'adversaire peut émettre des demandes supplémentaires de la même manière que celle décrite à la phase 1.

— **Déduction** : Puisque s_2 n'apparaît que dans C_1, l'adversaire ne peut effectuer que des requêtes impliquant s_2, qui sont sous la forme $k \cdot s_2$ plus d'autres termes, où k est une constante. Si l'adversaire effectue deux requêtes q_1 et q_2, qui sont des polynômes inégaux mais qui deviennent identiques lorsque nous remplaçons s_1 par s_2, la vue de l'adversaire sera toujours différente lorsque $s_1 = s_2$. Ainsi, $q_1 - q_2 = ks_1 - ks_2$ pour certains k. Nous montrons maintenant que l'adversaire ne peut pas récupérer la clé sous la forme ks_1.
Pour une autorité d'attribut corrompue AA, l'adversaire connaît les valeurs de la clé secrète $\{\alpha_i, \beta_i\}$ d'un attribut i, qui apparaît dans le coefficient des termes $C_{(x,1)}$ et $C_{(x,3)}$. nous rappelons que $\lambda_x = R_x \cdot v$ dont v est un vecteur $v = \{s_1, v_2 \ldots, v_n\} \in \mathbb{Z}_p^n$. Chaque λ_x est une partie partagé de s_1. Pour faire une requête sous la forme ks_1, l'adversaire doit choisir une constante b_x tel que $\sum_x \lambda_x \Delta_x = ks_1$ et construit : $\sum_x b_x(\frac{\lambda_x + \alpha_{rho(x)} \cdot r_x}{z_u}) \cdot \Delta_x$.
Si un attribut $\rho(x)$ appartient à une AA corrompue, l'adversaire peut facilement récupérer $\lambda_{\rho(x)}$ puisqu'il connaît déjà les deux $\alpha_{\rho(x)}$. Si l'attribut $\rho(x)$ appartient à un bon AA, l'adversaire ne peut pas annuler $\lambda_{\rho(x)} r_x$ en effectuant une requête sur l'attribut $\rho(x)$ comme décrit dans la phase 1.

Après avoir reçu la clé pour l'attribut demandé, l'adversaire peut construire un polynôme de requête sous la forme : $b_x \Delta_x((\lambda_x + \alpha_{\rho(x)} \cdot r_x)/z_u - ((\alpha_{\rho(x)} \cdot r_x)/z_u + \beta_{\rho(x)} \cdot h_{ID} \cdot r_x))$, mais le terme $-b_x \Delta_x \beta_{\rho(x)} \cdot h_{ID} \cdot r_x$ reste.
le terme $-b_x \Delta_x \beta_{\rho(x)} \cdot h_{ID} \cdot r_x$ peut etre annulé en utilisant $b_x \Delta_x(\beta_{\rho(x)} \cdot h_{ID} \cdot r_x + h_{ID} \omega_x)$. Pourtant, cela va ajouter un terme supplémentaire $b_x h_{ID} \omega_x$. L'adversaire

ne peut annuler $b_x h_i d\omega_x$ que s'il a tous les attributs pour satisfaire la structure d'accès, cela n'est possible que si tous les attributs nécessaires au déchiffrement des données appartiennent à la AA_s corrompue. Dans ce cas, l'adversaire aura brisé le jeu de la sécurité.
Dans le cas où certains des attributs requis appartiennent à AA_s non corrompu, l'adversaire ne pourra pas récupérer ks_1 puisqu'il est masqué par le terme $b_x h_{ID} \omega_x$.
En conséquence, nous avons montré que l'adversaire ne pouvait pas construire une requête de la forme ks_1. Ainsi, pour un s_2 aléatoire, l'adversaire ne peut pas distinguer s_2 de s_1, ce qui signifie que l'adversaire peut obtenir qu'un avantage négligeable dans le jeu de la sécurité.

Par conséquent, notre système est protégé contre la corruption statique d'autorités si tous les adversaires ont tout au plus un avantage négligeable dans le jeu de sécurité ci-dessus.

Analyse de performance

Soit I_u le nombre total d'attributs d'un utilisateur, I_c le nombre total d'attributs dans le texte chiffré. Pour un attribut révoqué x, $non_{c,x}$ le nombre des utilisateurs non révoqués qui détiennent les attributs révoqués et $n_{c,x}$ le nombre de textes chiffrées contenant l'attribut révoqué.

TABLE III.4 – Comparaison Compréhensive

Méthode	Autorité	Calcule		Message de Révocation	Sécurité de Révocation		Contrôleur de la Révocation	Responsable de mis-à-jour du texte chiffré
		Chiffrement	Déchiffrement		En arrière	En avant		
DACC[89]	Multiple	$O(I_c)$	$O(I_u)$	$O(n_{c,x} \cdot n_{non,x})$	Oui	Non	Propriétaire	Propriétaire
DACMACs[117]	Multiple	$O(I_c)$	$O(1)$	$O(n_{non,x})$	Non	Non	AA	Serveur
MABE-SOD [32]	Multiple	$O(I_c)$	$O(I_u)$	$O(n_{non,x})$	Oui	Oui	AA	Serveur
ESS-ODER	Multiple	$O(I_c)$	$O(1)$	$O(n_{non,x})$	Oui	Oui	AA	Serveur

On suppose qu'on a N autorités dans le système, chaque autorité k contrôle les attributs n_k. Soit $n_{u,k}$ le nombre d'attributs de l'autorité k dans la clé de l'utilisateur u, P_2 la taille des éléments dans G_1 et Z_p, I_u le nombre total d'attributs d'un utilisateur et I_c le nombre total d'attributs dans le texte chiffré.

Dans le tableau III.4, nous remarquons que, comme MABE-SOD et DAC-MACS, ESS-ODER peut générer une surcharge de révocation sur le propriétaire des données, car la révocation des attributs est effectuée par les autorités d'attribut et l'opération de re-chiffrement est menée par le serveur. Le propriétaire est uniquement responsable

de la génération d'une clé de re-chiffrement proxy et d'une nouvelle stratégie d'accès pour les données révoquées.

Nous pouvons également constater que, contrairement à la DACC, notre système est capable d'assurer à la fois une sécurité en avant et en arrière.

L'opération de déchiffrement dans notre méthode et DAC-MACS est en grande partie sous-traitée au serveur proxy. L'utilisateur de données n'est tenu d'effectuer qu'une opération pour récupérer la clé de chiffrement (tous les calculs de déchiffrement sont sous-traités au serveur proxy).

TABLE III.5 – Comparaison de fardeau de stockage

Méthode	Propriétaire	Autorité des attributs k	Serveur	Utilisateurs
DACC	$(n_c + 2 \cdot \sum_1^N n_k) \cdot \lvert P \rvert$	$2 \cdot n_{a,k} \lvert P \rvert$	$(3I_c + 1) \lvert P \rvert$	$(n_{c,k} + \sum_{n=1}^{N_A} n_{a,k}) \cdot \lvert P \rvert$
DAC-MACS	$(3N_A + \sum_{n=1}^{N} n_k) \cdot \lvert P \rvert$	$(n_{a,k} + 3) \lvert P \rvert$	$(3I_c + 3) \lvert P \rvert$	$(3N_A + 1 + \sum_{n=1}^{N_A} n_{a,k}) \cdot \lvert P \rvert$
MABE-SOD	$(n_c + 2 \cdot \sum_1^N n_k) \cdot \lvert P \rvert$	$2 \cdot n_{a,k} \lvert P \rvert$	$(3I_c + 1) \lvert P \rvert$	$(2N_A + \sum_{n=1}^{N_A} n_{a,k}) \cdot \lvert P \rvert$
ESS-ODER	$(n_c + 2 \cdot \sum_1^N n_k) \cdot \lvert P \rvert$	$2 \cdot n_{a,k} \lvert P \rvert$	$(3I_c + 1) \lvert P \rvert$	$(2 + \sum_{n=1}^{N_A} n_{a,k}) \cdot \lvert P \rvert$

Analyse du coût de stockage Les services du Cloud sont établis sur le concept de paiement à la demande, ce qui signifie que plus le propriétaire a besoin de stockage, plus le coût du service sera élevé. Cela fait de la surcharge de stockage l'un des principaux problèmes des systèmes de contrôle d'accès aux données dans les systèmes de stockage en Cloud.

Le tableau III.5 montre qu'ESS-ODER peut atteindre le niveau de sécurité et d'efficacité souhaité sans surcharge de stockage. Les coûts généraux de ESS-ODER pour le propriétaire sont les mêmes que ceux de DACC et de MABE-SOD,

Tandis que le système DAC-MACS permet, à long terme, de réduire les coûts de stockage du propriétaire, le surcharge de stockage des utilisateurs de ESS-ODER est inférieure à celle de DAC-MACS et de DACC, car nous ne disposons que de deux composants par attribut associé. De plus, nous n'avons pas besoin de suivre les composants de texte chiffré tels que DACC, car le processus de révocation est effectué en mettant à jour les parties de texte chiffré et les clés associées à l'attribut révoqué. La surcharge de stockage de notre méthode et celle de MABE-SOD sur le serveur Cloud sont inférieures à celles de DAC-MACS, car nous ne stockons que trois éléments par attribut utilisé dans les données chiffrées.

Analyse de coût de révocation L'opération de révocation d'attributs dans les méthodes ESS-ODER, MABE-SOD et DAC-MACS est effectuée par les autorités AA. Dans

ESS-ODER, cette opération ne nécessite que la mise à jour des parties liées à l'attribut dans les clés des utilisateurs et les textes chiffrés.

TABLE III.6 – Comparaison de coûts de communication de l'opération de révocation d'attributs

Méthode	DACC	DAC-MACS	MABE-SOD	ESS-ODER
Mise à jour de clés	N/A	$n_{non,x}\|P\|$	$n_{non,x}\|P\|$	$n_{non,x}\|P\|$
Mise à jour de message	$(n_{c,x} \cdot n_{non,x} + 1)\|P\|$	$\|P\|$	$2\|P\|$	$2\|P\|$
Mise à jour des droits d'accès	$(3I_c + 1)\|P\|$	$(3I_c + 3)\|P\|$	$(3I_c + 3)\|P\|$	$(3I_c)\|P\|$

Le tableau III.6 montre que notre méthode et le DAC-MACS n'entraînent pas de surcharge de communication dans le processus de révocation. En outre, contrairement à DACC, ESS-ODER n'oblige pas les utilisateurs à conserver le composant texte chiffré associé, ce qui rend notre processus de révocation plus efficace.

Résultats Expérimentaux Dans l'évaluation expérimentale, nous utilisons des structures d'accès de différentes tailles afin de tester la rapidité des opérations de chiffrement, de génération de clés, de déchiffrement de proxy et de déchiffrement. L'ensemble du système d'expérimentation est mis en œuvre à l'aide de *PYCHARM* [1] sous Python sur une machine Ubuntu 16.04 dotée du processeur Intel Core i7-4610M. Les résultats sont la moyenne de 50 essais.

Nous avons supposé que le jeu d'attributs utilisé dans la génération de clés appartient à un AA, en considérant que même dans le cas d'un système multi-autoritaire, chaque AA est responsable de la génération des clés pour ses attributs.

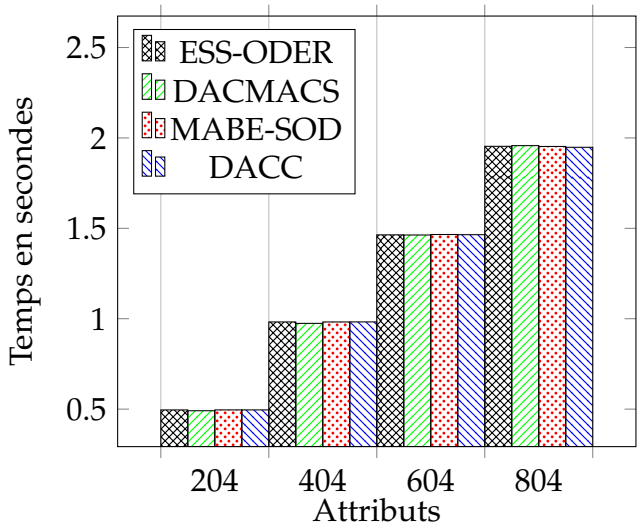


FIGURE III.14 – Comparaison du coût de calcul : Génération de clés

La figure III.14 montre que nous avons réussi à atteindre une efficacité similaire à celle de DACC et DAC-MACS en génération de clés. Étant donné que ce processus est moins utilisé que les processus de chiffrement ou de déchiffrement, l'optimisation du processus de génération de clés réduit le coût de la gestion de la clé d'attribut pour les autorités.

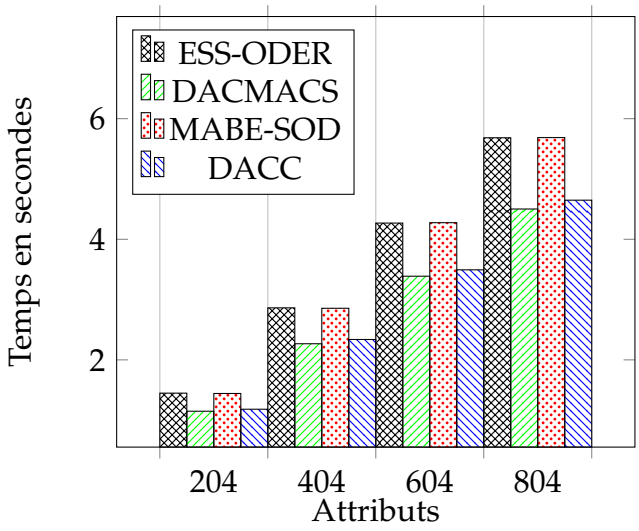


FIGURE III.15 – Comparaison du coût de calcul : Chiffrement

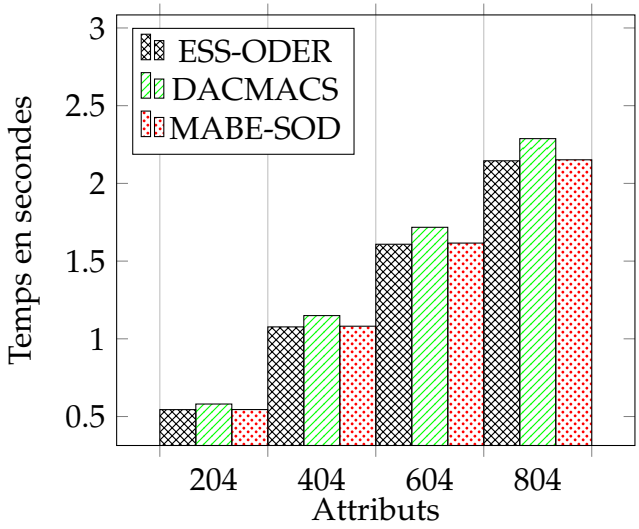


FIGURE III.16 – Comparaison du coût de calcul : Déchiffrement-Proxy

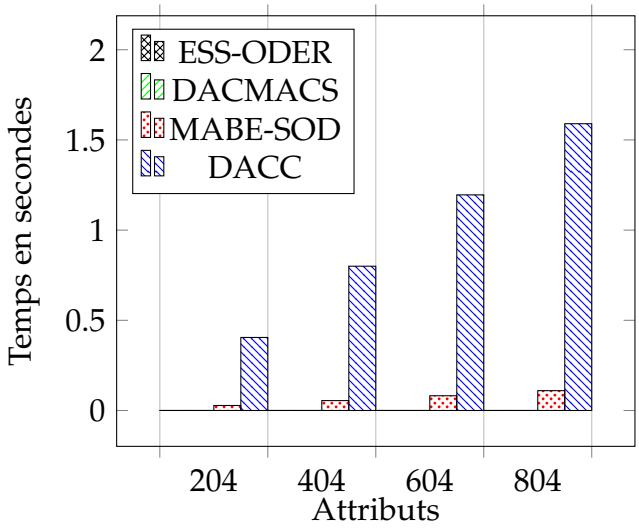


FIGURE III.17 – Comparaison du coût de calcul : Déchiffrement

Le processus de chiffrement de ESS-ODER, DACC, DAC-MACS et MABE-SOD est très coûteux (Figure III.15). Cependant, celle est négligeable dans ESS-ODER par rapport au gain d'efficacité du déchiffrement du proxy, d'autant plus que ce processus est utilisé plus souvent que le processus de chiffrement.

De plus, la surcharge du déchiffrement proxy n'influence pas directement le propriétaire des données ou les utilisateurs, elle affecte l'utilisation des ressources dans le serveur proxy. Cela se traduira par une utilisation élevée des ressources et une augmentation du prix que le propriétaire des données doit payer lorsqu'il a besoin d'un temps de calcul plus rapide. Cependant, en parallélisant l'opération de déchiffrement du proxy, la méthode peut pleinement utiliser toutes les ressources fournies par le serveur proxy, ce qui réduira le temps de calcul et permettra de gagner du temps et de l'argent.

La figure III.16 montre que nous avons réussi à réduire le temps du système de déchiffrement proxy par rapport au DAC-MACS, même sans paralléliser le calcul. De plus, en utilisant des calculs parallèles, nous avons pu réaliser une opération de déchiffrement proxy très efficace, bénéfique pour le propriétaire des données, l'utilisateur, et même le serveur proxy qui exploitera ses ressources efficacement.

L'opération de déchiffrement dans notre méthode et dans DAC-MACS a une complexité de $O(1)$, ce qui signifie que l'utilisateur de données aura un coût de calcul minimal dans l'opération de déchiffrement (Figure III.17). De plus, avec un serveur proxy puissant, l'utilisateur devra passer très peu de temps à récupérer la clé de déchiffrement des données chiffrées.

Notre système est capable d'obtenir une efficacité de déchiffrement du proxy supérieure à celle de DAC-MACS tout en n'ayant pratiquement aucune surcharge de déchiffrement. Cependant, notre chiffrement a une surcharge plus importante par rapport à DAC-MAC. Pour cette raison, nous avons décidé d'utiliser le traitement parallèle, considérant que presque tous les terminaux existants disposent de plusieurs processeurs principaux.

III.3.6 P-ESS-ODER : Parallélisation de ESS-ODER

Dans la section précédente, nous avons montré que l'opération de déchiffrement dans notre système est efficace. Les opérations de chiffrement, de génération de clés et de déchiffrement de proxy entraînent toujours une surcharge de calcul.

Puisque nous visons à fournir une méthode efficace pour les terminaux à faible puissance (téléphone mobile, voiture intelligente...), nous avons parallélisé les opérations de la méthode afin de tirer pleinement parti de la puissance de calcul des terminaux utilisés (presque tous les terminaux disposent d'un processeur multicœur.), afin d'augmenter l'efficacité des différentes opérations.
Dans notre méthode, nous nous sommes concentrés sur les opérations de génération de clés, de chiffrement et de déchiffrement proxy. L'opération de déchiffrement dans ESS-ODER nécessite que l'utilisateur effectue un calcul pour récupérer la clé de déchiffrement $(MK)^{z_u}$ [5].

Parallélisation de génération de clés

Dans l'opération de génération de clé d'ESS-ODER, l'autorité choisit un ensemble d'attributs qui décrit les privilèges de l'utilisateur. Ensuite, pour chaque attribut de l'ensemble d'attributs, l'autorité génère une clé d'attribut. Les attributs étant indépendants, cette opération peut être facilement mise en parallèle en calculant la clé de chaque attribut dans un thread différent.

5. voir la section III.14 pour plus de détails

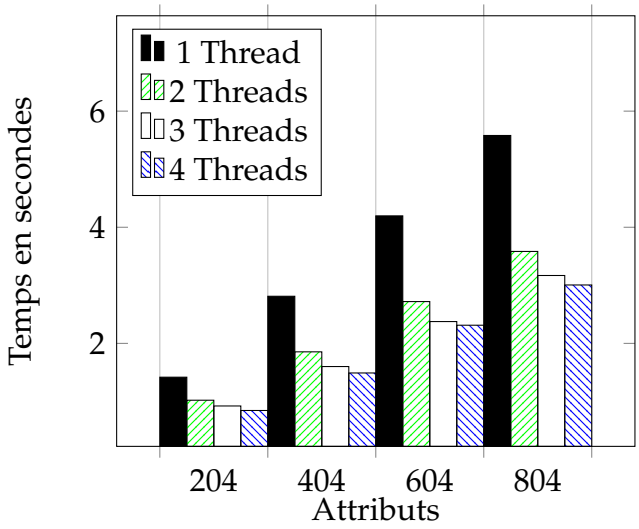


FIGURE III.18 – Test de parallélisation : Génération de clés

En utilisant cette idée, nous avons pu augmenter l'efficacité de la génération de clés, comme indiqué dans la Figure III.18. L'efficacité de la génération de clés d'ESS-ODER augmente avec le nombre de threads utilisés.

Parallélisation du chiffrement

Dans l'opération de chiffrement, à l'exception de l'analyse de la politique et de la génération de la matrice LSSS, le reste des opérations est indépendant pour chaque attribut, ce qui signifie que pour chaque attribut, le calcul peut être exécuté dans un processus séparé. Comme illustré dans les résultats expérimentaux III.19, la mise en parallèle des opérations de chiffrement réduit la charge de calcul d'ESS-ODER.

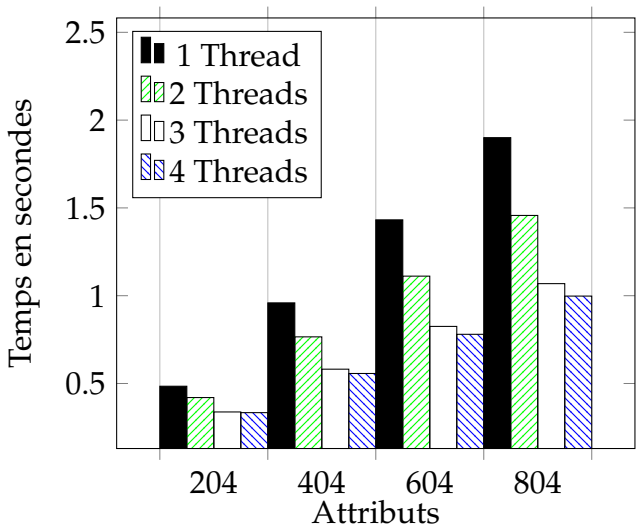


FIGURE III.19 – Test de parallélisation : Chiffrement

La Figure III.19 montre que l'utilisation d'un nombre croissant de threads réduit la charge de travail liée au chiffrement d'ESS-ODER.

Parallélisation du déchiffrement proxy

L'opération de déchiffrement proxy n'affecte directement ni le propriétaire des données ni l'utilisateur des données de notre système. Cependant, l'efficacité de notre opération de déchiffrement proxy peut influencer sur le choix et le prix du serveur proxy. Par conséquent, plus le processus de calcul est efficace, moins les exigences sont élevées pour le serveur proxy.

De plus, puisque le serveur proxy reçoit des données de calcul pour chaque attribut, il peut séparer et exécuter le déchiffrement pour chaque attribut dans un thread séparé, puis calcule la multiplication finale en utilisant les résultats de tous les threads.

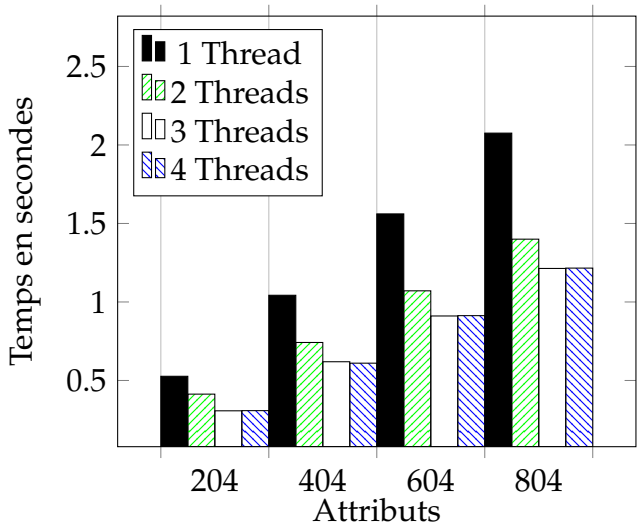


FIGURE III.20 – Test de parallélisation : Déchiffrement-Proxy

Comme le montre la figure III.20, l'efficacité de l'opération de déchiffrement proxy de notre système augmente avec le nombre de threads utilisés.

Conclusion

La confidentialité et gestion de contrôle d'accès font partie des axes indispensables de stockage distants des données, vu qu'ils permettent l'externalisation de stockage tout en maintenant la sécurité et la possibilité de partage des données. dans ce chapitre nous avons proposé deux méthodes :

— **Méthode 1 :** dans la première approche nous avons modifié une méthode existante afin d'étudier l'impact de l'utilisation des structures d'accès a seuil sur la flexibilité et la performance.

— **Méthode 2 :** dans la deuxième approche nous avons proposé une méthode permettant le partage granulaire des données, en utilisant les structures d'accès a seuil, avec un fardeau de calcul minimal sur les utilisateurs finales. Cela est réalisé en externalisant le fardeaux de calcul vers un serveur proxy sans compromettre la confidentialité des données.

Malgré que ces méthodes permettent de sécuriser les données externalisées, ces méthodes supposent que les données sont stockées dans un environnement parfait, cela veux dire que le CSP garantit que les données sont intègres sans aucune modification. En réalité la confidentialité seule ne permet pas de garantir la sécurité des données, il est primordial d'avoir une méthode de vérification l'intégrité des données externalisées sans compromettre la confidentialité ni la performance des systèmes de partages implémentées.s

Chapitre IV

Vérification de données à distance

Sommaire

Introduction

Les services Cloud sont devenus à la fois intéressants et omniprésents car ils permettent de réduire le coût et la complexité des systèmes d'informations. Bien qu'il y ait des avantages, il y a aussi des problèmes de sécurité. Les données sont stockées dans des emplacements distants sous les politiques et contrôles des fournisseurs de services Cloud. le serveur Cloud n'est pas toujours transparent par rapport à ces incidents ; il ne signalera pas nécessairement les incidents de perte

de données. De plus, une fois les données externalisées, les utilisateurs n'ont pas le contrôle physique de leurs données, ce qui signifie que les utilisateurs ont besoin de moyens pour vérifier facilement l'intégrité de leurs données.

Puisque les données sont externalisées sous le contrôle des fournisseurs de services Cloud, il sera inapproprié de permettre à l'utilisateur ou au serveur Cloud de mener le processus d'audit [1] :

— L'utilisateur ne peut pas faire confiance au fournisseur de services Cloud pour fournir un résultat d'audit impartial [107].

— La limitation des ressources des utilisateurs et la grande taille des données externalisées rend le processus d'audit très couteux.

Ces raisons justifient l'utilisation d'un service d'audit tiers. Ainsi, nous devons fournir un mécanisme d'audit des données dans lequel une autorité d'audit est en mesure d'auditer les données sans affecter leur confidentialité. En outre, les méthodes implémentées doivent prendre en considération la limitation des ressources utilisées par de nombreux utilisateurs des services de stockage.
Dans ce chapitre nous allons en premier temps discuter la problématique d'audit de données, les types de méthodes et les pré-requis d'une vérification publique, puis nous proposons une méthode de vérification permettant de satisfaire tous les pré-requis discutées avec un coût minimal sur les différents acteurs du système.

IV.1 Généralités

IV.1.1 Vérification de possession de données

La vérification de possession de données à distance permet aux utilisateurs de vérifier que le fournisseur de services Cloud stocke les données d'origine sans les récupérer. Ceci est possible en utilisant une preuve probabiliste où l'utilisateur vérifie l'intégrité d'un ensemble de blocs de données choisis au hasard.

En 2007, ATENIESE et al. [4] ont proposé un paradigme de possession de données prouvable [2] (PDP) où les utilisateurs peuvent vérifier l'intégrité des données sans récupérer l'intégralité du fichier. Selon l'exemple donné par ATENIESE et al. [4], en utilisant seulement 460 blocs de données choisis au hasard, le vérificateur peut détecter jusqu'à 1% de modification avec une probabilité supérieure à 99%.

1. Vérification d'intégrité des données
2. Provable data possession

À la suite des travaux pionniers d'ATENIESE, la vérification des possessions de données à distance[3] (RDPC) a attiré des recherches approfondies où de nombreux systèmes basés sur le système PKI traditionnel ont été proposés [5, 109, 34, 92, 112, 126, 115]. Cependant, ces systèmes nécessitent la gestion d'un grand nombre de certificats des utilisateurs, ce qui est gênant. En 2011, par exemple, les fournisseurs de navigateurs Web ont été contraints de révoquer tous les certificats émis par DigiNotar (une autorité de certification néerlandaise) après avoir découvert plus de 500 faux certificats.

Une solution à ce problème consiste à utiliser un générateur de clés privées centralisé (PKG). Le PKG génère une clé secrète pour chaque utilisateur correspondant à son identité, ce qui rend les clés publiques individuelles obsolètes (toutes les clés secrètes sont émises par le même PKG). Les méthodes fondées sur la cryptographie basée sur l'identité et celles à clé publique sans certificat sont un exemple de tels méthodes.

Dans la cryptographie à clé publique sans certificat, la clé privée de l'utilisateur est composée d'une valeur secrète choisie par l'utilisateur et de la clé privée partielle générée par le PKG [44, 104, 116, 46].

La cryptographie basée sur l'identité utilise l'identité du propriétaire comme clé publique ce qui élimine le besoin du certificat. WANG et al [110] ont proposé un modèle de sécurité pour la possession de données prouvables basée sur l'identification en plus d'une méthode concrète basée sur l'identité. WANG [108] a également présenté une méthode basé sur l'identité pour le stockage Cloud. Récemment, divers méthodes basées sur l'ID [**yu2016Cloud**, 68, 124, 108] ont été également proposés.

WANG et al. [110] a proposé un système d'audit de données à distance sécurisé qui permet aux utilisateurs de vérifier à distance l'intégrité de leurs données. Cependant, le processus de vérification du système proposé a un coût de calcul élevé pour l'auditeur. De plus, en utilisant une preuve de sécurité légère, une implémentation sécurisée nécessitera l'utilisation de clés de grande taille.

ZANG et al [123] ont proposé un système d'audit sécurisé avec une opération de vérification efficace en utilisant principalement des opérations à faible coût de calcul.

HE et al. [45] ont prouvé que le CSP peut passer le processus d'audit sans avoir le bloc de données audité, donc leur méthode n'est pas sécurisée contre les attaques contrefaites.

3. Remote data possession checking

IV.1.2 Vérification distante des données

Généralement, les systèmes d'audit de données peuvent être catégorisés sous l'une des deux catégories suivantes [118, 100] :

a. **Vérification par propriétaire (Data Owner auditing) :** Dans ce type de vérification le propriétaire des données s'occupe de l'audit.

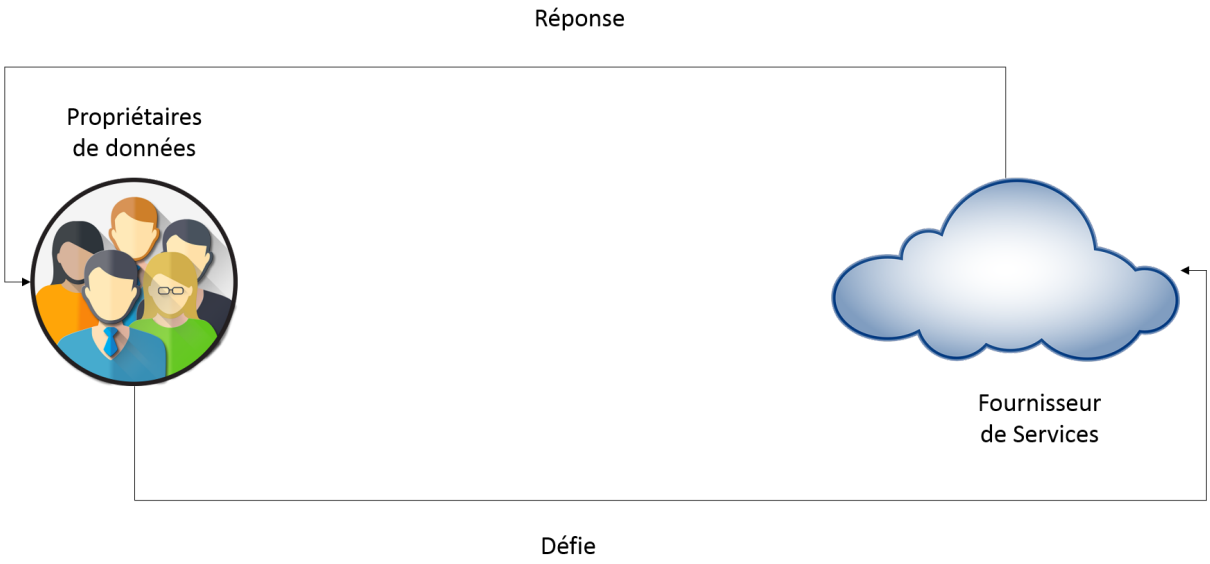


FIGURE IV.1 – Audit des données par le propriétaire

Le propriétaire des données dans ce type de méthode est capable d'auditer ses données en utilisant l'une des méthodes de vérification probabiliste (par exemple la méthode proposée par ATENIESE [4]). Par-contre, avec l'augmentation de taille des données externalisées, cette opération deviendra de plus en plus coûteuse pour un utilisateur avec des ressources limitées.

b. **vérification publique (Third party auditing) :** Dans ce type de système, le propriétaire des données délègue la vérification des données à un auditeurs tiers qui va s'occuper de la tâche de l'audit et notifier le propriétaire des résultats.

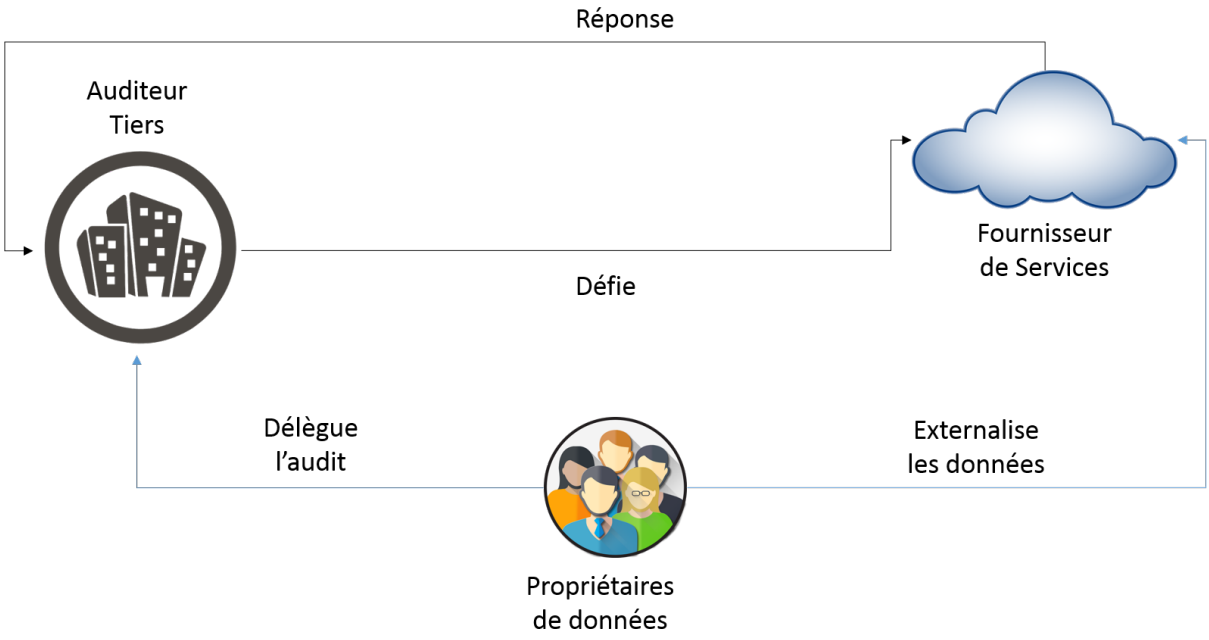


FIGURE IV.2 – Vérification publique

Malgré que La délégation d'audit des données élimine le coût de vérification pour les utilisateur. ce type d'approche nécessite d'avoir un processus sécurisé afin de garantir la confidentialité des données contre l'auditeur tiers.

IV.1.3 Techniques d'audit de données

L'audit de données (vérification d'intégrité) est le processus qui sert à détecter toute modification involontaire des données résultante de l'opération de stockage, traitement, défaillance matérielle inattendue et d'erreur humaine. Cette vérification peut être effectuée de plusieurs manières différentes :

Code d'authentification de message (MAC) [4] Un code d'authentification de message (MAC) maintient l'intégrité du message, valide l'identité de l'expéditeur et fournit la non-répudiation de l'origine. Les codes MAC sont générés par des fonctions de hachage, qui contiennent une valeur de hachage et le message à authentifier. Le récepteur utilise une clé de sécurité, connue du récepteur et de l'émetteur, pour générer le MAC.

Il est très simple d'utiliser les MAC dans l'audit des données stockées dans un service de stockage : l'utilisateur génére les codes MAC des blocs de données avant de les externaliser. les codes MAC seront envoyé (auditeur) avec la clé de génération correspondante.

4. Message authentication code

Lors de l'audit, l'auditeur récupére le bloc de données et génère son MAC. Ensuite il compare le MAC généré avec celui reçu de la part du propriétaire pour vérifier l'intégrité des blocs.

Cependant, ce type d'approche [12, 80, 69] nécessite l'envois des blocs de données au TPA pour effectuer la vérification[118] ce-qui impacte la confidentialité des données. Pour empêcher le TPA d'accéder aux données, le propriétaire (oû un utilisateur avec qui les données sont partagées) doit lui même effectuer la vérification de données, dans un tel scénario on est plus dans le cas de vérification publique.

Preuves de Possession de données (PDP) [5] Ce type de méthodes offre des garanties probabilistes. La garantie probabiliste utilise une technique d'échantillonnage qui signifie que le serveur génère la preuve de possession de données en accédant à un ensemble aléatoire de blocs.

Ce protocole permet à l'utilisateur qui souhaite externaliser son fichier sur un serveur de stockage semi-fiable [6] de vérifier si le serveur possède le fichier. L'utilisateur pré-traite son fichier pour générer des méta-données permettant la vérification. Ensuite, il envoie le fichier avec les méta-données au serveur. Pour vérifier les données, l'utilisateur demande au serveur de calculer une preuve pour un ensemble de blocs choisis aléatoirement. Le serveur calcule la preuve et la renvoie à l'utilisateur. Enfin, l'utilisateur vérifie la réponse sans récupérer les blocs de fichiers [4].

Une telle vérification peut être externalisée vers un auditeur tiers, mais cela nécessite une garantie de confidentialité des données.

Preuves de récupérabilité de données (POR) : [7] Dans ce type de méthode, L'utilisateur chiffre le fichier et insère des valeurs aléatoires appelées sentinelles. En outre, le code de correction d'erreur est appliqué au fichier pour le récupérer en cas de corruption limitée [8]. Il envoie ensuite le fichier au serveur.

Ultérieurement, l'utilisateur interroge le serveur en lui demandant de renvoyer des valeurs sentinelles spécifiques. Le serveur calcule une preuve et la renvoie à l'utilisateur. Enfin, l'utilisateur vérifie la réponse sans récupérer les blocs de fichiers [53]. Cependant ce type de méthode permet un nombre limité de vérification car le nombre total dépend du nombre de sentinelles.

5. Prouvable Data Possession
6. Serveur semi-fiable : qui peut exploiter les données de ses clients afin d'atteindre un gain
7. Proof of Retrievability
8. Le fichier peut être récupéré si le taux de corruption ne dépasse pas la limite spécifiée par la méthode

IV.2 Un système d'audit publique efficace et sécurisé pour le stockage dans le Cloud (SERDAS)

Nous proposons un système de vérification publique distribué, que nous abrégeons SERDAS [9], efficace et sécurisé pour le stockage dans le Cloud. Notre méthode permet à l'utilisateur d'externaliser ses données vers les serveurs Cloud tout en conservant la possibilité d'auditer les données soit par le propriétaire ou a l'aide d'un tiers auditeur (*TPA*).

IV.2.1 Pré-requis de la vérification distante

L'objectif est de permettre à l'audit publique, dans le cadre de stockage distribué (Cloud) des données, de devenir une réalité. Ainsi, la conception de l'architecture de service dans son ensemble ne doit pas seulement être cryptographiquement forte [10], mais aussi avoir un coût minimal sur les différents acteurs (CSP,TPA et propriétaire). Nous allons décrire un ensemble de propriétés souhaitables pour satisfaire un tel principe de conception [107, 110] :

— **Exactitude de stockage :** Le CSP ne doit pas être capable de valider le processus d'audit si les données sont corrompus (modifié, supprimé ...).

— **Coût d'audit minimal :** Le coût d'audit imposé au CSP ne doit pas impacter ses avantages. En plus, le coût de ce traitement chez le propriétaire de données doit être aussi faible que possible. Idéalement, Le propriétaire des données doit simplement profiter du service de stockage sans se soucier de l'exactitude de l'audit du stockage.

— **Confidentialité des données :** Le TPA doit être capable d'auditer efficacement les données sans avoir une copie locale ni de savoir le contenu des données.

— **Audit publique :** Le propriétaire des données doit être capable d'externaliser le processus d'audit vers un auditeur tiers TPA.

— **Audit par lot :** Lors de la réception de plusieurs tâches d'audit de différentes délégations de propriétaires, un TPA doit être en mesure d'effectuer un audit groupé des différents fichiers. Cette propriété pourrait essentiellement permettre l'évolutivité d'un service d'audit publique même avec un grand nombre de propriétaires de données.

9. A Secure and Efficient Remote Data Auditing Scheme for Cloud Storage
10. Ce terme "cryptographiquement fort" est souvent utilisé pour décrire un algorithme de cryptage qui a une grande résistance aux attaques.

— **Nombre d'audit illimité :** Le nombre vérifications possible doit être illimité. Cela signifie que, avec les même méta-données, l'auditeur doit être capable de produire plusieurs défis avec différentes réponses pour les même données. Le serveur ne doit pas être capable de répondre a un défi avec une réponse précédente si le même défie est émis auparavant pour les mêmes données.

IV.2.2 Description de SERDAS

Nous proposons un système sécurisé et publique de vérification distante de l'intégrité des données externalisée dans le Cloud qui est partiellement inspiré par le système de possession de données prouvable de Wang et al. [110].

La méthode SERDAS permet à l'utilisateur d'externaliser ses données vers les serveurs Cloud tout en conservant la possibilité d'auditer les données soit par lui-même, soit à l'aide d'un auditeur tiers (TPA). La contribution de notre méthode se divise en trois points principaux :

1. Nous permettons au propriétaire de données de déléguer le processus d'audit au TPA, cela est possible en partageant la clé de délégation et en utilisant la clé d'audit lors de la génération d'identités pour les blocs de données. De cette manière, le TPA est capable d'exécuter facilement le protocole d'audit avec une faible charge de calcul et de communication sans donner au CSP la capacité de forger des preuves de données.
2. Nous nous concentrons sur la réduction des coûts à la fois sur l'utilisateur et sur le TPA. Étant donné que le TPA peut gérer des opérations d'audit simultanées pour plusieurs utilisateurs, la réduction des coûts de calcul et de communication sur le TPA améliorera l'efficacité du mécanisme d'audit.
3. Nous prouvons la sécurité de notre système contre les attaques de contrefaçon et de remplacement. De plus, nous prouvons que le TPA n'est pas en mesure de récupérer les blocs de données pendant le processus d'audit.

IV.2.3 Modélisation

Modèle du Système

Notre système, illustré dans la figure IV.3 , est composé de quatre acteurs Principaux :

— **Fournisseur de services (CSP) :** Offre un espace de stockage et une puissance de calcul sous forme de services dont les utilisateurs peuvent bénéficier. Il est également responsable de la maintenance des différents services fournis.

— **Propriétaire des données :** Utilisateurs du service bénéficiant de l'espace de stockage offert par le CSP. Le propriétaire des données sous-traite le stockage de ses données aux serveurs de Cloud et leurs audit au TPA.

— **Auditeur tiers** (TPA) : Une autorité semi-fiable qui est en mesure de vérifier l'intégrité des données de l'utilisateur sur demande. Le TPA est honnête, mais curieux, ce qui signifie qu'il suivra le modèle fourni, mais il peut essayer de récupérer les données des réponses reçues.

— **Générateur privé de clés (PKG)** : Approuvé par les CSP, TPA et les utilisateurs, cette entité génère sa clé principale et l'utilise pour générer les clés privées des utilisateurs.

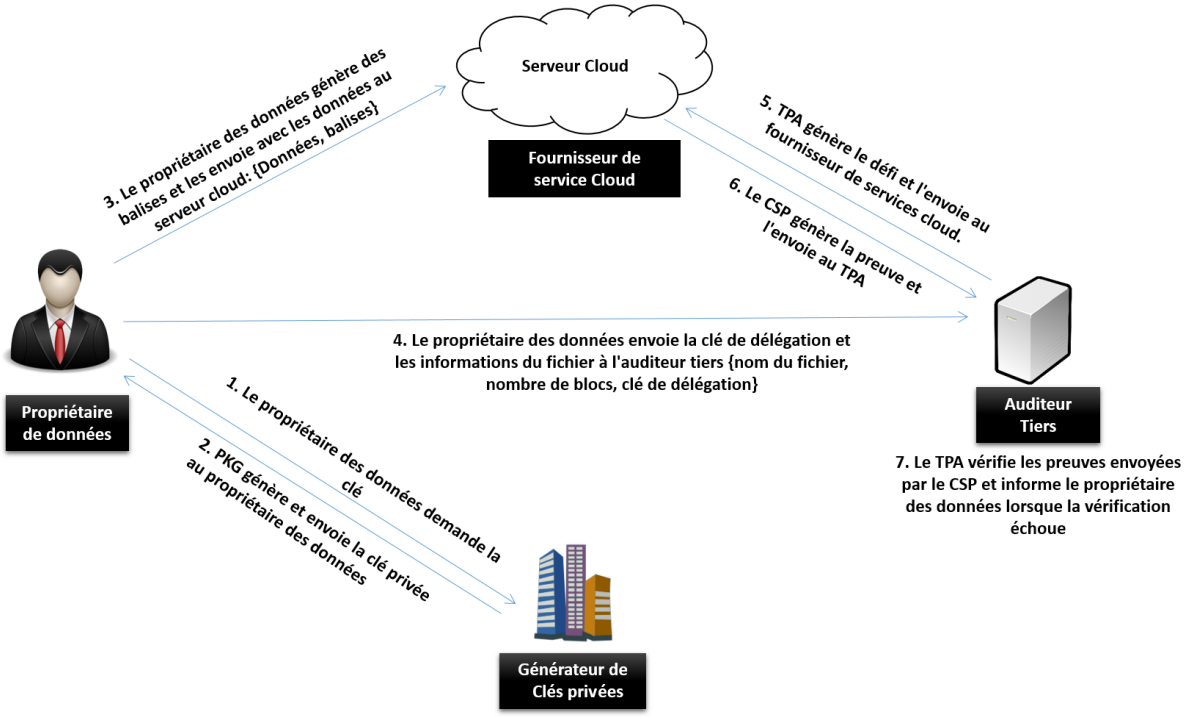


FIGURE IV.3 – Description des étapes de SERDAS

Modèle de sécurité

Nous considérons le rôle de chaque entité :

— **PKG** : est une partie de confiance qui produira honnêtement la clé publique et privée pour chaque utilisateur.

— **CSP** : est une entité semi-fiable qui peut essayer de supprimer ou de modifier les données stockées. Le CSP tentera de générer un message de vérification valide afin de convaincre le TPA et les utilisateurs qu'il stocke fidèlement les données dans ses serveurs.

— **TPA** : Est un tiers honnête, mais curieux qui exécutera fidèlement l'algorithme de vérification. Cependant, il peut essayer de récupérer les données pendant le processus de vérification.

Selon le modèle de sécurité, CSP peut lancer :

— **Attaque par forgerie** : CSP essaiera de forger un message de vérification valide $Proof$ d'un bloc de données (corrompu/supprimé) et essaiera de passer le processus de vérification à l'aide de $Proof$.

— **Attaque par remplacement** lorsque le CSP reçoit un défi pour le bloc de données corrompu m_i, le CSP essaiera de remplacer la paire de blocs de données et le tag m_i, Tag_i par la paire m_j, Tag_j d'un bloc de données non corrompu m_j. Le CSP essaiera d'utiliser ces informations afin de passer le processus de vérification.

Nous considérons également le TPA comme une partie curieuse qui peut essayer de récupérer le bloc de données audité pendant le processus d'audit.

Exigences : Pour répondre aux pré-requis décrit dans la section IV.2.1, notre méthode doit satisfaire les conditions suivantes :

Délégation d'audit : L'utilisateur doit pouvoir déléguer le processus d'audit à un tiers sans compromettre la confidentialité des données stockées. Le TPA devrait être en mesure de mener facilement le processus d'audit sans récupérer toutes les données du serveur Cloud.

Exactitude de stockage : L'utilisateur doit être en mesure de vérifier (personnellement ou avec l'aide de TPA) que ses données sont fidèlement stockées dans les serveurs Cloud.

Préservation de la confidentialité : Pendant le processus d'audit, le TPA ne devrait pas pouvoir récupérer les blocs de données audités.

Vérification par lot : L'auditeur doit pouvoir effectuer plusieurs vérifications de données en même temps.

IV.2.4 Démarche de SERDAS

Soit G_1 et G_2 les groupes multiplicatifs de même ordre premier q et $e : G_1 \times G_1 \rightarrow G_2$ l'application bilinéaire, g_1 le générateur de G_1.

Configuration Le générateur de clés privées commence par définir deux fonctions de hachage $h : \{0,1\}^* \rightarrow Z_q^*$, $H : \{0,1\}^* \rightarrow G_1$.
Le PKG choisit alors $s \in Z_q$ comme clé principale et $Y = g^s$ comme clé publique.

Extraction L'utilisateur du service de stockage envoie une demande contenant son ID au PKG pour récupérer sa clé. après la réception le demande de l'utilisateur, le PKG calcule $K = H(ID)^s$.

Le PKG envoie la clé privé $sk_{ID} = \{K\}$ à l'utilisateur, qui va vérifier l'authenticité de sa clé en s'assurant que $e(K, g) = e(Y, H(ID))$.L'utilisateur accepte sa clé si et seulement si la vérification d'authenticité est validée.

TagGen : L'utilisateur commence par choisir deux nombres $r_1, r_2 \in Z_q^*$ et puis calculer : $S_1 = H(ID)^{r_1}$ et $S_2 = H(ID)^{r_2}$.

Soit F le fichier de l'utilisateur, F est décomposé en n blocs de données $M = \{m_1, m_2, m_3 \cdots, m_n\}$. Ensuite, pour chaque bloc de données $m_i \in M$ l'utilisateur calcule :

$Tag_i = K^{m_i \cdot r_1 + h(id_i) \cdot r_2}$ avec $h(id_i) = h(S_1||ID||filename||i)$ l'identité unique du bloc m_i.

Enfin, l'utilisateur envoie le fichier $F = \{m_i, Tag_i\}, i \in [1, \cdots, n]$ au fournisseur de services CSP.

Délégation Quand l'utilisateur veut déléguer l'audit d'un fichier au TPA, il commence par calculer $V1 = e(S_1, Y), V2 = e(S_2, Y)$ et envoie $\{S_1, S_2, V1, V2\}$ au TPA.

Le TPA vérifie l'authenticité de ses clés en vérifiant si $V_1 == e(S_1, Y)$ et $V_2 == e(S_2, Y)$. Si les clés sont valides le TPA utilise S_1, S_2 comme une clé de délégation de l'utilisateur pour le fichier.

Seul l'utilisateur et le TPA ont accès a S_1, S_2, cela signifie que le CSP n'a aucune information sur la clé de délégation.

GenProof L'auditeur (propriétaire de données/TPA) demande au CSP de générer la preuve pour un ensemble d'indexes de blocs k choisis parmi l'ensemble I des indexes du fichier :

L'auditeur choisi un nombre noté $\xi \in Z_q^*$ aléatoirement et envoie le défi $Chal = \{I = \{i_1, i_2 \cdots i_k\}, \xi\}$ au CSP.

Après la réception du défi, le CSP génère la preuve comme suit :

$$\begin{aligned} \xi_i &= \xi^i \mod q \\ A &= \sum_{i \in I} \xi_i \cdot m_i, i \in I \\ B &= \prod_{i \in I} Tag_i^{\xi_i} \end{aligned} \tag{IV.1}$$

Ensuite Le CSP envoie $Poof = \{A, B\}$ à l'auditeur.

CheckProof Après la réception de la preuve auprès du CSP l'auditeur commence le processus de vérification en calculant l'équation suivante :

$$\begin{aligned} \xi_i &= \xi^i \mod q \\ h(id_i) &= h(S_1||ID||filename||i) \\ X &= \sum_{i \in I} \xi_i \cdot h(id_i) \end{aligned} \tag{IV.2}$$

L'auditeur vérifie par la suite si $e(B, g) = e(S_1^A \cdot S_2^X, Y)$ est valide, si c'est le cas l'auditeur peut confirmer que les données sont intègres.

$$\begin{aligned} e(B, g) &= e(\prod_{i \in I} Tag_i^{\xi_i}, g) \\ &= \prod_{i \in I} e(Tag_i^{\xi_i}, g) \\ &= \prod_{i \in I} e(H(ID)^{(r_1 \cdot m_i + r_2 \cdot h(id_i))\xi_i \cdot s}, g) \\ &= \prod_{i \in I} e(S_1^{m_i \xi_i s} \cdot S_2^{h(id_i)\xi_i s}, g) \\ &= \prod_{i \in I} e(S_1^{m_i \xi_i} \cdot S_2^{h(id_i)\xi_i}, Y) \\ e(B, g) &= e(S_1^A \cdot S_2^X, Y) \end{aligned} \tag{IV.3}$$

IV.2.5 Audit par lots : Méthode étendue pour un traitement multi-utilisateurs

L'auditeur tiers propose le service d'audit à un grand nombre d'utilisateurs afin de leur permettre de vérifier l'intégrité de leurs données externalisées. Cependant, l'auditeur peut recevoir plusieurs demandes d'audit de différents utilisateurs en même temps. Afin d'augmenter l'efficacité, nous généralisons notre système afin de permettre au TPA d'auditer simultanément plusieurs fichiers de données de différents utilisateurs à l'aide d'un audit par lot.

Étant donné que l'auditeur n'intervient que dans le processus d'audit, nous nous concentrons principalement sur les deux derniers processus ($GenProof$, $CheckProof$).

Soit U_l l'ensemble des utilisateurs liés à la tentative d'audit. Le processus d'audit se déroulera comme suit :

GenProof Pour chaque utilisateur de données $i \in U_l$, le TPA générera le défi $Chal_i$ comme suit :

L'auditeur choisira au hasard un ensemble d'index de blocs de données choisis au hasard I_i et un $\xi_i \in Z_q^*$, puis définira le défi $Chal_i = \{I_i, \xi_i\}$.

Le TPA enverra alors le défi : $Chal = \{Chal_i, i \in U_l\}$.

Après avoir reçu le défi, le CSP générera la preuve comme suit :

pour chaque utilisateur $i \in U_l$:

$$\begin{aligned} \xi_{i,j} &= \xi_i^j \mod q, j \in I_i \\ A_i &= \sum_{j \in I_i} \xi_{i,j} \cdot m_{i,j}, j \in I_i \\ B_i &= \prod_{j \in I_i} Tag_{i,j}^{\xi_{i,j}} \\ Proof_i &= \{A_i, B_i\} \end{aligned} \tag{IV.4}$$

Le CSP envoi $Poof = \{Proof_i, i \in U_l\}$ au TPA.

CheckProof : Après avoir reçu $Proof$ d'auprès le CSP le TPA vérifie l'intégrité des donnée par lots en calculant :

$$\prod_{i \in U_l} e(B_i, g) = \prod_{i \in U_l} e(S_{1,i}^{A_i} \cdot S_{2,i}^{X_i}, Y) \tag{IV.5}$$

IV.2.6 Analyse de sécurité

Théorème 1 : Supposons que lorsqu'un TPA envoie un défi au CSP, l'un des blocs-tag contestés $\{m_j, Tag_j\}$ est corrompu et le CSP le remplace par un bloc-tag valide $\{m_l, Tag_l\}$ et forge une réponse $\{A^*, B^*\}$. La réponse ne passera le test d'épreuve qu'avec une probabilité négligeable.

Proof : Nous supposons que l'auditeur (propriétaire/TPA) envoie un défi $Chal = \{I = \{i_1, i_2 \cdots i_k\}, \xi\}$ au CSP.

Lors de la génération de la preuve, le CSP remplacera les méta-données $\{m_j, Tag_j\}$ du bloc corrompu j par les méta-données d'un bloc l valide $\{m_l, Tag_l\}$. Le CSP procédera alors à la génération de la preuve pour l'utilisateur :

$$\begin{aligned} &\xi_i = \xi^i \mod q, i \in I \\ &A_i = \xi_i \cdot m_i, \forall i \in I, i \neq j \\ &A_j = \xi_j \cdot m_l \\ &B = Tag_l^{\xi_j} \cdot \prod_{i \in I, i \neq j} Tag_i^{\xi_i} \end{aligned} \tag{IV.6}$$

puis envoie $Poof = \{A, B\}$ à l'auditeur. Pour que la vérification soit validée par l'auditeur, l'égalité suivante doit être valide :

$$e(B, g) = e(S_1^A, S_2^X, X \cdot Y) \tag{IV.7}$$

Nous avons :

$$\begin{aligned} e(B, g) &= e(Tag_l^{\xi_j} \cdot \prod_{i \in I, i \neq j} Tag_i^{\xi_i}, g) \\ &= e(Tag_l^{\xi_j}, g) \cdot e(\prod_{i \in I, i \neq j} Tag_i^{\xi_i}, g) \\ &= e(K^{(m_l \cdot r_1 + h(id_l) \cdot r_2) \cdot \xi_j}, g) \cdot e(\prod_{i \in I, i \neq j} K^{(r_1 \cdot m_i + r_2 \cdot h(id_i)) \cdot \xi_i}, g) \\ &= e(K, g)^{(m_l \cdot r_1 + h(id_l) \cdot r_2) \cdot \xi_j} \cdot e(K, g)^{\sum_{i \in I, i \neq j} (m_i \cdot r_1 + h(id_i) \cdot r_2) \cdot \xi_i} \end{aligned} \tag{IV.8}$$

$$
\begin{aligned}
e(S_1^A \cdot S_2^X, Y) &= e(H(ID)^{(r_1)\cdot A}, g^s) \cdot e(H(ID)^{(r_2)\cdot X}, g^s) \\
&= e(K^{\xi_j \cdot m_l \cdot r_1}, g) \cdot e(K^{\sum_{i\in I, i\neq j} \xi_i \cdot m_i \cdot r_1}, g) \\
&\cdot e(K^{\xi_j \cdot h(id_j)\cdot r_2 + \sum_{i\in I, i\neq j} \xi_i \cdot r_2 \cdot h(id_i)}, g) \\
&= e(K^{\xi_j \cdot m_l \cdot r_1}, g) \cdot e(K^{\sum_{i\in I, i\neq j} \xi_i \cdot m_i \cdot r_1}, g) \\
&\cdot e(K^{\sum_{i\in I, i\neq j} \xi_i \cdot h(id_i)\cdot r_2}, g) \cdot e(K^{\xi_j \cdot h(id_j)\cdot r2}, g) \\
&= e(K^{\xi_j \cdot (r_1 \cdot m_l + r_2 \cdot h(id_j)}, g) \cdot e(K^{\sum_{i\in I, i\neq j} \xi_i \cdot r_1 \cdot m_i}, g) \\
&\cdot e(K_f^{\sum_{i\in I, i\neq j} \xi_i \cdot r_2 \cdot h(id_i)}, g) \\
&= e(K, g)^{\xi_j \cdot (m_l \cdot r_1 + h(id_j)\cdot r2)} \\
&\cdot e(K, g)^{\sum_{i\in I, i\neq j} \xi_i \cdot (m_i \cdot r_1 + h(id_i)\cdot r2)}
\end{aligned}
\tag{IV.9}
$$

Donc $e(B, g) = e(S_1^A \cdot S_2^X, Y)$
cela signifie

$$
e(K, g)^{\xi_j \cdot (r_1 \cdot m_l + r_2 \cdot h(id_j))} = e(K, g)^{(r_1 \cdot m_l + r_2 \cdot h(id_l))\cdot \xi_j} \tag{IV.10}
$$

Cela ne sera vrai que si $h(id_j) = h(id_l)$, et comme h est une fonction de hachage forte [11], la preuve générée par CSP ne peut pas passer l'audit. Par conséquent, notre protocole est sécurisé contre l'attaque par remplacement.

Théorème 2 : L'auditeur ne peut obtenir aucune information sur les données externalisées pendant le processus d'audit.

Preuve : Pendant le processus d'audit, les blocs de données sont transférés à l'auditeur sous la forme $\sum_{i\in I} \xi_i \cdot m_i$. bien que l'auditeur soit capable de calculer ξ_i, la récupération de $\xi_i \cdot m_i$ signifie que l'auditeur doit résoudre un système linéaire avec un nombre $|I|$ de variables inconnues avec seulement une équation, où $|I| > 1$. Ainsi, la probabilité que l'auditeur puisse obtenir $\xi_i \cdot m_i$ est $\frac{1}{q^{|I|-1}}$. Par conséquent, l'auditeur ne pourrait pas récupérer d'informations sur les blocs de données pendant le processus d'audit.

Théorème 3 : Le problème CDH peut être résolu s'il existe un adversaire A qui peut forger une preuve valide avec une probabilité ϵ.

Preuve : Supposant qu'un adversaire A, qui est capable de forger une preuve valide sans connaître la clé privée d'un utilisateur ID existe, alors nous pouvons construire un algorithme AL capable de résoudre le problème CDH. Dans ce jeu de sécurité, chaque ID ne peut être interrogé qu'une seule fois. Nous considérons $\{h, H\}$ comme

11. Probabilité de collusion est négligeable

des fonctions de hachage d'oracle aléatoires.
L'adversaire A peut effectuer des requêtes H, $Extract$ et $TagGen$ de manière adaptative. Soit $j \in \{1, \cdots, q_h\}$ l'index de l'identité contestée.

Setup **:** Soit G_1 et G_2 deux groupes cycliques d'ordre premier q et g le générateur de G_1. L'algorithme AL définit $Y = aP$ comme clé publique du PKG et envoie $\{G_1, G_2, g, Y, e\}$ à AL et A.

$h - oracle$ **:** Lorsque A crée une requête h avec l'identité d'un bloc id_i, il vérifie d'abord si la requête n'a pas été émise précédemment, si tel est le cas, l'algorithme choisit une valeur $h_i \in Z_q^*$ aléatoirement et l'envoie à l'adversaire, puis il stock (id_i, h_i) dans la $h - list$ qui est initialement vide.

$H - oracle$ **:** lorsque A effectue une requête H avec l'identité d'un utilisateur ID_i, l'algorithme vérifie d'abord si la requête n'a pas été émise précédemment, si c'est le cas, l'algorithme choisira un t_i aléatoire dans Z_q :

— si $i \neq j$, la valeur de $H(ID_i)$ devient $H(ID_i) = g^{t_i}$

— si $i = j$, la valeur de $H(ID_i)$ devient $H(ID_i) = g^{t_i \cdot b}$

AL renvoie $H(ID)$ à l'adversaire et stocke $(ID_i, H(ID_i), t_i)$ dans la liste $H - list$ qui est vide initialement.

Extract **:** Lorsque l'adversaire fait une requête d'extraction sur l'identité ID_i à AL, l'algorithme répondra comme suit :

— si $i \neq j$, AL récupère $(ID_i, H(ID_i))$ et ajoute $K = Y^{t_i \cdot h(ID_i)}$ à la liste des clés générées L_k et renvoie $\{K, S\}$ a A.

— si $i = j$, AL abandonne la requête.

TagGen **:** Lorsque AL reçoit une requête (ID_i, m_k) de A, l'algorithme répondra comme suit :

— si $ID_i \neq ID_j$, AL recherche (K_i, S_i) à partir de L_k puis calcule $Tag_k = K^{m_k + h(id_k)}$. AL ajoutera alors (Tag_k, m_k, ID_i) à la liste des balises générées L_T avec $S_1 = S_2 = H(ID_i)$ et renverra enfin (Tag_k) à A.

— Sinon : AL abandonne la requête.

Finalement, A répondra avec une preuve valide (B', m') sur le bloc m' sous l'identité de l'utilisateur ID' avec un avantage non négligeable ϵ. L'adversaire remportera le jeu de sécurité si les conditions suivantes sont remplies :

— (m', ID') n'ont pas été interrogés auparavant

— $ID_j = ID'$

— (B', m') est une preuve valide pour le bloc m'

Puisque (B', m') est une preuve valide, nous avons l'équation suivante :

$$\begin{aligned} e(B', g) &= e(S_1^{m'} \cdot S_2^{h(id')}, Y) \\ &= e(H(ID_j)^{m'} \cdot H(ID_j)^{h(id')}, g^a) \\ &= e(H(ID_j)^{(m'+h(id'))}, g^a) \\ &= e(g^{a \cdot b \cdot t' \cdot (m'+h(id'))}, g) \cdot \end{aligned} \tag{IV.11}$$

cela signifie :

$$B' = g^{a \cdot b \cdot t' \cdot (m'+h(id'))} \tag{IV.12}$$

on peut donc avoir :

$$g^{ab} = (B')^{\frac{1}{t' \cdot (m'+h(id'))}} \tag{IV.13}$$

L'algorithme connaît $t', m', h(id')$, donc AL peut gagner le CDH avec un avantage non négligeable ϵ qui est une contradiction en raison de la difficulté de résoudre le problème CDH. Par conséquent, notre système est sécurisé contre les attaques contrefaites.

IV.2.7 Satisfaction des pré-requis

Pré-requis : La méthode proposée est capable de répondre aux exigences décrites en IV.2.1 :

Vérifiabilité déléguée : Le processus de vérification peut être délégué par l'utilisateur des données au TPA. Le TPA est capable de vérifier l'intégrité des données de l'utilisateur à l'aide des algorithmes $ProofGen$ et $ProofVerify$.

Exactitude du stockage : **Théorème 1** et **Théorème 3** montre que le CSP ne peut passer la vérification que si tous les blocs de données sont intègres, d'où notre méthode est capable de fournir l'exactitude de stockage des données de l'utilisateur.

Préservation de la confidentialité : **Théorème 2** montre que TPA est incapable de récupérer les blocs de données audités car il faudra résoudre une équation linéaire avec I inconnus et une seule équation. Par conséquent, notre système est capable de maintenir la confidentialité des données pendant le processus d'audit.

TABLE IV.1 – Comparaison de coûts de stockage sur le CSP and TPA

Méthode	Serveur	Auditeur
Wang et al.[110]	$\|M\| + 8 * 160 * n$	$\|ID\| + \|n\| + 320$
Zhang et al. [123]	$\|M\| + 160 * n$	$\|ID\| + \|n\|$
Méthode proposé	$\|M\| + 160 * n$	$\|ID\| + \|n\| + 320$

Vérification par lots : Le TPA est capable d'effectuer des audits par lots en utilisant les algorithmes $ProofVerify$ et $ProofGen$ pour un grand nombre de vérifications.

Nombre d'audit : Au cours de chaque opération d'audit(équation IV.1), l'auditeur choisis un nombre ξ aléatoire pour construire le défie, cela signifie que pour un bloc de données m_i l'auditeur peut générer plusieurs défis en choisissant différentes valeurs ξ. Par conséquent, le nombre d'audit dans notre méthode est illimité.

IV.2.8 Analyse de stockage

Nous comparons le coût du stockage entre notre méthode, WANG et al. [110] et ZHANG et al. [123]. Nous notons que [123] et [40] ont montré que l'utilisation de la preuve de sécurité légère (loose security proof) [12] nécessitera une augmentation de α^3 pour atteindre le même niveau de sécurité qu'une réduction de sécurité stricte avec un paramètre de sécurité α. Par conséquent, afin d'obtenir le même niveau de sécurité où la taille de $|G_1|$ dans [110] est de 8 * 160 bits, à la fois notre méthode et Zhang et al. La méthode ne demandera que $|G_1|$ pour avoir la taille de 160 bits. Dans cette analyse, nous avons défini la force de sécurité à 80 bits (ce qui signifie que $|q| = 160$) et nous supposons que la taille des données est $|M|$. Le fichier de données est divisé en blocs de données n et chaque bloc de données correspond à une balise d'authentification. Dans notre analyse nous avons défini la taille des blocs de données à 160 *bits*.

Dans le tableau IV.1, nous comparons le coût de stockage de nos deux méthodes, WANG et al. [110] et ZHANG et al. à condition d'avoir le même niveau de sécurité. Comme expliqué par ZHANG et al. [123] et GOH et al. [40] ,en fournissant une preuve de sécurité légère, la méthode nécessitera un coût supplémentaire de stockage plus important selon le même niveau de sécurité. Ainsi, le coût de stockage de leur méthode est plus important par rapport à la nôtre et à ZHANG et al. ; la notre et ZHANG et al. nécessite $|M| + 160 * n$ pour stocker les données dans le Cloud. La méthode de WANG et al nécessitera $|M| + 160 * 8 * n$ pour stocker les données tout en conservant les mêmes exigences de sécurité et le même nombre de blocs. Par conséquent, notre

12. L'adversaire doit interroger un grand nombre de signatures avant de produire une valeur valide [42]

méthode est en mesure de réduire le coût du stockage des données sur le Cloud par rapport à WANG et al. selon les mêmes hypothèses de sécurité.

Bien que notre système oblige l'auditeur à conserver la clé de délégation, ce qui entraîne un léger désavantage par rapport à ZHANG et al. en termes de surcharge de stockage, cette délégation permet à la méthode proposée d'atteindre une plus grande efficacité dans les opérations de génération des tags, génération des défis et de vérification des preuves comme illustré dans les figures IV.4,IV.5,IV.6.

De plus, nous avons pu réduire le coût de vérification sur le TPA tout en maintenant la sécurité du processus, car l'identité du bloc de données contient la clé de délégation qui rend impossible pour le CSP de calculer le hachage et de forger une preuve pour les blocs de données manquants. Ainsi, ce compromis nous a permis d'atteindre une efficacité de vérification élevée sans avoir les mêmes problèmes de sécurité que ZHANG et al. (He et al. [45] montrent que le CSP dans la méthode de ZHANG et al est capable de forger avec succès un message de vérification valide sans avoir le bloc de données correspondant).

IV.2.9 Analyse de performance

Nous avons effectué une évaluation expérimentale dans laquelle nous faisons varier le nombre de balises générées et de blocs contestés afin de voir la vitesse des $TagGen$, $ProofGen$ et $ProofVerif$ dans notre méthode et [110]. L'ensemble du système d'expérience est implémenté en utilisant le framework de charme [1] avec Python sous Ubuntu 16.04 avec processeur Intel Core i7-4610M. Les résultats sont la moyenne de 200 essais.

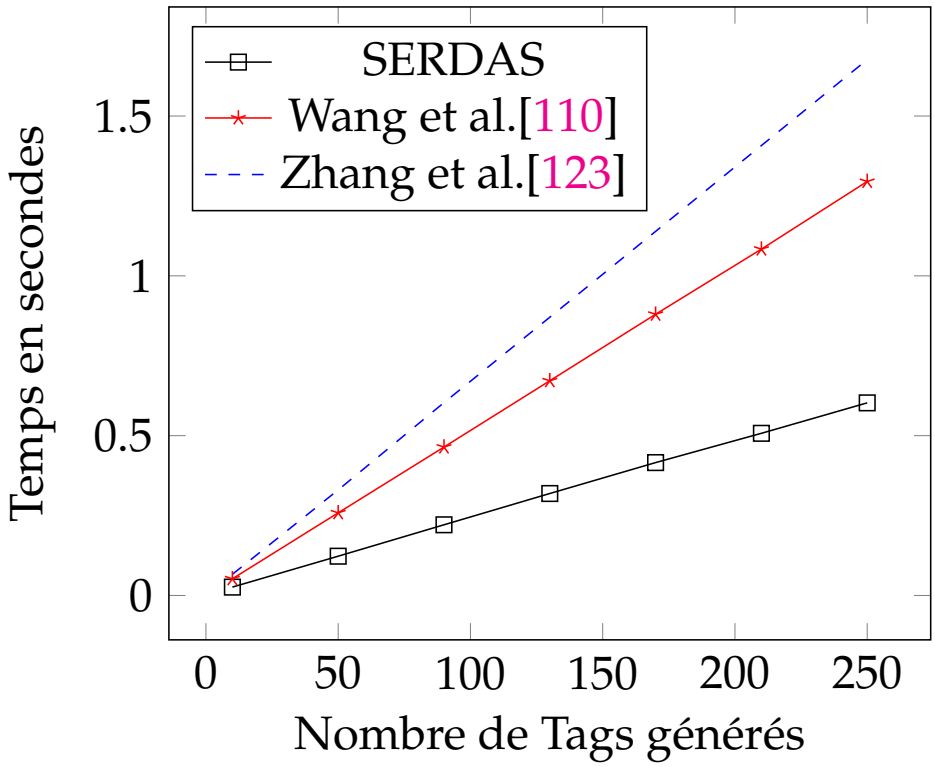


FIGURE IV.4 – Comparaison de coûts de génération des méta-données (Tags) pour l'utilisateur

Le coût de calcul pour les utilisateurs :L'utilisateur de notre système est une entité qui a accès à des terminaux de faible puissance de calcul, donc la réduction de la surcharge de calcul pour l'utilisateur est l'une des priorités de notre méthode.

La figure IV.4 montre que l'efficacité de calcul de la génération de Tag dans notre méthode est supérieure à WANG et al. [110] et [123] ZHANG et al. Ceci est dû au fait que notre méthode compte principalement la fonction de hachage $h : \{0,1\}^* \rightarrow Z_q^*$ qui, comme mentionné par he et al. [46], a un temps de calcul bien inférieur à celui de la fonction de hachage $H : \{0,1\}^* \rightarrow G_1$. Par conséquent, nous avons pu réduire considérablement les coûts de calcul des utilisateurs.

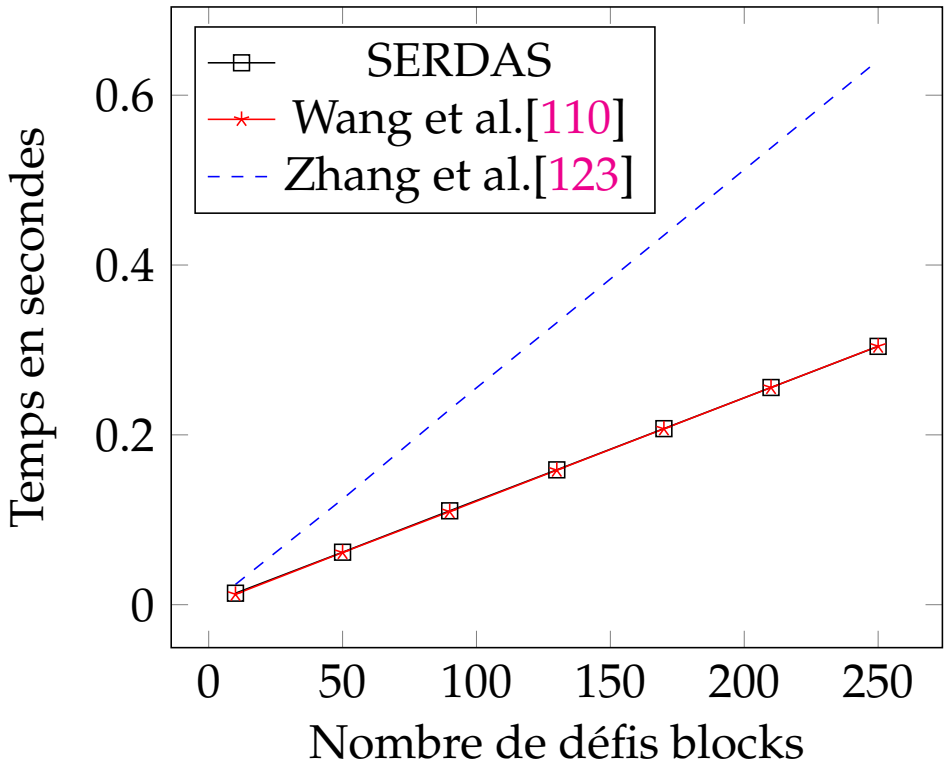


FIGURE IV.5 – Comparaison de coûts de génération de preuves sur le CSP

Coût de calcul pour le serveur Cloud : Le fournisseur de services Cloud est une entité qui a une grande puissance de calcul. Cependant, le fait que les services Cloud sont utilisés par un grand nombre d'utilisateurs, il est nécessaire de disposer d'un algorithme de génération d'épreuves efficace. Bien que le coût de la génération de preuves soit similaire à celui de [110] figure IV.5, les deux méthodes sont capables d'atteindre une efficacité de calcul plus élevée que la méthode de Zhang et al.

Coût de calcul pour l'auditeur : Similaire au fournisseur de services Cloud, le TPA n'a pas de restrictions en terme de puissance de calcul. Par conséquent, bien qu'il soit nécessaire de fournir un système de preuve efficace, nous nous sommes concentrés sur la réduction des coûts pour l'utilisateur. La figure IV.6 montre que par rapport à WANG et al. , notre coût de vérification des preuves est plus faible, ce qui augmente l'efficacité de notre système, d'autant plus que le TPA peut effectuer plusieurs opérations d'audit avec plusieurs CSP en même temps en utilisant un audit par lots. De plus, notre méthode et ZHANG et al. nécessitent un nombre réduit d'opérations. Ainsi, nous obtenons une efficacité plus élevée à la fois dans la génération des tags et dans la génération des défis tout en conservant un surcharge faible de vérification, ce qui est important en particulier en cas d'opérations de vérification par lots.

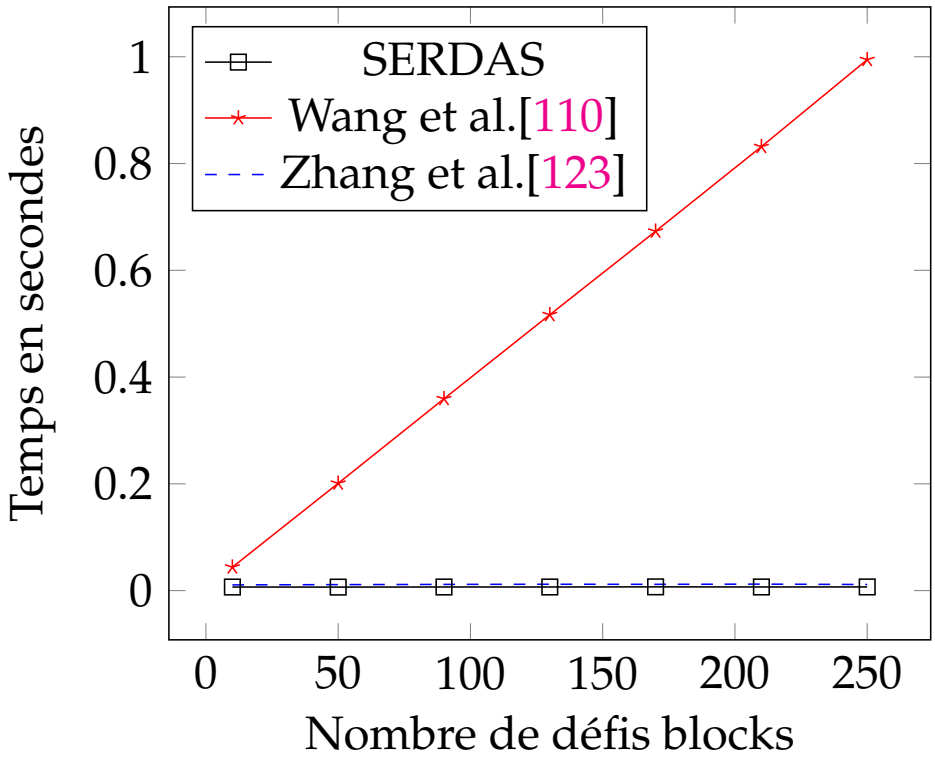


FIGURE IV.6 – Comparaison de coûts de vérification sur le TPA

IV.2.10 Analyse des coûts de communication

Soit $|G_1|$ bits et $|Z_q^*|$ la longueur d'un l'élément dans G_1 et Z_q^* respectivement.

TABLE IV.2 – Coûts de communication du processus d'audit

Wang et al. [110]	$4\|Z_q^*\| + \|G_1\|$
Zhang et al. [123]	$2\|Z_q^*\| + 2\|G_1\| + 1\|Hash\|$
La Méthode Proposé	$\|G_1\| + 2\|Z_q^*\|$

Notre méthode nécessite moins de frais de communication, car notre processus d'audit nécessite l'échange de plus petites quantités de données (tableau IV.2) par rapport aux deux méthodes de ZHANG et al. et WANG et al.

Conclusion

Avec la popularité croissante des services de stockage dans le Cloud publique, la nécessité d'un système de vérification de l'intégrité des données devient d'une importance capitale. De plus, étant donné que les services Cloud sont largement accessibles à

partir de plusieurs types d'appareils, il est important de prendre en compte l'efficacité du système.

Dans ce chapitre, nous avons proposé un système de vérification d'intégrité efficace et sécurisé. Nous montrons que notre méthode a une efficacité de calcul globale plus élevée que [110] et [123] tout en maintenant une surcharge de communication plus faible. Nous avons également montré que notre système est capable d'assurer l'intégrité des données tout en étant protégé contre les attaques de contrefaçon et de remplacement sans compromettre la confidentialité des données stockées. Dans les travaux futurs, nous étendrons le système pour inclure la récupération des données, ce qui permettra aux utilisateurs de récupérer leurs données d'origine en cas de corruption de données.

Conclusion Et Perspectives

Tout au long de cette thèse, notre objectif principal était d'améliorer et de proposer des mécanismes cryptographiques efficaces, afin d'assurer la sécurité des données dans des environnements de stockage de données Cloud. Nous nous sommes basés sur l'hypothèse qui suppose que les CSP ne sont pas totalement fiables, du coup, leurs clients sont incapables de leur faire totalement confiance. En plus, même si on suppose que le CSP est fiable, il est lui même exposé aux attaques ou pannes matérielles.

Pour conclure, ce travail a été l'occasion d'examiner un assortiment divers de concepts, modèles et technologies dans les domaines de la sécurité d'information distribués. Notre objectif était d'étudier les problèmes de sécurité des données stockées chez un fournisseur de services Cloud, tout en se concentrant sur la confidentialité des données et les problèmes de vérification d'intégrité des données à distance. Nous avons identifié trois objectifs principaux et proposé de nouvelles approches cryptographiques en réponse à ces objectifs.

Pour répondre à ces problèmes et aux objectifs qu'on a défini dans l'introduction nous avons proposé trois principales contributions :

Dans notre première contribution nous avons proposé une amélioration de la méthode P2E. Cette amélioration nous a permis de maximiser la flexibilité des structures d'accès et de minimiser l'impact de la complexité des structures d'accès sur la performance des systèmes.

Notre deuxième contribution consiste à définir une nouvelle méthode garantissant un partage dynamique, flexible et sécurisé des données tout en maintenant un coût de calcul minimal chez les clients. Nous avons proposé une approche sécurisée basée sur la méthode CP-ABE avec une délégation totale de la surcharge de chiffrement. Cette méthode permet à l'utilisateur de bénéficier des avantages des plateformes cloud sans influencer négativement les performance à cause des problèmes liés à l'utilisation d'une méthode de gestion de contrôle d'accès.

Notre troisième contribution se focalisait sur l'assurance d'intégrité des données distribuées. Nous avons proposé une méthode sécurisée de vérification publique d'intégrité permettant à l'utilisateur de déléguer d'une manière sécurisée la vérification d'intégrité de ses données à un auditeur tiers. En outre, notre méthode permet l'audit groupé des données ce qui augmente la performance et minimise le coût chez l'auditeur et le fournisseur de service.

Nos perspectives de recherche comprennent :

- Implémentation de nos contributions dans une plate-forme Cloud et réaliser des tests de performances en temps réel pour détecter les différents points à améliorer.
- Étudier l'efficacité des méthodes proposées dans un environnement Multi-Cloud.
- Faire une étude des types d'utilisation du service de stockage afin de définir les différentes approches adaptées aux besoins de groupes d'utilisateurs.

Enfin, étant donné que les caractéristiques de l'environnement Cloud et sa large utilisation, que ce soit au niveau personnel ou corporatif, la sécurité de stockage de données dans le cloud sera toujours un domaine plein de défis et d'une importance majeure malgré les solutions existantes. En outre, de nombreux nouveaux défis sont identifiés suites aux développement technologiques.

Bibliographie

[1] J. A. AKINYELE et al. « Charm : a framework for rapidly prototyping cryptosystems ». In : *Journal of Cryptographic Engineering* 3.2 (2013), p. 111-128.

[2] R AL-DAHHAN. « Efficient Ciphertext-policy Attribute Based Encryption for Cloud-Based Access Control ». Thèse de doct. Liverpool John Moores University, 2019.

[3] C. C. S. ALLIANCE. « Security Guidance for Critical Areas of Focus in Cloud Computing ». In : ().

[4] G. ATENIESE et al. « Provable data possession at untrusted stores ». In : *Proceedings of the 14th ACM conference on Computer and communications security*. Acm. 2007, p. 598-609.

[5] G. ATENIESE et al. « Remote data checking using provable data possession ». In : *ACM Transactions on Information and System Security (TISSEC)* 14.1 (2011), p. 12.

[6] E. BARKER et al. « Nist special publication 800-57 ». In : *NIST Special publication* 800.57 (2007), p. 1-142.

[7] A. BEIMEL. *Secure schemes for secret sharing and key distribution*. Technion-Israel Institute of technology, Faculty of computer science, 1996.

[8] M. BELLARE, R. CANETTI et H. KRAWCZYK. « Keying hash functions for message authentication ». In : *Annual international cryptology conference*. Springer. 1996, p. 1-15.

[9] M. BELLARE et O. GOLDREICH. « On defining proofs of knowledge ». In : *Annual International Cryptology Conference*. Springer. 1992, p. 390-420.

[10] J. BETHENCOURT, A. SAHAI et B. WATERS. « Ciphertext-policy attribute-based encryption ». In : *Security and Privacy, 2007. SP'07. IEEE Symposium on*. IEEE. 2007, p. 321-334.

[11] D BIDER et M BAUSHKE. « SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol ». In : *Request for Comments* 6668 (2012).

[12] M. BLUM et al. « Checking the correctness of memories ». In : *Algorithmica* 12.2-3 (1994), p. 225-244.

[13] D. BONEH, X. BOYEN et E.-J. GOH. « Hierarchical identity based encryption with constant size ciphertext ». In : *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2005, p. 440-456.

[14] D. BONEH et M. FRANKLIN. « Identity-based encryption from the Weil pairing ». In : *Annual international cryptology conference*. Springer. 2001, p. 213-229.

[15] D. BONEH, B. LYNN et H. SHACHAM. « Short signatures from the Weil pairing ». In : *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2001, p. 514-532.

[16] M. BORDA. *Fundamentals in information theory and coding*. Springer Science & Business Media, 2011.

[17] J. E. BORITZ. « IS practitioners' views on core concepts of information integrity ». In : *International Journal of Accounting Information Systems* 6.4 (2005), p. 260-279.

[18] M. CHASE. « Multi-authority Attribute Based Encryption ». In : *Proceedings of the 4th Conference on Theory of Cryptography* 4392 (2007), p. 515-534. ISSN : 3029743. DOI : 10.1007/978-3-540-70936-7. URL : http://www.springerlink.com/index/10.1007/978-3-540-70936-7.

[19] R. CHELLAPPA. « Intermediaries in cloud-computing : A new computing paradigm ». In : *INFORMS Annual Meeting, Dallas*. 1997, p. 26-29.

[20] X. CHEN et D. S. WONG. « Multi-Authority Ciphertext-Policy Attribute-Based Encryption with Accountability Categories and Subject Descriptors ». In : ().

[21] L. CHEUNG et C. NEWPORT. « Provably secure ciphertext policy ABE ». In : *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, p. 456-465. ISBN : 159593703X.

[22] R. CHOW et al. « Controlling data in the cloud : outsourcing computation without outsourcing control ». In : *Proceedings of the 2009 ACM workshop on Cloud computing security*. ACM. 2009, p. 85-90.

[23] A. E. C. CLOUD. « Amazon web services ». In : *Retrieved November* 9 (2011), p. 2011.

[24] L COLUMBUS. « Cloud computing adoption continues accelerating in the enterprise ». In : *Forbes, November* (2014).

[25] W. COMMONS. URL : https://commons.wikimedia.org/.

[26] D. A. COX. *Galois Theory*. T. 61. John Wiley & Sons, 2011.

[27] H. CUI et al. « An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited ». In : *Computer Networks* 133 (2018), p. 157-165. ISSN : 1389-1286. DOI : https://doi.org/10.1016/j.comnet.2018.01.034. URL : http://www.sciencedirect.com/science/article/pii/S138912861830046X.

[28] J. R. DAVID REINSEL John Gantz. « The Digitization of the World From Edge to Core ». In : *International Data Corporation (IDC)* (2018), p. 1-28.

[29] W. DIFFIE et M. HELLMAN. « New directions in cryptography ». In : *IEEE transactions on Information Theory* 22.6 (1976), p. 644-654.

[30] X. DONG et al. « Achieving an effective, scalable and privacy-preserving data sharing service in cloud computing ». In : *Computers & Security* 42 (2014), p. 151-164. DOI : 10.1016/j.cose.2013.12.002.

[31] I. EL GHOUBACH, R. B. ABBOU et F. MRABTI. « A secure and efficient remote data auditing scheme for cloud storage ». In : *Journal of King Saud University-Computer and Information Sciences* (2019).

[32] I. EL GHOUBACH, F. MRABTI et R. B. ABBOU. « Efficient secure and privacy preserving data access control scheme for multi-authority personal health record systems in cloud computing ». In : *Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on*. IEEE. 2016, p. 174-179.

[33] I. EL GHOUBACH, F. MRABTI et R. BEN ABBOU. « Achieving Secure, Flexible, Effective and Privacy Preserving Data Access Control in Cloud Computing ». In : *Recent Patents on Computer Science* 11.3 (2018), p. 196-205.

[34] C. C. ERWAY et al. « Dynamic provable data possession ». In : *ACM Transactions on Information and System Security (TISSEC)* 17.4 (2015), p. 15.

[35] X. FAN. « On remote data integrity checking of the cloud storage ». In : (2013).

[36] I. FOSTER et al. « Cloud computing and grid computing 360-degree compared ». In : *arXiv preprint arXiv :0901.0131* (2008).

[37] E. F. FOUNDATION. *Cracking DES : Secrets of encryption research, wiretap politics and chip design*. 1998.

[38] R. GELLMAN. « Privacy in the clouds : Risks to privacy and confidentiality from cloud computing ». In : *World privacy forum*. T. 23. 2009.

[39] I. E. GHOUBACH, R. B. ABBOU et F. MRABTI. « Efficient and secure data sharing with outsourced decryption and efficient revocation for cloud storage systems ». In : *International Journal of Security and Networks* 14.3 (2019), p. 133-145.

[40] E.-J. GOH et S. JARECKI. « A signature scheme as secure as the Diffie-Hellman problem ». In : *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2003, p. 401-415.

[41] V. GOYAL et al. « Attribute-based encryption for fine-grained access control of encrypted data ». In : *Proceedings of the 13th ACM conference on Computer and communications security*. Acm. Acm, 2006, p. 89-98. ISBN : 1595935185.

[42] F. GUO, Y. MU et W. SUSILO. « How to prove security of a signature with a tighter security reduction ». In : *International Conference on Provable Security*. Springer. 2009, p. 90-103.

[43] D. HANKERSON, A. J. MENEZES et S. VANSTONE. « Guide to elliptic curve cryptography ». In : *Computing Reviews* 46.1 (2005), p. 13.

[44] D. HE, B. HUANG et J. CHEN. « New certificateless short signature scheme ». In : *IET Information Security* 7.2 (2013), p. 113-117.

[45] D. HE et al. « Insecurity of an identity-based public auditing protocol for the outsourced data in cloud storage ». In : *Information Sciences* 375 (2017), p. 48-53.

[46] D. HE et al. « Privacy-preserving certificateless provable data possession scheme for big data storage on cloud ». In : *Applied Mathematics and Computation* 314 (2017), p. 31-43.

[47] I. N. HERSTEIN. *Topics in algebra*. John Wiley & Sons, 2006.

[48] L. IBRAIMI et al. « Efficient and provable secure ciphertext-policy attribute-based encryption schemes ». In : *Information Security Practice and Experience*. Springer, 2009, p. 1-12. ISBN : 3642008429.

[49] F. INSIGHTS. « Seeding the Cloud - Enterprises Set Their Strategies for Cloud ». In : (2010).

[50] M. IORGA et A. KARMEL. *Cloud Computing Security : Foundations and Challenges-Chapter 14 : Cloud Computing Security Essentials and Architecture*. Rapp. tech. 2016.

[51] P. T. JAEGER, J. LIN et J. M. GRIMES. « Cloud computing and information policy : Computing in a policy cloud ? » In : *Journal of Information Technology & Politics* 5.3 (2008), p. 269-283.

[52] Y. JIANG et al. « Ciphertext-policy attribute-based encryption supporting access policy update and its extension with preserved attributes ». In : *International Journal of Information Security* (2017), p. 1-16.

[53] A. JUELS et B. S. KALISKI JR. « PORs : Proofs of retrievability for large files ». In : *Proceedings of the 14th ACM conference on Computer and communications security*. Acm. 2007, p. 584-597.

[54] N. KAANICHE. « Cloud data storage security based on cryptographic mechanisms.(La sécurité des données stockées dans un environnement cloud, basée sur des mécanismes cryptographiques). » Thèse de doct. Telecom & Management SudParis, évry, Essonne, France, 2014.

[55] B. KALISKI et J. STADDON. *RFC2437 : PKCS# 1 : RSA Cryptography Specifications Version 2.0*. 1998.

[56] S. KAMARA et K. LAUTER. « Cryptographic cloud storage ». In : *International Conference on Financial Cryptography and Data Security*. Springer. Springer, 2010, p. 136-149. ISBN : 364214991X.

[57] G. H. KIM et E. H. SPAFFORD. « The design and implementation of tripwire : A file system integrity checker ». In : *Proceedings of the 2nd ACM Conference on Computer and Communications Security*. ACM. 1994, p. 18-29.

[58] J. KIM et al. « On the security of HMAC and NMAC based on HAVAL, MD4, MD5, SHA-0 and SHA-1 ». In : *International Conference on Security and Cryptography for Networks*. Springer. 2006, p. 242-256.

[59] N. KOBLITZ. « Modular Forms ». In : *Introduction to Elliptic Curves and Modular Forms*. Springer, 1984, p. 98-175.

[60] G. KULKARNI et al. « Cloud security challenges ». In : *2012 7th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*. IEEE. 2012, p. 88-91.

[61] N. S. KUMAR, G. R. LAKSHMI et B BALAMURUGAN. « Enhanced attribute based encryption for cloud computing ». In : *Procedia Computer Science* 46 (2015), p. 689-696.

[62] J. LAI, R. H. DENG et Y. LI. « Expressive CP-ABE with partially hidden access structures ». In : (2012).

[63] D. W. LEUNG. « Quantum vernam cipher ». In : *arXiv preprint quant-ph/0012077* (2000).

[64] A. LEWKO et B. WATERS. « Decentralizing attribute-based encryption ». In : *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2011, p. 568-588.

[65] J. LI et al. « Fine-grained access control system based on outsourced attribute-based encryption ». In : *European Symposium on Research in Computer Security*. Springer. 2013, p. 592-609.

[66] J. LI et al. « Multi-authority ciphertext-policy attribute-based encryption with accountability ». In : *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM. 2011, p. 386-390.

[67] L. LI et al. « "P-CP-ABE : Parallelizing Ciphertext-Policy Attribute-Based Encryption for clouds" ». In : *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference on*. IEEE. 2016, p. 575-580.

[68] Y. LI et al. « Fuzzy identity-based data integrity auditing for reliable cloud storage systems ». In : *IEEE Transactions on Dependable and Secure Computing* (2017).

[69] M. LILLIBRIDGE et al. « A cooperative internet backup scheme ». In : *Proceedings of the annual conference on USENIX Annual Technical Conference*. USENIX Association. 2003, p. 3-3.

[70] C.-W. LIU et al. « A Survey of Attribute-based Access Control with User Revocation in Cloud Data Storage. » In : *IJ Network Security* 18.5 (2016), p. 900-916.

[71] Z. LIU et Z. CAO. « On Efficiently Transferring the Linear Secret-Sharing Scheme Matrix in Ciphertext-Policy Attribute-Based Encryption ». In : *IACR Cryptology ePrint Archive* 2010 (2010), p. 374.

[72] Z. LIU, Z. CAO et D. S. WONG. « Efficient generation of linear secret sharing scheme matrices from threshold access trees ». In : *Cryptology ePrint Archive : Listing* (2010).

[73] D. LUCIANO et G. PRICHETT. « Cryptology : From caesar ciphers to public-key cryptosystems ». In : *The College Mathematics Journal* 18.1 (1987), p. 2-17.

[74] T. MATHER, S. KUMARASWAMY et S. LATIF. *Cloud security and privacy : an enterprise perspective on risks and compliance*. " O'Reilly Media, Inc.", 2009.

[75] P. MELL, T. GRANCE et al. « The NIST definition of cloud computing ». In : (2011).

[76] V. S. MILLER. « Use of elliptic curves in cryptography ». In : *Conference on the theory and application of cryptographic techniques*. Springer. 1985, p. 417-426.

[77] J. S. MILNE. *Group theory*. T. 3. Citeseer, 2003.

[78] R. A. MOLLIN. *Codes : the guide to secrecy from ancient to modern times*. Chapman et Hall/CRC, 2005.

[79] S. MULLER, S. KATZENBEISSER et C. ECKERT. « on Multi-Authority Ciphertext-Policy Attribute-Based Encryption ». In : *Bulletin of the Korean Mathematical Society* 46.4 (2009), p. 803-819. ISSN : 1015-8634. DOI : 10.4134/BKMS.2009.46.4.803. URL : http://koreascience.or.kr/journal/view.jsp?kj=E1BMAX{\&}py=2009{\&}vnc=v46n4{\&}sp=803.

[80] M. NAOR et G. N. ROTHBLUM. « The complexity of online memory checking ». In : *Journal of the ACM (JACM)* 56.1 (2009), p. 2.

[81] P. C. P. B. NARESH KUMAR SEHGAL. *Cloud Computing*. 1st ed. Springer International Publishing, 2018. ISBN : 978-3-319-77838-9,978-3-319-77839-6.

[82] S. PEARSON et A. CHARLESWORTH. « Accountability as a way forward for privacy protection in the cloud ». In : *IEEE international conference on cloud computing*. Springer. 2009, p. 131-144.

[83] A. POPOV. « RFC 7465 : Prohibiting RC4 cipher suites ». In : *Internet Engineering Task Force (IETF)* (2015).

[84] H. QIAN et al. « Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation ». In : *International Journal of Information Security* 14.6 (2015), p. 487-497.

[85] M. R. RAHIMI et al. « Mobile cloud computing : A survey, state of art and future directions ». In : *Mobile Networks and Applications* 19.2 (2014), p. 133-143.

[86] V RAJARAMAN. « Cloud computing ». In : *Resonance* 19.3 (2014), p. 242-258.

[87] K. RANNENBERG, D. ROYER et A. DEUKER. *The future of identity in the information society : Challenges and opportunities*. Springer Science & Business Media, 2009.

[88] V. RIJMEN et J. DAEMEN. « Advanced encryption standard ». In : *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology* (2001), p. 19-22.

[89] S. RUJ, A. NAYAK et I. STOJMENOVIC. « DACC : Distributed access control in clouds ». In : *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*. IEEE. 2011, p. 91-98.

[90] A. SAHAI et B. WATERS. « Fuzzy identity-based encryption ». In : *Advances in Cryptology–EUROCRYPT 2005*. Springer, 2005, p. 457-473. ISBN : 3540259104.

[91] A. SALOMAA. *Public-key cryptography*. Springer Science & Business Media, 2013.

[92] F. SEBÉ et al. « Efficient remote data possession checking in critical information infrastructures ». In : *IEEE Transactions on Knowledge and Data Engineering* 20.8 (2008), p. 1034-1038.

[93] H. SHACHAM et B. WATERS. « Compact proofs of retrievability ». In : *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2008, p. 90-107.

[94] M. A. SHAH et al. « Auditing to Keep Online Storage Services Honest. » In : *HotOS*. 2007.

[95] A. SHAMIR. « Identity-based cryptosystems and signature schemes ». In : *Workshop on the theory and application of cryptographic techniques*. Springer. 1984, p. 47-53.

[96] Y. SHI et al. « Directly revocable key-policy attribute-based encryption with verifiable ciphertext delegation ». In : *Information Sciences* 295 (2015), p. 221-231.

[97] M. SHINDE et H. HINGOLIWALA. « Implement MA-CPABE with Multi Central Authority for Personal Health Record in Cloud Computing ». In : *International Journal of Advance Research in Computer Science and Management Studies* (2015).

[98] M. V. SHINDE et H. HINGOLIWALA. « Secure Cloud Storage using Multi Attribute authority with Multi Central authority ». In : *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)* 3.4 (2015), p. 1797-1801.

[99] S. SINGH. *The Black Chamber : Hints and Tips*. 2012.

[100] M. SOOKHAK et al. « A review on remote data auditing in single cloud server : Taxonomy and open issues ». In : *Journal of Network and Computer Applications* 43 (2014), p. 121-141.

[101] W. STALLINGS. *Cryptography and network security : principles and practice*. Pearson Upper Saddle River, 2017.

[102] D. E. STANDARD. « Data encryption standard ». In : *Federal Information Processing Standards Publication* (1999).

[103] S. SUBASHINI et V. KAVITHA. « A survey on security issues in service delivery models of cloud computing ». In : *Journal of network and computer applications* 34.1 (2011), p. 1-11.

[104] T.-T. TSAI et Y.-M. TSENG. « Revocable certificateless public key encryption ». In : *IEEE Systems Journal* 9.3 (2015), p. 824-833.

[105] C. WANG et al. « Privacy-preserving public auditing for data storage security in cloud computing ». In : *Infocom, 2010 proceedings ieee*. Ieee. 2010, p. 1-9.

[106] C. WANG et al. « Privacy-preserving public auditing for secure cloud storage ». In : *IEEE transactions on computers* 62.2 (2011), p. 362-375.

[107] C. WANG et al. « Toward publicly auditable secure cloud data storage services ». In : *IEEE network* 24.4 (2010).

[108] H. WANG. « Identity-based distributed provable data possession in multicloud storage ». In : *IEEE Transactions on Services Computing* 8.2 (2015), p. 328-340.

[109] H. WANG. « Proxy provable data possession in public clouds ». In : *IEEE Transactions on Services Computing* 6.4 (2012), p. 551-559.

[110] H. WANG et al. « Identity-based remote data possession checking in public clouds ». In : *IET Information Security* 8.2 (2013), p. 114-121.

[111] L. WANG et al. « Scientific cloud computing : Early definition and experience ». In : *2008 10th ieee international conference on high performance computing and communications*. Ieee. 2008, p. 825-830.

[112] Q. WANG et al. « Enabling public auditability and data dynamics for storage security in cloud computing ». In : *IEEE transactions on parallel and distributed systems* 22.5 (2011), p. 847-859.

[113] R. E. WEBER. *United States diplomatic codes and ciphers, 1775-1938*. Routledge, 2017.

[114] J. WOHLWEND. *Elliptic curve cryptography : Pre and post quantum*. Rapp. tech. MIT, Tech. Rep, 2016.

[115] H. WU et B. ZHAO. « Overview of current techniques in remote data auditing ». In : *Applied Mathematics and Nonlinear Sciences* 1.1 (2016), p. 145-158.

[116] H. XIONG. « Cost-effective scalable and anonymous certificateless remote authentication protocol ». In : *IEEE Transactions on Information Forensics and Security* 9.12 (2014), p. 2327-2339.

[117] K. YANG et X. JIA. « DAC-MACS : Effective data access control for multi-authority cloud storage systems ». In : *Security for Cloud Storage Systems*. Springer, 2014, p. 59-83.

[118] K. YANG et X. JIA. « Data storage auditing service in cloud computing : challenges, methods and opportunities ». In : *World Wide Web* 15.4 (2012), p. 409-428.

[119] S. YU et al. « Achieving secure, scalable, and fine-grained data access control in cloud computing ». In : *INFOCOM, 2010 Proceedings IEEE*. Ieee. Ieee, 2010, p. 1-9. ISBN : 1424458366.

[120] S. YU et al. « Attribute based data sharing with attribute revocation ». In : *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*. ACM, 2010, p. 261-270. ISBN : 1605589365.

[121] Y. YU et al. « Cloud data integrity checking with an identity-based auditing mechanism from RSA ». In : *Future Generation Computer Systems* 62 (2016), p. 85-91.

[122] K. ZENG. « Publicly verifiable remote data integrity ». In : *International Conference on Information and Communications Security*. Springer. 2008, p. 419-434.

[123] J. ZHANG et Q. DONG. « Efficient ID-based public auditing for the outsourced data in cloud storage ». In : *Information Sciences* 343 (2016), p. 1-14.

[124] J. ZHANG, P. LI et J. MAO. « IPad : ID-based public auditing for the outsourced data in the standard model ». In : *Cluster Computing* 19.1 (2016), p. 127-138.

[125] Y. ZHANG et al. « Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing ». In : *Information Sciences* 379 (2017), p. 42-61.

[126] Y. ZHU et al. « Cooperative provable data possession for integrity verification in multicloud storage ». In : *IEEE transactions on parallel and distributed systems* 23.12 (2012), p. 2231-2244.