Composition of cryptographic mechanisms and watermarking for the protection of externalized genetic data

David Niyitegeka

DOCTORAT BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THESE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPERIEURE MINES-TELECOM ATLANTIQUE
BRETAGNE PAYS DE LA LOIRE - IMT ATLANTIQUE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par

David NIYITEGEKA

Composition de mécanismes cryptographiques et de tatouage pour la protection de données génétiques externalisées

Thèse présentée et soutenue à l'IMT Atlantique, le 15 Décembre 2020
Unité de recherche : Laboratoire de Traitement de l'Information Médicale (LaTIM / UMR 1101)
Thèse N° : 2020IMTA0225

Rapporteurs avant soutenance :

Anne-Sophie JANNOT	MCU-PH (HDR), Hôpital Européen Georges Pompidou-AP-HP, Université de Paris
Huazhong SHU	Professeur, School of Computer Science and Engineering, Southeast University

Composition du Jury :

Président :	Marc CUGGIA	Professeur des Universités – Praticien Hospitalier, Université de Rennes 1, CHU Rennes
Examinateurs :	Reda BELLAFQIRA	Maître de Conférences, IMT Atlantique
	Anne-Sophie JANNOT	MCU-PH (HDR), Hôpital Européen Georges Pompidou-AP-HP, Université de Paris
	Huazhong SHU	Professeur, School of Computer Science and Engineering, Southeast University
Dir. de thèse :	Gouenou COATRIEUX	Professeur, IMT Atlantique
Co-dir. de thèse :	Emmanuelle GENIN	Directrice de Recherche, INSERM UMR 1078

Composition of cryptographic mechanisms and watermarking for the protection of externalized genetic data

David NIYITEGEKA

A thesis presented for the degree of
Doctor of Philosophy

LaTIM Laboratory
ITI Department, IMT Atlantique
Brest, France
15th December 2020

In loving memory of my uncle Jean Claude NTIRENGANYA.

Acknowledgements

It was a great pleasure for me to do my PhD thesis at the LaTIM laboratory, IMT Atlantique. With these few lines, I would like to thank everyone who in one way or another participated to the good progress and achievement of this work, hoping that I did not forget anyone.

I would like to express my sincere gratitude to my thesis advisors. In would like to thank Professor Gouenou coatrieux. It has been an honor to be his PhD student. Thank you for your ideas, your help, your guidance, your patience and all encouragements you gave me in these years of learning and hard working. Thank you for encouraging me even when conditions were not the most favorable. I would like to thank Professor Emmanuelle Genin for the time she spent helping, guiding and correcting me. Your advices and guidance allowed me to have a better understanding of genetic data and each of our discussions was helpful. I would like to express my deeply-felt thanks to Doctor Reda Bellafqira for his support, his patience and for pushing me toward my goal. Thank you for believing in me and in my abilities, and for the time and the patience you have given me over these years. You gave me a lot of precious advices, not only in thesis work but also in my real life.

I would like to address many thanks to Professor Marc Cuggia for accepting to be the president of evaluation jury. I would like to thank Professor Anne-Sophie Jannot and Professor Huazhong Shu for accepting the evaluation of my thesis work. The comments and suggestions you provided helped me to improve the quality of my dissertation.

Thank you also to Thomas E. Ludwig, Musab Al-Ghadi, Younes Talibi Alaoui, Mario Südholt, and Fatima-Zahra Boujdad from the privGen project. Our discussions were always helpful and full of new ideas.

Many thanks to my friends: Sahar, Khalid, Haja, Ibrahim, Abdoulaye, José, Jules, Olivier, Bruno, Aimable, Honoré, Fils Vainqueur, Théophile, François-Xavier, Evariste, Samson, Léonard, Vedaste, Vital, Youssef Omari, Désiré, Félicien, Fidèle, Angélique, Emil, Mica, Jacqueline, Abdelkader, Placide, Henriette, Lony, Francine, Raymond and Patrick for your help, support and encouragement. I wish you all the best in everything you do.

I would like to express my deepest gratitude to my family, in particular my father, my mother, my brothers, my sister, and my uncles Félicien Semajoro and Jean Claude Bunane for supporting and

encouraging me since the beginning of my studies. Finally, a heartfelt thank you to my lovely wife Solange for always staying by my side, for her love, caring and encouragement over these years.

Composition of cryptographic mechanisms and watermarking for the protection of externalized genetic data

David NIYITEGEKA
ITI Department, IMT Atlantique
15th December 2020

Abstract

Today, cloud computing allows researchers and health professionals to flexibly store and process large amounts of genetic data remotely, without a need to purchase and to maintain their own infrastructures. These data are especially used in genome-wide association studies (GWAS) in order to conduct the identification of genetic variants that are associated with some diseases. However genetic data outsourcing or sharing in cloud induces many security issues in terms of privacy, integrity, traceability and confidentiality. Therefore, there is a need for protecting genetic data during their sharing, storage and their processing in the cloud environments.

During this PhD thesis, the conducted work aims at securing genetic data that are being stored and/or processed on the cloud. To do so, we developed several new security tools that are based on watermarking and cryptographic mechanisms (e.g., encryption, secure hash functions, secure multiparty computation), as well as on the combination of them such that watermarking and encryption that we call as "crypto-watermarking" mechanisms. Basically, watermarking consists on the imperceptible insertion of a message into data, and this message can be used for identifying data ownership, or controlling data integrity, etc. The main advantage of this mechanism is that it enables the access to the data while keeping them protected. On its side, secure multiparty computation (SMC) allows two or several parties to jointly compute a function over their own data while keeping these data private. Regarding encryption, it is the process that converts a clear message into an encrypted message which is comprehensible for only the person who has the secret decryption key.

In the first part of this work, we have focused on developing new solutions that are based on homomorphic encryption (HE) as well as the watermarking of encrypted data. HE allows performing linear operations such as additions or multiplications on encrypted data without decrypting them, with the guarantee that the result equals to the one computed on clear data. This allows data processing without getting access to clear data. We developed a privacy-preserving method that allows to compute the secure collapsing method based on the logistic regression model using fully homomorphic encryption. Next, we have exploited the semantic security property of some homomorphic encryption schemes in order to develop a crypto-watermarking method that allows the verification of integrity for encrypted data.

Homomorphic encryption have been proposed in various solutions for conducting privacy-preserving GWAS. However, they have significant computational and storage overhead, which makes them often impractical for real life applications. To overcome this issue, in the second part of our work, we have developed a framework that allows secure performing of GWAS for rare variants. Association studies performed in this framework are cases-control studies and are secured based on the combination of several mechanisms such as secure hash functions and encryption. At last, we studied watermarking of genetic data used in GWAS. We have

developed a robust watermarking method. The way this scheme has been designed allows to ensure that the distortion introduced in genetic data by watermarking procedures does not interfere with the genetic association tests studied in this thesis. Our method is based on quantization index modulation (QIM) and majority vote, and can be used for traitor tracing and copyright protection of genetic data used in GWAS.

Key words: Security, genetic data, genome-wide association studies, watermarking, homomorphic encryption.

Composition de mécanismes cryptographiques et de tatouage pour la protection de données génétiques externalisées

David Niyitegeka
Département ITI, IMT Atlantique
15 Décembre 2020

Résumé

De nos jours, le "cloud computing" permet aux professionnels de santé et aux chercheurs de stocker et traiter de manière flexible de grandes quantités de données génétiques à distance ; cela à un coût minime et sans avoir besoin de maintenir une infrastructure propre. Ces données mutualisées sont notamment utilisées dans des études d'association pangénomiques ("Genome-Wide Association Studies" ou GWAS) afin d'identifier des variants génétiques associés à certaines maladies. Cependant, l'externalisation ou le partage de ces données sur le cloud induit de nombreux problèmes de sécurité en termes d'intégrité, de traçabilité, de confidentialité et du respect à la vie privée. De plus, le génome humain est par nature une donnée très sensible étant une identité biologique unique d'un individu, en lien aussi avec ses proches. Par conséquent, il est impératif de protéger ces données lors de leur partage, stockage et traitement sur le cloud.

L'objectif de ces travaux de thèse est d'assurer la sécurité de données génétiques externalisées. Nous avons développé différents outils de sécurité fondés sur le tatouage, des mécanismes cryptographiques et leur combinaison. Dans un premier temps, nous avons proposé une version originale sécurisée de la méthode d'analyse "collapsing method", qui s'appuie sur la régression logistique, en utilisant le chiffrement homomorphe. Ensuite, nous avons exploité la sécurité sémantique des schémas de chiffrement homomorphes afin de tatouer des données génétiques chiffrées externalisées sur le cloud. L'objectif de cette méthode est de permettre au fournisseurs de cloud de protéger en termes d'intégrité, les bases de données sous sa responsabilité. Pour pallier les problèmes liés aux complexités de calcul et de mémoire des méthodes basées sur le chiffrement homomorphes, nous avons proposé un protocole qui permet de mener des tests d'association génétiques pour les variants rares de manière externalisée entre plusieurs unités de recherche en génétique. Ce protocole profite de la combinaison de plusieurs outils de sécurité tels que les fonctions de hachage, le chiffrement et PGP (Pretty Good Privacy) afin de sécuriser les données génétiques sensibles en termes de respect de confidentialité et du droit à la vie privée, sans augmenter les complexités de calculs et de communication de l'étude d'association à mener.

Enfin, pour tracer les données externalisées et assurer un service de "traçage de traître", nous avons développé une toute première méthode de tatouage robuste qui permet d'identifier l'utilisateur ou le fournisseur de services cloud qui détournerait ou divulguerait des données génétiques utilisées dans des GWAS.

Mots clés : Sécurité, données génétiques, études d'association pangénomiques, tatouage, chiffrement homomorphe.

Contents

List of Figures

List of Tables

Résumé en Français

De nos jours, les technologies de séquençage du génome progressent à un rythme rapide, et cela coïncide avec l'évolution rapide des technologies du multimédia, de communication et du cloud computing. En conséquence, des grandes quantités de données génétiques sont largement collectées, stockées, partagées et traitées de manière flexible, par des entreprises, des particuliers, des professionnels de santé ou chercheurs pour diverses raisons. Cela se fait à un coût minime et sans avoir besoin de maintenir une infrastructure propre. Dans le domaine de la santé, les données génétiques, en particulier les variants génétiques tels que les polymorphismes nucléotidiques ("Single Nucleotide Polymorphisms" ou SNP) peuvent guider plusieurs décisions médicales. Par exemple, il a été démontré que les femmes présentant certains variants génétiques dans des gènes BRCA ont jusqu'à 80% de chances de développer le cancer du sein [4]. En conséquence, l'identification des personnes porteuses de ces variants peut les aider à opter pour des mastectomies préventives [5].

En recherche scientifique, les données génétiques sont utilisées dans des études de population pour par exemple établir la relation entre différents groupes ethniques, ou pour effectuer des études d'associations génétiques qui permettent de découvrir des nouveaux variants ou traits génétiques associés à certaines maladies. Dans ce dernier cas, ces tests d'association sont généralement menés à l'aide des études d'association pangénomiques ("genome-wide association studies" ou GWAS), dont l'objectif est de fournir une meilleure compréhension de l'étiologie d'une maladie en détectant les variants génétiques impliqués dans cette maladie, pour un échantillon d'individus [6]. Pour ce faire, on parte du principe qu'une meilleure compréhension conduira à la prévention ou à un meilleur traitement de la maladie. Pour tester l'association dans des GWAS, l'approche la plus commune est l'étude cas-témoins, où les distributions de génotypes à différents marqueurs génétiques sont comparées entre deux grands groupes d'individus, un groupe témoin qui contient des individus en bonne santé et un groupe des cas, qui contient des individus affectés par la maladie.

Des études d'association pangénomiques nécessitent une grande quantité de données génétiques afin d'atteindre une certaine signification statistique. Dans ce cas, il est souvent nécessaire de partager ou externaliser ces données via des environnements cloud, entre différentes équipes de recherche génétiques travaillant sur la même pathologie. Cette externalisation permet d'accéder aux puissances importantes de calcul et de stockage offertes par le cloud. Cependant, le partage

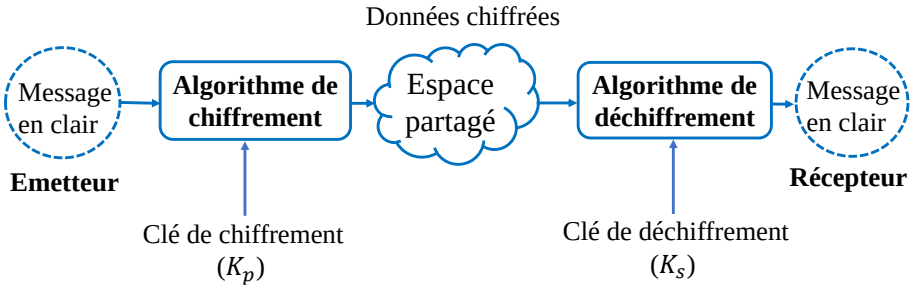

Figure 1: Étapes principales d'une chaîne de chiffrement classique. Les données en clair sont chiffrées par un émetteur à l'aide d'une clé de chiffrement K_s. Nous considérons que les données chiffrées sont ensuite partagées (par exemple via Internet), puis déchiffrées par un récepteur à l'aide d'une clé de déchiffrement K_p.

et/ou l'externalisation des données génétiques induit plusieurs problèmes de sécurité dus au fait qu'un ADN (acide désoxyribonucléique) humain est très sensible et représente l'unique identité biologique de son propriétaire [7]. Une simple fuite de données peut conduire à la divulgation de données génétiques et d'autres informations relatives à la santé sur pour des millions d'individus. Avec ces fuites, certains individus peuvent par exemple être traités différemment par leurs employeurs ou compagnies d'assurance car ils présentent un risque élevé de maladie ou un trouble héréditaire [8]. En outre, la fuite de données génétiques peut entraîner une divulgation indésirable des antécédents médicaux et l'identification des descendants ou des parents des individus concernés, car ils partagent certaines de leurs caractéristiques génétiques.

Ainsi, l'externalisation ou le partage de données génétiques doit être protégé et cette protection est une obligation légale qui varie d'un pays à l'autre, mais qui reste restrictive. Par exemple, le rapport présidentiel américain sur la sécurité des données génétiques donne les directives et les techniques de protection pour des données génétiques [9]. En France, la Commission nationale sur l'informatique et la liberté (CNIL) a récemment publié un aperçu de la législation concernant la collecte, le traitement et l'utilisation de données génétiques. Il précise que la protection de ces données est une condition essentielle lors de leur collecte, utilisation ou traitement [10]. Par conséquent, il faut protéger ces données lors de leur partage ou externalisation dans un environnement cloud. Cette protection consiste à assurer divers objectifs de sécurité, tels que:

- **Respect à la vie privée** la propriété qui consiste à protéger des informations sensibles des individus.
- **Confidentialité** qui consiste à s'assurer que l'information n'est accessible qu'aux seules personnes autorisées.
- **Intégrité** la propriété qui consiste à assurer l'exactitude de l'information, en évitant les modifications de données non autorisées.
- **Traçabilité** qui correspond à la capacité d'identifier tous les éléments ou individus ayant accédé, transféré, modifié ou supprimé une information depuis son origine jusqu'à son utilisation finale ou dans un laps de temps donné.

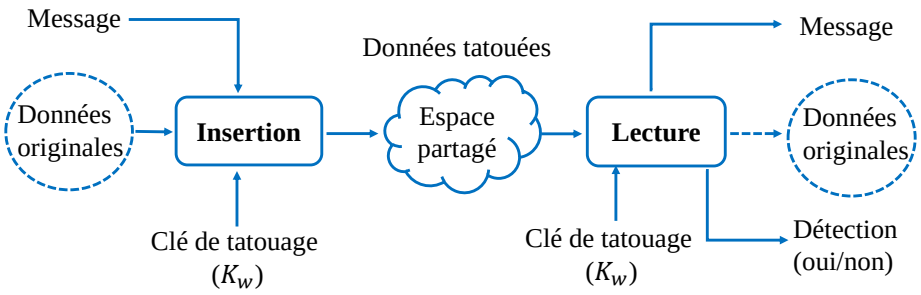

Figure 2: Étapes principales d'une chaîne de tatouage classique. Dans cette chaîne, on considère que les données tatouées sont partagées (par exemple via Internet) et qu'elles peuvent être modifiées ou manipulées illégalement entre les étapes d'insertion et de lecture. Au stade de la lecture, le message inséré est lu et/ou extrait, et dans le cas d'un tatouage réversible, les données originales peuvent être entièrement récupérées.

Plusieurs mécanismes ont été proposés afin d'assurer la sécurité des données génétiques externalisées et/ou partagées. Une liste non exhaustive comprend le contrôle d'accès, la gestion des droits des utilisateurs, la confidentialité différentielle, les signatures numériques, les fonctions de hachage, le calcul multipartite sécurisé, le chiffrement et le tatouage. Tel que décrit en Figure 1, le chiffrement permet de transformer à l'aide d'un algorithme de chiffrement et d'une clé de chiffrement K_p, un message en clair en un message chiffré incompréhensible. Le message chiffré ne peut être déchiffré avec l'algorithme de déchiffrement que si le récepteur du message dispose de la clé de déchiffrement K_s. De son côté, le chiffrement homomorphe permet d'effectuer des opérations linéaires telles que des additions et multiplications sur des données chiffrées sans avoir besoin de la clé de déchiffrement; le résultat, une fois déchiffré est égal à ce qui serait obtenu sur des données en clair.

Le chiffrement homomorphe permet d'assurer la confidentialité des données ainsi que le traitement des données chiffrées. Néanmoins, il ne permet pas d'effectuer toutes les opérations sur ces données, en particulier les opérations non linéaires telles que la comparaison ou la division. Le calcul multipartite sécurisé (SMC) est alors une solution pour effectuer ces types de traitements. Il permet à un ensemble de parties ou de participants différents (au moins un client et un serveur) d'évaluer en toute sécurité une fonction sur leurs données privées respectives en tant qu'inputs de la fonction. Autrement dit, la fonction s'évalue de telle manière qu'aucune information autre que la sortie de la fonction ou un résultat convenu ne soit disponible pour les participants. Ce résultat qui est connu de tout le monde peut être, par exemple, un booléen, ou l'index de l'élément dans une base de données, et peut avoir diverses applications, y compris la prise de décision préservant la confidentialité sur des données génétiques.

Quant aux fonctions de hachage cryptographiques, ce sont des algorithmes qui prennent en entrée des quantités arbitraires de données et produisent des sorties de longueurs fixes appelées "hash". Ces valeurs peuvent ensuite être stockées à la place de données eux-mêmes, puis utilisées pour diverses applications, y compris la vérification de l'intégrité de données, la génération de nombres pseudo aléatoires, la vérification de mots de passe ou l'authentification de messages.

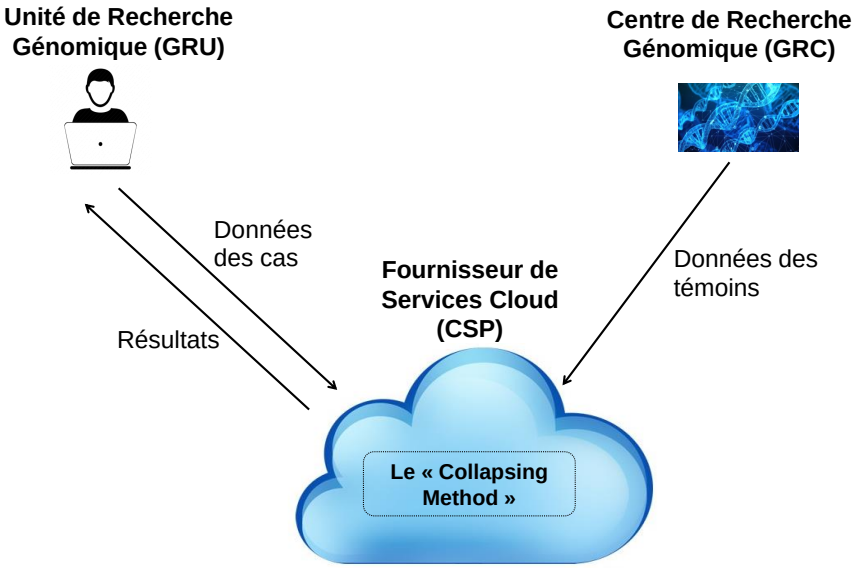

Figure 3: Scénario considéré dans l'externalisation des données génétiques sur le cloud

Les mécanismes de sécurité ci-dessus sont limités et plutôt a *priori*, car une fois outrepassées, les données ne sont plus protégées [11]. C'est en particulier le cas des données déchiffrées. C'est dans ce contexte que le tatouage s'impose, car il maintient une protection alors que les données sont accessibles et manipulées. C'est une protection complémentaire pour l'information. Par définition, et comme le montre la Figure 2, le tatouage consiste à insérer une marque ou un message sur la base d'une clé de tatouage K_w, dans un document multimédia hôte qui peut être une image, un signal audio, un signal vidéo ou une base de données. L'objectif de cette insertion peut varier selon le contexte, et dépend du lien entre le message et son hôte. Le message inséré peut servir: à la protection des droits d'auteur, au contrôle d'intégrité, à assurer la traçabilité des données, à l'ajout de méta-données, etc. Cette versatilité fait du tatouage une solution très intéressante dans le cadre de la protection des données génétiques [12].

Aucun de ces mécanismes (chiffrement, tatouage, SMC, etc) n'offre plus d'un seul objectif de sécurité, et il y a un intérêt à combiner différents mécanismes tels que le tatouage et le chiffrement, afin de bénéficier de leurs avantages respectifs et d'atteindre plusieurs objectifs de sécurité. Dans cette thèse, nous nous sommes intéressés à la protection des données génétiques externalisées lors de leur stockage ou de leur traitement dans des environnements cloud. Cette protection se base sur différents outils de sécurité qui sont le chiffrement (chiffrement homomorphe, chiffrement symétrique/asymétrique), les fonctions de hachage, le tatouage, le calcul multipartite sécurisé ainsi que la combinaison de plusieurs outils tels que le chiffrement et tatouage afin de développer de nouvelles solutions permettant une protection *a priori/ a posteriori* des données génétiques partagées et/ou externalisées.

Dans un premier temps, nous nous sommes focalisés sur la protection des études d'association pangénomiques dans des environnements cloud en utilisant le chiffrement complètement homomorphe. Ce dernier permet d'effectuer un nombre illimité d'opérations (additions et multiplications), sur les données chiffrées sans les déchiffrer. Comme le montre la Figure 3, la solution que

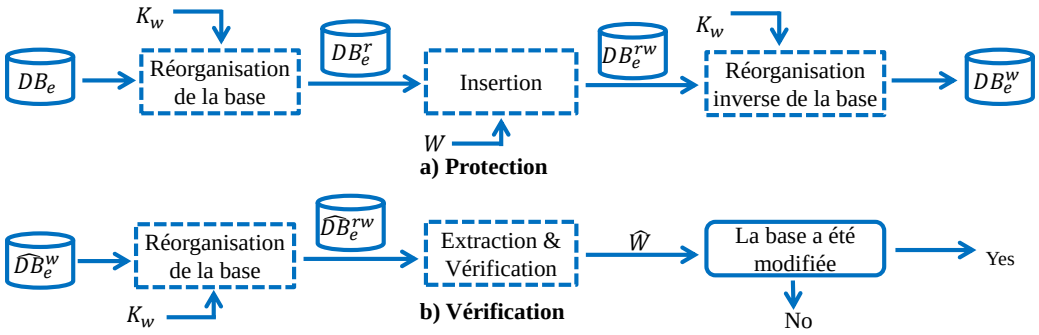

Figure 4: Architecture générale de la méthode proposée. K_w, W, $\hat{W}$ représentent la clé secrète de tatouage, le message ou la marque insérée ainsi que la marque récupérée, respectivement.

nous avons proposée est basée sur un scénario à trois entités: une unité de recherche génomique (GRU) qui possède des variants génétiques des individus malades (cas), un centre de recherche génomique (GRC) possédant des variants génétiques des individus non malades (témoins) et un fournisseur de services cloud (CSP). L'objectif est de comparer statistiquement les données de GRU à ceux de GRC afin de déterminer s'il y a une relation entre un gène et une maladie donnée. Par conséquent, toutes les données de GRU peuvent être stockées sur le cloud et un test d'association tel que le "collapsing method" [13], une méthode basée sur le modèle de régression logistique peut être effectué de manière sécurisé. Pour ce faire, nous profitons d'une combinaison de chiffrement complètement homomorphe et le calcul multipartite sécurisé afin de sécuriser le "collapsing method". Nos résultats expérimentaux indiquent que la méthode proposée fournit les mêmes résultats sur des données chiffrées que celles obtenues sur des données en clair.

Les solutions basées sur le chiffrement homomorphe garantissent le respect à la vie privée et la confidentialité des données mais ces données chiffrées peuvent rencontrer d'autres problèmes de sécurité en termes d'intégrité du point de vue du fournisseur de services cloud. Cela peut être causé par des erreurs de transmission ou des modifications non autorisées qui peuvent être effectuées par des attaquants ou des sous traitants malveillants dans le cas où le cloud externalise aussi ces données. Pour résoudre ce type de problèmes, nous avons proposé une solution qui combine le chiffrement homomorphe et le tatouage afin de garantir à la fois la confidentialité et l'intégrité des données génétiques externalisées. Pour ce faire, nous exploitons la sécurité sémantique (propriété par laquelle un message clair peut avoir différents messages chiffrés) que possèdent certains schémas de chiffrement homomorphe, afin d'insérer un message dans une base de données chiffrée. Cela permet aux fournisseurs de services cloud de vérifier l'intégrité de bases de données chiffrées homomorphiquement et externalisées par leurs propriétaires, à l'aide du tatouage.

La figure 4a. illustre l'architecture générale du système qui permet de vérifier une base de données sur la base de notre solution. On peut y voir deux procédures principales : la protection et la vérification de la base de données. La procédure de protection permet d'insérer une preuve d'intégrité ou un message binaire W dans une base de données chiffrée DB_e. Pour ce faire, cette base est d'abord organisée de manière sécrète par le biais d'une fonction de hachage cryptographique et une clé secrète K_w. Ensuite, la base organisée est subdivisée en plusieurs blocs. Un bit du message est insérée dans chaque bloc et après l'insertion du message, la base de données est réorganisée pour obtenir une base chiffrée et tatouée DB_e^w. La procédure de vérification est menée

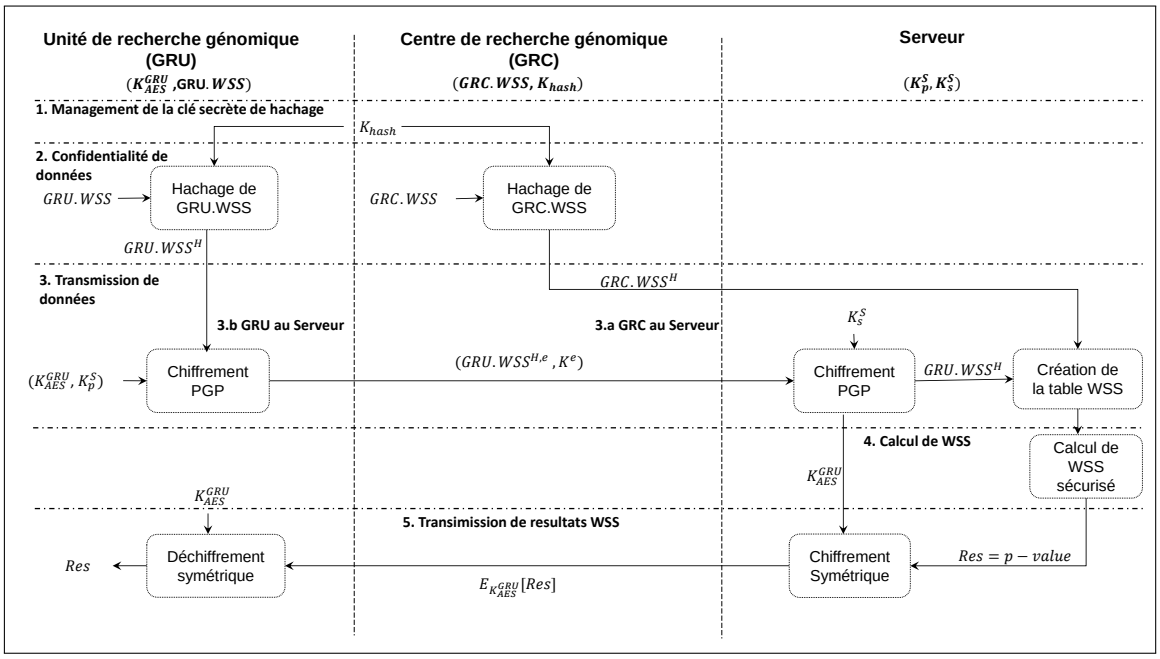

Figure 5: Différentes étapes de notre protocole WSS sécurisé dans le cas d'un gène.

de manière similaire (voir Figure 4b.). Pour vérifier l'intégrité d'une base de données suspectée $\widehat{DB_e^w}$, nous effectuons d'abord sa réorganisation secrète basée sur la clé K_w. Ensuite, un message $\widehat{W}$ est extrait et comparé au message W. Si $\widehat{W}$ et W sont différents, la base de données d'origine a été modifiée illégalement. De plus, l'étape de vérification nous permet d'identifier les éléments de la base qui ont été modifiés.Cette solution est dynamique dans le sens où les procédures de protection et de vérification d'intégrité peuvent être menées tout au long du cycle de vie de la base de données. c'est-à-dire qu'elle permet d'effectuer des opérations de mise à jour telles que la modification, la suppression ou l'ajout de données dans la base de données tout en étant toujours protégée par le tatouage. Les résultats expérimentaux effectuées sur une base de données contenant des variants génétiques ont montré une efficacité et une capacité élevées de notre solution, dans la détection de différentes modifications illégales de données, avec une précision de localisation élevée. Les solutions précédentes ainsi que plusieurs solutions proposées dans la littérature basées sur le chiffrement homomorphe [14–20] permettent d'effectuer des GWAS de manière sécurisée. Cependant, ils présentent des complexités de calcul et de stockage importantes, ce qui les rend souvent inutilisables pour les applications dans le monde réel [21]. Pour résoudre ces types de problèmes, nous proposons une nouvelle méthode de sécurité qui permet d'effectuer des études d'association génétiques de manière sécurisé, sans augmenter la complexité de calcul et de stockage. Notre solution permet sécuriser des algorithmes tels que le "Weighted-Sum Statistics" ou WSS utilisés dans des études d'association pour des variants rares. Comme introduit précédemment, notre solution s'appuie sur une architecture composée par un GRU possédant des variants génétiques des individus atteints par la maladie (cas), un GRC avec des variants génétiques issues des individus non malades (témoins), et un fournisseur de services cloud. Pour procéder à l'identification de gènes avec des variants génétiques rares impliquées dans une maladie, le GRU doit comparer les cas aux témoins grâce à des études d'association pangénomiques.

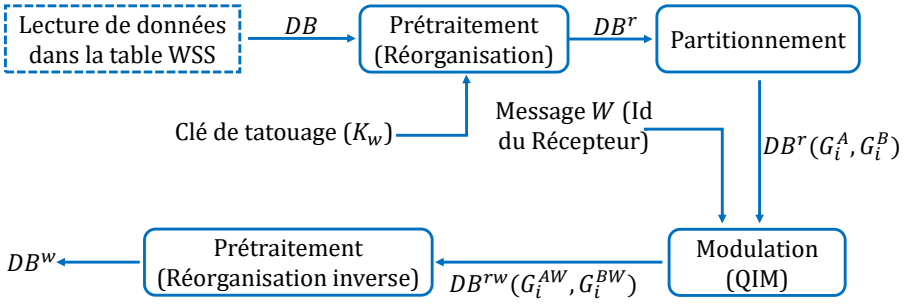

Figure 6: Architecture générale de la méthode de tatouage robuste proposée pour des données GWAS.

Notre solution positionne GRC comme un proxy entre GRU et le fournisseur de cloud. Cela permet d'utiliser des mécanismes cryptographiques classiques pour conduire en toute sécurité un GWAS sans augmenter la complexité de calcul, contrairement à l'état de l'art actuelle. Plus précisément, nous montrons comment la confidentialité des données sensibles peut être assurée avec une fonction de hachage cryptographique basé sur une clé secrète sans avoir besoin de modifier les algorithmes statistiques ou leurs résultats. Dans notre protocole, le cloud effectue simplement des analyses statistiques sur des données partiellement hachées. De plus, nous introduisons une nouvelle contrainte de confidentialité: l'identité de GRU doit rester inconnue du cloud car cette connaissance peut lui donner des indices sur les données de GRU (par exemple, les maladies et les gènes d'intérêt). Nous montrons comment le "Pretty Good Privacy" (PGP) peut être utilisé pour résoudre ce type de problème. Nous illustrons notre protocole dans le cas d'un test d'association de variantes rares, l'algorithme WSS, réalisé sur des données génétiques réelles. Le WSS sécurisé donne les mêmes résultats que sa version non sécurisée sans augmenter la complexité. De plus, notre protocole peut être étendu aux différents algorithmes de tests d'association génétiques utilisés pour des variants rares. La Figure 5 décrit les différentes étapes de notre protocole dans le cas de la protection du WSS pour un test effectué sur un gène.

La plupart des méthodes de tatouage proposées pour les données génétiques se concentrent sur l'ADN moléculaire pour diverses raisons (dissimulation des données, protection du droit d'auteur, contrôle d'intégrité ou tout simplement le stockage des données dans l'ADN) [22–27, 27–33, 33, 34]. Cependant, à notre connaissance, aucune solution de tatouage n'a été proposée pour les données génétiques utilisées dans des études d'association génétiques, comme celles utilisées pour le WSS. Le tatouage de ces données peut permettre d'assurer leur intégrité, la divulgation illégale d'informations ou la protection des droits d'auteur. Ainsi, nous proposons une nouvelle méthode de tatouage robuste permettant de tatouer des données génétiques utilisées dans des GWAS. Elle vise à assurer la traçabilité des données génétiques, c'est-à-dire identification de la personne ou entité qui est à l'origine d'une divulgation illégale d'informations ou de la protection du droit d'auteur de ces données. La solution que nous proposons est basée sur la modulation par quantification d'index (QIM) et le vote majoritaire [35]. Comme le montre la Figure 6, pour insérer un message, les données sont d'abord collectées dans une base de données DB. Ensuite, cette base de données est réorganisée en utilisant une clé de tatouage K_w. La base de données réorganisée

est partitionné en plusieurs groupes et chaque groupe est aussi divisée en deux sous groupes. Dans la suite, un bit du message est inséré dans tous les colonnes de chaque groupe en modulant la cardinalité du nombre de génotypes égales zéros dans les sous groupes. Lors de la lecture, un bit du message est détecté et extrait dans chaque colonne du groupe et, un vote majoritaire permet de décider quel bit du message est le bon. Dans notre solution, le message est secrètement inséré dans la base de données les données sans compromettre les résultats des tests d'association génétiques qui peuvent être effectués sur ces données. Cela veut dire que l'identification des variants candidats ou des gènes impliqués dans une pathologie donnée donne les mêmes résultats que sur les données originales. Ceci est confirmé par les résultats expérimentaux conduits sur les données utilisées dans WSS.

Introduction

Recently, genome sequencing technologies have progressed at a rapid pace, and this coincided with the rapid evolution of cloud computing and communication technologies. As a consequence, a large amount of genetic data are widely collected, stored, shared and processed by companies, individuals, health professionals or researchers for various reasons. In healthcare, genetic data, especially genetic variants such as single nucleotide polymorphisms (SNPs) can guide several medical decisions. For example, it has been shown that individuals with certain genetic variants in the BRCA genes have up to 80% chance of developing breast cancer [4]. Therefore, identification of individuals who carry these variants can help them to opt for preventive solutions such as mastectomy [5]. In research, genetic data are being used for population studies where these data are used for example to establish relations between different ethnic groups, or for discovering new associations in-between genetic traits and some diseases. In latter case, association tests are usually conducted using genome-wide association studies (GWAS). The objective of GWAS is to allow the better understanding of disease aetiology by detecting the correlation in-between genetic variants and disease traits in population samples [6]. To test for association in GWAS, the usual design is a case-control one where genotype distributions at different genetic markers are compared between samples of individuals affected by the disease of interest (cases) and unaffected individuals from the same population (controls).

GWAS require large amount of genetic data in order to achieve statistical significance. In this case, it is often necessary to outsource or/and share these data via cloud environments, between different genomic research teams that are working on the same pathology. However, genetic data sharing or/and outsourcing induces several problems in terms of data security due to the fact, a human DNA is sensitive and represents the unique biological identity of its owner [7]. A single data breach can leak genetic data and other health-related information on millions of individuals. With this leakage, individuals may for instance be treated differently by their employers or insurance companies because they have particular variants in a gene that can cause or increase the risk of an inherited disorder or disease [8]. In addition, genetic data leakage may cause an unwanted disclosure of medical history of individuals and identification of descendants or relatives of the affected individuals as they share some of their genetic characteristics. Thus, outsourcing or sharing of genetic data must be protected and this protection is a legal obligation which varies from one country to another, but it remains restrictive. For instance, the U.S. Presidential report on genetic

data security discusses policies and techniques to protect genetic data [9], or more recently, the National Commission on Informatics and Liberty has published an overview on legislative about the collection, the processing and the use of genetic data. It states that the protection of genetic data is essential condition during their collection, use or processing [10]. Therefore, the protection of genetic data consists in ensuring various security objectives [36]:

- **Privacy** the property which consists on the protection of sensitive information of individuals.
- **Confidentiality** which consists on ensuring that information is only accessible to authorized users
- **Integrity** that consists on avoiding unauthorized modifications of data
- **Traceability** which corresponds to the capacity of identifying all the elements that have accessed, transferred, modified or deleted an information from its origin to its final use or in a given period of time.

Several mechanisms have been proposed in order to ensure the security of outsourced and/or shared genetic data. A non-exhaustive list includes access control, user rights management, differential privacy, digital signatures, secure multiparty computation, encryption, watermarking and secure cryptographic hardware. Secure multiparty computation allow multiple parties to compute a common function without revealing their inputs. It is used for ensuring the confidentiality of data. Encryption is the process of converting an information or a message into unintelligible in such a way that only authorized parties who have the secret decryption key can get the access to clear message. On its side, homomorphic encryption allows the computation of linear operations such as additions and multiplications on encrypted data discarding the need for the decryption key; the output when decrypted equals to what would be obtained on unencrypted data. However, each of these mechanisms rarely responds more than one security objective at a time [11]. Encryption ensures the confidentiality of data, it offers an *a priori* protection or in other words, once data are decrypted they are no longer protected. Watermarking was proposed as a complementary mechanism which offers an *a posteriori* protection of data. It leaves data accessible and processed while maintaining them protected by an imperceptible message which can be security attributes, a digital signature or an authentication code. Thus, there is an interest in combining different mechanisms in order to benefit from their respective advantages and ensure more than one security objective.

In this thesis work, we have focused on the protection of outsourced genetic data during their storage or processing in cloud environments, by using different mechanisms that are encryption (homomorphic, symmetric, asymmetric), watermarking, secure multiparty computation as well as the combination of encryption mechanisms with watermarking techniques so as to develop new solutions that make possible an *a priori/a posteriori* protection of shared and/or outsourced genetic data.

This thesis is structured as follows: chapter 1 provides some general definitions about the main domains we addressed in order to position the problems we focused on. We will thus come back

on: introduction to genetic data and the security needs for shared or outsourced genetic data. We expose the ethical and legislative rules which impose the protection of genetic data in terms of several security objectives such as privacy, confidentiality, integrity and traceability. We then give an overview about different data security mechanisms (e.g., encryption, watermarking, secure multiparty computation, hash functions). We discuss the possible combination of several security mechanisms, in particular the combination of watermarking and cryptographic mechanisms as well as the limits of these mechanisms. Finally, We will take this opportunity to present an exhaustive state of the art of existing security approaches from the literature that were developed for the protection of shared and/or outsourced genetic data.

In the second chapter, we present the first contribution of our work that consists in the protection of genome-wide association studies (GWAS) in cloud environments using fully homomorphic encryption. This method allows a Genomic Research Unit (GRU) who possesses genetic variants of cases to statistically compare his/her data with genetic variants of controls from a Genomic Research Center (GRC). Therefore, all data of GRU can be stored in the cloud and a secure association test [13] based on the logistic regression model can be performed. To do so, we take advantage of fully homomorphic encryption and of secure multiparty computation so as to conduct collapsing method in a secure manner. Experiment results indicate that the proposed method provides the same results on encrypted data as the ones achieved on clear data, and it allows to ensure the confidentiality of genetic variants used in GWAS.

Even if homomorphic encryption-based methods ensure data confidentiality, encrypted data may face other security issues in terms of integrity. This can be caused by the transmission errors or unauthorized alterations performed by attackers. Chapter 3 overcomes these issues by combining homomorphic encryption and watermarking so as to ensure at the same time the confidentiality, privacy and integrity of outsourced data. To do so, we exploit semantic security that some homomorphic encryption cryptosystems have, so as to allow the cloud service providers to verify the integrity of encrypted databases outsourced by their owners, with the help of watermarking. The proposed method is dynamic in the sense that, it allows update operations such as modification or addition of data into the database while still protected by the watermarking. As in chapter two, the performance of this scheme is evaluated and tested. It shows high efficiency and capability in detection of different illegal data modifications with a high location precision.

Several solutions based on homomorphic encryption have been proposed for conducting privacy-preserving GWAS. However, they have significant computational and storage overhead, which makes them often impractical for real life applications [21]. Chapter 4 overcomes this issue by addressing a new privacy-preserving GWAS framework that allows perform of rare variant case-control association studies such as weighted-sum statistic (WSS) algorithm is a secure way. It relies on a Genomic Research Unit (GRU) with genetic variants from cases, a Genomic Research Center (GRC) with genetic variants from controls and the cloud. To conduct the identification of genes with rare genetic variants that are involved in a certain disease, GRU needs to compare cases to controls through genome-wide association studies. Our scheme positions GRC as a proxy between GRU and the cloud. That makes it possible to use classical cryptographic mechanisms for securely conducting GWAS without increasing computation complexity, contrarily to actual

state of the art proposals which are of very high complexity. In particular, we show how sensitive data confidentiality can be ensured with secret key-based cryptographic hash with no need to modify statistical algorithms. In our protocol, the cloud simply conducts statistical analyses on partially hashed data. In addition, we introduce a novel privacy constraint: GRU's identity should remain unknown to the cloud as this knowledge can give it clues about GRU's data (e.g., diseases and genes of interest). We exhibit how Pretty Good Privacy (PGP) can be used to solve this problem. We illustrate our protocol in the case of one rare variant association test, the Weighted-Sum Statistic (WSS) algorithm, carried out on real genetic data. This secure WSS achieves the same accuracy as its nonsecure version with no increase of complexity. Furthermore, we establish that our protocol can be extended to the different association test algorithms used for rare variants.

Most of the watermarking methods proposed for genetic data are focusing on molecular DNA for various reasons (data hiding, copyright protection, integrity control or data storage). However, to the best of our knowledge there is no watermarking solution that was proposed for genetic data used in GWAS. Watermarking of these data can allow to ensure their integrity, illegal information disclosure or copyright protection. Thus, in chapter four, we presents a new robust watermarking method. It aims at ensuring traitor tracing of genetic data, i.e., identifying the person who is the origin of an illegal information disclosure or copyright protection of these data, and it is based on Quantization Index Modulation (QIM) and majority vote [35]. In this solution, the watermark is secretly embedded within genetic data used in GWAS, without violating the identification of candidate variants or genes involved in a given pathology, i.e genetic association studies results are not compromised. Finally, performances of scheme are theoretically evaluated and empirically tested.

CHAPTER 1

Security of outsourced and shared genetic data

With the rapid development of technology, whole genome sequencing has become less expensive and offers a great promise of research advances that could benefit all of the society. As a result, genomic research has quickly opened the way to several genetic data treatments that are used in personalized medicine, tests for predisposition to diseases, genealogical analysis, etc [36]. However, even though this evolution is interesting, it comes with several needs in terms of information security, and these ones must be defined before proposing new and more appropriate solutions.

This chapter aims at giving general definitions of genomic data, especially genetic data and detailing security needs for these data. It is divided into three sections: in the first section, we will introduce genetic data, the possible processes that could be conducted on these data, their domain of use as well as the security risks they are submitted to. In the second section, we will provide different security mechanisms proposed in the literature so as to ensure the protection of data such as access control, homomorphic encryption, watermarking, secure multiparty computation, etc. and all of these mechanisms must be integrated into a framework defined by one or more security policies. Finally, We will provide an overview on methods that were proposed for securing genetic data before conclude the chapter.

1.1 Genetic data

It is important to know what genetic data is, its use, how and why it is shared and/or outsourced before describing why and how this data can be secured. This section addresses these different questions.

1.1.1 Human genome

The human body is made up of billions of cells where each has one nucleus, and this nucleus contains 23 pairs of chromosomes. These chromosomes contain our genetic information which corresponds to our DNA (deoxyribonucleic acid). Basically, DNA is composed of two strands of

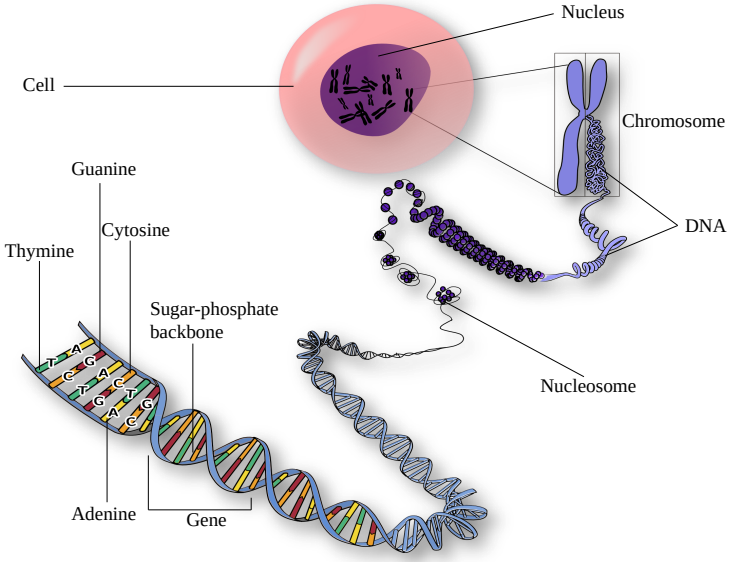

Figure 1.1: The structure of human DNA. From pixabay, a bank of copyright free images [1]

four nucleotides or bases that are adenine(A), cytosine (C), thymine (T) and guanine (G), where A bonds with the complementary T, G bonds with the complementary C, and vice versa (see Figure 1.1). The complete set of all DNA contained in one cell is called genome, and the total number of bases in one genome is estimated to three billions.

In our DNA, the basic unit of heredity is a particular sequence of bases called gene. One gene contains about 1000 to over 2 millions bases and the number of genes in a human genome is estimated to 20300 genes [37]. In each gene, every three successive nucleotides make up a codon. Since there are only four bases, the total number of possible codons is $4^3 = 64$. All these codons constitute what we call the genetic code, a set of rules used by living organisms to translate each information encoded in DNA into proteins [38]. More clearly, genetic code defines how sequences of codons, specify which amino acid will be added next during protein synthesis. The figure 1.2 illustrates all 64 codons and their corresponding amino acids. Notice that in all 64 codons, three of them are called STOP codons and they do not correspond to any amino acid but instead, they indicate the end of the protein chain. The remaining 61 codons correspond to 20 amino acids. As there are only 20 amino acids for 61 codons, some codons represent more than one amino acid and this is referred as degeneracy. In addition, for each codon of each amino acid, the first two bases are the same. For instance, the amino acid Alanine (Ala/A) can be represented by one of four codons “GCA, GCC, GCG and GCT” and the first two bases for Alanine are “GC”. As we will see in chapter 5, these properties are of importance in developing some kind of DNA watermarking methods.

In each genome or gene, there are two distinct regions: protein-coding (pcDNA) regions and non-protein coding (ncDNA) regions. Protein-coding regions are responsible for the encoding or translation of organism’s proteins. On other hand, non-protein coding regions do not encode any proteins and it was believed for long time that these regions have non function in living organisms. However, recent works demonstrated that up to 80% of these regions may have some functions of

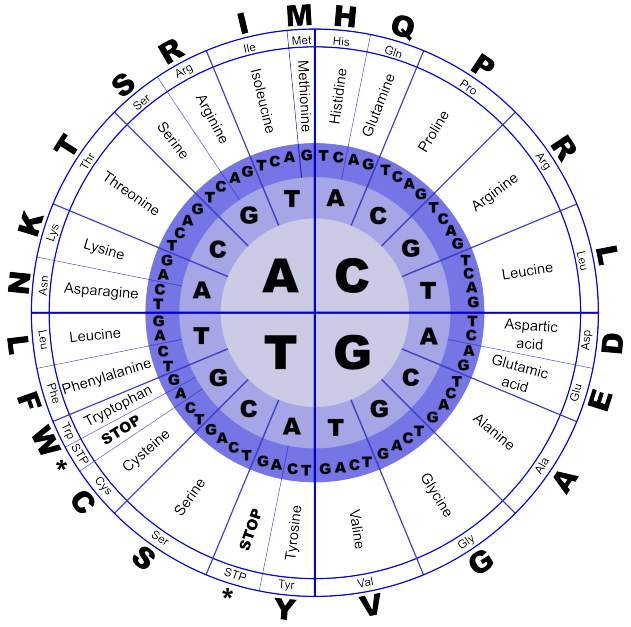

Figure 1.2: Representation of genetic code with all 20 amino acids and 3 STOP codons. From Openclipart [2]

regulation of gene expression [39]. The remaining part contains DNA with no function, referred to as junk DNA. We will see in chapter 5 that these regions can be used for message embedding as their modifications do not affect organisms.

The living beings from the same species have all the same number of genes, each controlling a particular behaviour. However, excepts for identical twins, individual's DNA is unique with one chromosome of each pair coming from the father and one from the mother. These chromosome may show some differences in genes due to mutations that change one base to another. These differences may lead to different protein and thus have an impact on individual characteristics. For instance, for each person, there is a gene responsible of eyes' color but nucleotides which are in the gene of an individual with blue eyes are not the same for an individual with green eyes. These differences in-between individuals' genomes are called genetic variants.

Depending on the frequency and effect of the variants, one will call them polymorphisms if they are frequent (generally with a frequency of the minor allele above 1%) with no functional effect or, one will call them mutations when they are rare and potentially deleterious. One can distinguish three types of polymorphisms as described below:

- **SNPs (Single Nucleotide Polymorphisms):** As shown in Figure 1.3, they correspond to a substitution of a single base or nucleotide that occurs at a specific position of the genome. In individual genome, SNPs occur almost once in every 1,000 nucleotides on average. Thus, there are an estimated 4 to 5 million SNPs in one individual's genome [40].
- **Indels:** An insertion/deletion, commonly abbreviated "Indel" is a type of polymorphism in which a specific DNA sequence is inserted or deleted in an individual gene or genome.

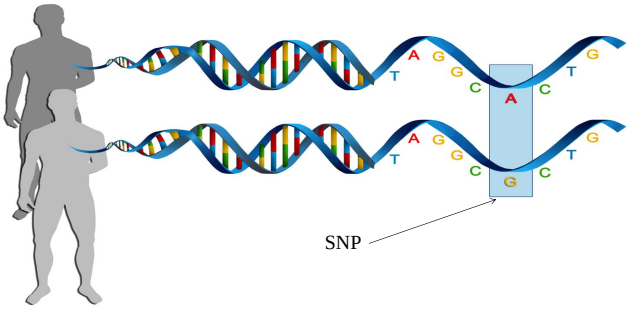

Figure 1.3: An example of genetic variant. Case of a single nucleotide polymorphism where A is substituted by G. From [3]

Indels are widely spread across the genome and one genome contains an estimated 0.5 to 1 million [40].

- **Structural variants:** These are all genetic variations that can occur over a large part of genome. They can be chromosomal rearrangements of genes where several DNA sequences are broken off or located at some other positions on the chromosome (translocation), inversions of nucleotide orders, presence of several copies of genes, etc. One genome contains an estimated 2,100 to 2,500 structural variants [40].

As we will see in next chapters, genetic variants are of importance in many genetic processing such as genomic/genetic testing or genome-wide association studies where they are used in order to decide if a specific party of genome such as gene is associated with a disease.

1.1.2 What is genetic data ?

Recent years, genetic data have been the subject of several legal definitions. The European Council has proposed two formulations [41]. Genetic data refers to "*all data, of whatever type, concerning the hereditary characteristics of an individual or concerning the pattern of inheritance of such characteristics within a related group of individuals*". It also refers to "*all data on the carrying of any genetic information (genes) in an individual or genetic line relating to any aspect of health or disease, whether present as identifiable characteristics or not. The genetic line is the line constituted by genetic similarities resulting from procreation and shared by two or more individuals*". In article two of International Declaration on Human Genetic Data adopted by UNESCO on 16 October 2003, genetic data are defined as "*information about heritable characteristics of individuals obtained by analysis of nucleic acids or by other scientific analysis*" [42]. More recently, the General Data Protection Regulation (EU) 2016/679 (GDPR) [43] defined genetic data in its article four as "*personal data relating to the inherited or acquired genetic characteristics of a natural person which give unique information about the physiology or the health of that natural person and which result, in particular, from an analysis of a biological sample from the natural person in question*". From all these definitions, we can already emphasize the personal and hereditary characters of genetic data as well as their privacy. Thus, genetic data corresponds to all data relating to genetic characteristics and gives unique information on physiology or health status for an

individual. Some people consider genetic data as health data. However, genetic data have several characteristics that differentiate them with common medical or other data. Genetic data are unique, static, familial, valuable and contain individual health behaviors. We detail these characteristics as follows.

- Individual DNA contains information about her/her blood relatives. Therefore, genetic data are familial data. Depending on the context, these data can reveal individual's biological paternity, his/her susceptibility to certain diseases or implication in a criminal case. Thus, genetic data may reveal many things about people other than the individual from whom they were derived. In addition, it has been demonstrated that if we have genomes of few people in the family, it is possible to infer other family members' genomes [44].
- The DNA of any two individuals are different and can be easily differentiated from one another. This means that genetic data of an individual are unique. As a consequence, individual genetic data correspond to his/her biological identity, and this is useful in many domains for several purposes (e.g., criminal forensics).
- Individual DNA does not change much over time. This means that genetic data are relatively static and they remain relevant to their owner over long periods of time, even between many generations. As a consequence, the value of genetic data are likely to increase over time because the information we are able to derive from studying those data will improve. For instance, before 1980s, it was not possible to identify an individual who has committed a crime using his or her DNA but nowadays, DNA analysis is helping for this identification [45]. Therefore, the release of genetic information are not limited in time contrary to classic medical data whose value decline with time.
- Our DNA contains more valuable information and till now, we do not know everything about human genome. This conducts to different ways of its public perception. For instance, violent behaviors of an individual are influenced by the environment. However, we do not know either if this comes from their genomes or not [46].
- Genetic data contains information about individual health and behavior. It is now possible to determine genome parties or genes that are associated with some diseases or behaviors. For instance, Jia *et al* have identified and confirmed several pleiotropic genes such as CLEC16A, CUX2, etc., that are associated with seven autoimmune/autoinflammatory diseases. In another example, breast cancer can be diagnosed using BRCA1 and BRCA2 genes [47].

In this Ph.D. work, we are focusing on the protection of genomic data during their sharing and/or outsourcing. These data are usually collected and kept in variant call format (VCF) files [48] which are used for storing genetic variants for each sequenced individual. We describe these files in the next section.

VCF header	##file format=VCFv4.1											
	##ALT=<ID=NON_REF, Description="Represents any possible alternative allele at this location">											
	##FORMAT=<ID=GT, Number=1,Type=String, Description="Genotype">											
	##FORMAT=<ID=GQ, Number=1,Type=Integer, Description="Genotype Quality">											
	#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	Patient 1	Patient 2	Patient 3	...
body	1	69511	rs2691305	A	G	3789.90	.	AC=6;AF=1	GT:AD:DP:GQ :PL	1/1:0	1/1:0	1/1:0	...
	1	721290	rs12565286	G	C	155.16	.	AC=1;AF=0.167	GT:AD:DP:GQ :PL	0/0:13	0/0:7	0/1:9	...
	12	721450	rs2977675	G	A	213.16	.	AC=2;AF=0.333	GT:AD:DP:GQ :PL	0/0:89	0/1:78	0/1:83	...

Figure 1.4: An example of VCF file.

1.1.3 How are genetic data generated ?

As introduced in previous sections, we are interested in protecting genetic data, especially during their storage or processing by genetic association studies. To conduct these studies, genetic data such as genetic variants are used. Genetic variants are the result of a long processing process which begins by data sequencing. More clearly, in order to obtain genetic variants, samples are collected for several individuals and are sequenced using appropriate sequencers [49]. These ones output FASTQ files each contains the raw NGS reads for each individual. These reads are then aligned on the human reference sequence so as to produce a SAM (sequence alignment map) file which is also equivalent to BAM (Binary Alignment Map) file for each individual. Notice that BAM stores the same data in a compressed binary representation. As there are different human reference sequences, for comparison purpose, one needs to make sure that the same reference sequence is used for the different individuals in a sample and across the specific study. The next step consists the variant calling that corresponds to extracting genetic variants from BAM files. Results are stored in variant call format (VCF) files [48] and each reports all the positions on the genome where the individual has a variant compared to the reference sequence. The variant call format (VCF) was developed in order to standardize large scale genetic variants sharing and storage in order to facilitate genetic studies such as GWAS. A VCF file corresponds to text file that consists of three parties which are meta-data lines, a header line and data lines (see Figure 1.4). Meta-data lines which begin the file and included after ## provide the descriptions about data lines. The header line started by # names the columns for data lines. Finally, data lines follow the header line and each data line or record represent one variant at a given position in the genome. In a VCF file, a data line contains several columns including:

- CHROM which is an identifier from the reference genome and corresponds to chromosome number. It indicate the chromosome in which the variant belongs;
- POS that refers to the position of first base on reference sequence;
- ID which is a unique identifier for each record if exists;
- REF that represents reference base(s);
- ALT that corresponds to alternate base(s);

- QUAL which is a measure of the quality in the identification of ALT;
- FILTER that represents filter status of the variant and INFO that contains additional information on the description of the variant such as number of individuals, frequency alleles, protein coding regions, etc.)

These columns are followed by FORMAT of variant for genotyped individuals. Notice that FORMAT specifies the data type for genotypes of each individual. In a VCF file, genotypes are reported as numbers separated by '|' or '/'. Thus, we have genotype $0/0$ if the individual is homozygous reference, $0/1$ if the individual is heterozygous and $1/1$ if the individual is homozygous alternate.

To conduct genetic studies such as genetic testing or genome-wide association studies (GWAS), individuals that are either unaffected (controls) and affected (cases) by disease are genotyped so as to produce a sample composed of thousands or up to millions of genetic variants that are then stored into VCF files. After that, an intermediary step is performed in order to generate other files with filtered data, and are specific for each GWAS algorithm. In the sequel, we will come back to these files in chapters 4 and 5, especially Weighted-Sum Statistic (WSS) files which contain genetic variants extracted from VCF files in order to conduct WSS algorithm.

1.1.4 Genetic data processing and its applications

As explained in introduction, genetic data contains more valuable information and can have several applications. In this section, we discuss the importance of genetic data, especially, their applications in healthcare, direct-to-consumer services, genomic research as well as legal and forensic services.

1.1.4.1 Healthcare applications

It has been demonstrated that variations in human genome can influence health. In fact, some changes in a particular gene will have an adverse immediate effect on individual's health or at some point in the future generations [50]. Nowadays, many traits that are associated with diseases have been reported in the literature [36] and their identification allows the discovery of new treatments. Genetic information has allowed the identification of several neurodegenerative diseases such as Huntington's disease (HD) [51], blood disorders such as Sickle cell anemia (SCA) [52] or metabolic disorders such as phenylketonuria (PKU) [53]. HD is caused by a mutation in the HTT gene within the chromosome 4, SCA is caused by the mutation is HBB gene and PKU is caused by two compound heterozygous mutations that are c.165 delT and c.284-286 delTCA in the PAH gene. Even though some genetic diseases have no known intervention to assist in the improvement of an individual's health status, others are manageable through changes in diet or pharmacological treatments. For instance, the identification of X-linked hypohydrotic ectodermal dysplasia (XLHED) which is caused by the mutation in the gene EDA has allowed the treatment of several fetuses. To do so, prenatal interventions have been conducted in order to administrate

proteins to fetuses. The infants were able to sweat normally and XLHED-related disease had not developed [54]. Therefore, in healthcare several genetic testing can be conducted either requested by the doctors or individuals. Herein, we resume some examples of genetic tests that can be conducted in healthcare:

- to prevent risks of illness or if there is no cure for the illness, anticipated genetic testing allow some life decisions.
- to investigate the cause of an observed phenotype. These tests can be performed any time for any individual from an in utero fetus through to old age individual.
- make, confirm, refute or clarify a diagnosis, particularly prenatal, for a genetic disease .
- to establish the genes that are likely to be at the origin of the development of a disease for individual or or his family.
- to allow the adaptation of the medical care of a patient according to his/her genetic profile in order to identify the drugs presenting a particular risk of ineffectiveness or toxicity.
- for family members who are likely to be at increased risk of genetic disease (or to carry it) due to family history (risk of recurrence).
- in carrier testing i.e., genetic testings that are conducted in order to identify either healthy individuals who may have inherited a mutated gene for a particular disease but which is not expressed in those individuals or healthy individuals who are carriers of balanced chromosomal rearrangements such as translocation and whose future generations are at risk of being affected.

1.1.4.2 Genetic data in research

It has been demonstrated that some parts of human genome is associated with a significant number of traits and complex disorders, and till nowadays, many new associations are being discovered. In order to facilitate these discoveries, several large scale genome sequencing projects have been initiated so as to identify and characterize genome or genome parties such as genes of interest in human populations. For instance, the Human Genome Project [55] is the first project that allowed the sequencing of whole human genome. Since then, other projects such as the 1000 Genomes Project [56], an international collaboration project between China, the UK, Germany and the USA, or the 100,000 Genomes Project, a UK Government project that has allowed the sequencing of whole genomes from National Health Service patients [57] have been developed.

In addition, with the decrease of the cost of genome sequencing, large scale genetic data is being collected, stored in order to be used by researchers for identifying new genes or genetic variants that are associated with diseases and in some cases, this may help for developing appropriate personalized treatments for patients [58]. This is the case of UK Biobank [59] that was filled with genetic data of 500,000 participants so as to be used in genetic and health research or the Michigan Genomics Initiative (MGI) which is a collaborative research effort among physicians

and researchers at the University of Michigan with the goal of harmonizing patient electronic medical records with genetic data to gain novel biomedical insights. Notice that genetic data from large populations increase the probability of finding the genetic correlation between genetic variants and diseases or traits. To do so, different technologies, analytical tools and study designs such as genome-wide association studies (GWAS) [6] are being used. GWAS are successfully uncovering several genetic variants or genes associated with complex traits and disorders. For example, in [60] authors proposed a genome-wide association study that has allowed the identification of genetic variants which increase the risk for emergence of suicidal ideation (TESI) during treatment with antidepressants.

On the other hand, in order to simplify data sharing between individuals or researchers, several web services called "beacons" as well as genome aggregation databases were developed [61]. They provide allele-presence responses to different queries such as "Do you have a genome that has a specific nucleotide base T at position 12217 on chromosome 2 in your genome ?". For example, the beacon SFARI contains genetic data from families that have children affected by autism spectrum disorder. It has supported more than 550 investigators studying autism-related research worldwide [62]. Thus, collecting, sharing and storing large scale genetic data is one of the principles keys in genetic research, especially in genetic association studies.

1.1.4.3 Direct-to-consumer services

With the rapid diminution of sequencing costs, at-home genetic test services commonly known as direct-to-consumer (DTC) services have become a major industry [63]. DTC is a type of genetic testing that is available directly to individuals without having to go through hospital or other health care professionals. They allow individuals or consumers collecting their genetic data, their processing and analysis without the involvement of a health professional. For instance, in 1996 an online company ancestry.com was launched in order to allow individuals to conduct historical searches and family records so as to obtain genealogical clues, as well as genetic tests for learning about their genetic ancestry. Since then, several companies such as 23andMe, iGENEA, DNA Tributes or Family Builder have been created [64], and they offer various genetic applications for consumers [65]. They propose many services including ancestry tests, paternity tests and ethnicity tests, genealogy tests, etc.

In some cases, DTC companies enable consumers to perform genetic compatibility tests with potential partners, or allow volunteer individuals the opportunity to provide their genetic data in order to support genetic research projects. This is the case of 48 000 individuals that have been recruited by 23andMe in order to participate in a scientific study about major depressive disorder, schizophrenia and bipolar disorder [66]. In health care, DTC are being used for determining disease susceptibility risk but this kind of genetic test is always contested in the context of DTC because of lack of regulations. In this case, genetic test is usually conducted at specific parties of genome such as genes and the possible corresponding diseases. For instance, the genes BRCA1 and BRCA2 are known to have genetic variants that are responsible of a certain number of hereditary cancers such as ovarian, breast and prostate cancers. Thus, results of this kind of testing can

potentially lead to important health decisions such as mastectomy if an individual is a carrier of those variants in BRCA genes.

Searching for better lifestyle of curiosity is another application of DTC. A study conducted by Johns Hopkins University researchers on genetic data of 1,046 individuals from three companies 23andMe, Navigenics, and deCODE, has demonstrated that, 94% of consumers decided to take DTC tests for curiosity reasons while 91% did these tests for learning about potential future diseases [64]. Notice that DCT genetic testing have many benefits compared to traditional genetic testing which are part of the health care system. They are accessible and affordable for everyone at any time.

1.1.4.4 Use of genetic data in legal and forensic

Nowadays, genetic data are being used for identification of individuals in legal and forensic investigations, due to the fact that genetic data does not change or changes little over a lifetime, i.e., it is static. This allows the identification of a given individual in an investigation purpose. For example, genetic data taken from individuals and crime scenes have been used as evidence by law enforcement authorities in order to identify criminals and to exonerate innocent individuals. In 2005, Ricky Davis from California (USA) was convicted of second-degree murder of 54 years old Jane Hylton committed in the 1985. However, in 2020 he becomes the first person in California to be exonerated with the help of DNA analysis combined with family tree research [45]. As DNA is inherited, genetic data from a family members can also used for criminal investigations so as to identify unknown suspected individual by comparing his or her DNA to relatives who are not themselves directly involved in a crime. In addition, genetic data can also be used for conducting DNA-based parentage testing in the case of denial of paternity.

To perform these identifications, several techniques such as restriction fragment length polymorphism (RFLP) [67] or short tandem repeats (STRs) analysis [68]. RFLP consists on analysing long fragments of genetic variants using southern blot. The major drawback of this technique is that large quantities of genetic data are needed. To overcome this issue, STRs analysis was proposed. As seen in section 1.1, STRs are genetic variants with repeated units that are 2 to 7 nucleotides in length, with the number of repeats varying from an individual to another, making STRs effective for individual identification purposes. For example for a particular repeat, like TCGTT, some individuals inherited four copies of it from one parent, others inherited six, eight or ten. This has made these repeats useful variants. In [68], authors state that human identification in legal and forensic investigations can settled using a small number of STRs variants. In the US, 13 variants are needed while in most of european countries 10 STRs variants are needed for identifying an individual in forensic cases, missing person investigations or and paternity testing.

Even though, in many countries, the number of DNA identification databases is growing, it is not clear how law enforcement agencies will continue to collect, store, and use this information in future. The Supreme Court of the United States has recently ruled that law enforcement can collect and store the DNA of suspects, even if they are subsequently exonerated. We leave our DNA behind nearly everywhere we go; currently there are no restrictions on how the police can

collect the DNA of criminal suspects in the hope of solving cases where other strategies have been unsuccessful [69]. While collecting the DNA without a warrant, known as "abandoned DNA", by the police can be very useful, leaving it unregulated not only affects police behaviour but also challenges the individual right to security. While abandoned DNA is a very hot topic, it can at the time of writing be collected, sequenced and used by anyone without consent in the USA. Furthermore, the question of "whose DNA profiles should be kept in the DNA databases ?" is the most controversial policy issues about the formation of these databases. Authors of [70] argue that having population-wide databases with strict privacy protections would be more effective and fair compared to store the profiles of only convicted or arrested individuals.

1.1.5 Security risks for genetic data

As discussed in section 1.1.4, genomic data has numerous distinguishing features and is subject to several applications. During their collection, sharing, storing or processing, genetic data can be subjected to many security threats or risks due to the fact that individual genome is unique. In this section, we give an overview on these risks so as to show on the one hand, the need of genetic data protection and in the other hand to find the "bast" protection mechanisms that can be deployed.

These risks can be classified into three categories that are accidents, errors or malicious attacks [71], and separable according to the nature of the threats (technical, physical, environmental, human, etc.) [72]. These risks independently or jointly affect many security objectives in terms of privacy, integrity, confidentiality, traceability and availability. We will details these security objectives in next section.

- **Accidents**: They correspond to all problems related to the environment or functionalities of information system that hosts genetic data. There are many accidents including:
  - Partial or total destruction of hardware or software materials due to forces of nature such as floods, earthquakes, tornadoes, landslides, electrical storms or fire, etc.
  - Hardware or software malfunctions which may be caused by power failure, network loss, faulty memory medium, etc.
  - All events that are caused by negligence, failure or absence of individuals in charge of information system, system handling and maintenance.

  Whatever we can do, most of these risks will always be present and the only thing we can do is trying to restrict their consequences.
- **Errors**: The responsibility of users and stakeholders is important but the design flaws of software and systems play a significant role. Thus, in a information system (IS), errors may come from several sources such as:
  - Input errors,
  - Information transmission errors,
  - Manipulation errors in IS operating functions,

  - Errors from the misuse of the IS.

- **Malicious attacks**: If errors are often identified risks, this is not the case of malicious attacks which are unpredictable and unavoidable. As soon as the human factor is present, it becomes difficult to assess these attacks. In addition, it is particularly difficult to find examples. Many factors can be the origin of malicious attacks including blackmailing or economic interests. Malicious alterations can be ranged from removing evidence of a prescription or diagnostic error to the liability of a third party. Thus, they can be the consequence of direct, total or partial physical destruction of files and software or their backup, or indirect (virus, malware), or even the result of identity theft or intrusion by a third party allowing access to the operating functions of the information system. Genetic data are highly sensitive, as an individual genome enables its unique identification. Thus, it is the biological identity of each individual. In addition, as seen in previous sections, genetic data may reveal the current and future susceptibility of specific diseases for a given individual or his/her relatives. Therefore, these uniqueness of genetic data impose greater security risks for these data and their owners from malicious attacks. Genetic data risks can be classified into three major groups according to where and how these data are used:

  - **Risks in genetic data sharing**: We have shown in section 1.1.4 that large scale genetic data are being shared in order to facilitate genomic research or other services. This is the case of the beacon SFARI that contains genetic data from families with a child affected by autism spectrum disorder, and it is used by researchers who work on autism disorder. However, SFARI could leak not only membership information for a given individual, but also phenotype information for that individual. Several attacks have been proposed whereby an attacker retrieves the identity of a target individual by relying on quasi-identifiers such as demographic information (e.g., linking to public records such as voter registries), date of birth, data communicated via social media, and/or search engine records, etc. For instance, the study proposed in [73] has reported that the identification of 30% of Personal Genome Project (PGP) participants can be conducted using demographic profiling including zip code and birthday dates. Sometimes quasi-identifier attributes such as zip code or date of birth are removed from these databases in order to protect participants (data anonymization) but it has shown that this kind of technique is ineffective [74]. For example, an attacker can infer the phenotype of the individual for an anonymized genome and use this information to identify the anonymous individual in others types of databases. In [74], it has demonstrated that genetic variants on the chromosome Y are correlated with the last name of male individuals, and this last name can be retrieved using public available databases such as genealogy databases. After recovering the name of the person, the complete identity can be found using other databases such as vote databases, etc. Finally, as these databases contain the disease association of the participants, their security must be ensured. Many complications such as false identification of surnames may compromise the success of this attack. In addition, in some societies, a surname is not a strong identifier and there is few chance to succeed individual identification. For example, 400 million people in China hold one of the ten common surnames and the

top hundred surnames cover almost 90 % of the population [75]. Thus, this strongly reduces the utility of surname inference for individual identification. Shringarpure and Bustamante [76] have presented an other example of inference attacks where they conducted it against beacons. In their attack, they repeatedly submit queries for genetic variants present in the genome of the targeted individual.

- **Risks in genetic data computation and storage**: During their storage or computation, genetic data may face several risks and attacks. For instance, we have seen that large scale data are being collected, stored and computed so as to enable genomic researches such as genome-wide association studies (GWAS). In most cases, storage and computation operations are conducted on the cloud as the cloud computing services are fast and cheap. Even without using the cloud services providers, allowing a third party to compute or store genetic data without any protection involves unwanted risks, as data might leak information from the secure enclosure of researchers [77]. Most of attacks that are conducted in this cases use genomic profile of the victim. In fact, an attacker gains access to the genetic variants of the victim. Then, it is used for identifying the victim from genetic databases with sensitive attributes (e.g., cases with hypertension, drug abuse, etc). Any match between the victim genome and the database links the person and the attribute. Pakstis *et al* [78] demonstrate that this attack requires only a small number of single nucleotide polymorphisms (SNPs), and a set of 45 SNPs is sufficient to provide matches between individual genome and his data in genetic database. Other attacks of this type have been studied [79–83].
- **Risks in genetic data analysis results**: Genetic data are used in several genomic researches such as genomic association studies (see section 1.1.4). However, it has demonstrated that results from these studies can still leaks information about participants. For instance, Homer *et al* [80] demonstrated that it is possible to identify the presence of an individual in a case group during a case-control association study. More clearly, a participant in a GWAS can be identified using aggregate allele frequencies and his DNA profile through the analysis these allele frequencies for a large number of SNPs. In addition, another study in [84] has shown that even a small set of statistics such as results of GWAS published can be used to identify the presence of an individual in the case group. This kind of attacks is conducted based on the pairwise correlation such as linkage disequilibrium among approximately hundreds of SNPs.

It is necessary to identify various risks to which genetic data are subjected in order to determine the security objectives. These latter are defined in next section with the intention of countering identified risks as we will see in section 1.2.

1.1.6 Security needs in genetic data sharing and outsourcing

Genetic data security is regulated by strict deontological ethics as well as national and international legislative rules. This is due to the sensitive nature, personal identifiable and the nominative aspect of pieces of genetic data, and to the fact that they are stored, shared/outsourced in open

environments such as cloud. Thus different security needs have to be considered and well defined before establishing appropriate security solutions. Many countries have been active in adapting their legislation to the protection of genomic data. For instance, in the USA, one must take care of the privacy and security rules imposed by the Health Insurance Portability and Accountability Act (HIPAA) [85]. These rules aim at ensuring that individuals' health information, including genetic information, is properly protected while allowing the flow of health information needed to provide and promote high quality health care and to protect the public's health and well being. In 2008, the president of the USA has signed into law the Genetic Information Non-discrimination Act (GINA), the goal of which is to protect individuals against discrimination using their genetic data and it makes it illegal for health insurance companies or employers to request or require individuals' genetic data or their family members [86]. In another example, the united kingdom government and the Association of British Insurers (ABI) have agreed on a policy framework, the Concordat and Moratorium on Genetics and Insurance which ensures that genetic data of an individual can not be used in an unfair or unclear manner by insurance companies and that individuals should not be treated differently based only on their genetic data [87]. All these regulations developed a set of commitments in terms of information security that medical entities, researchers or individuals must ensure.

All security commitments imposed by national legislative rules of different countries or international rules, include protecting: individual privacy, data integrity, data confidentiality and availability. These four main security objectives are also completed by authenticity control as well as traceability of information, usually considered in order to secure the complete flow of information. Beyond these legislative and deontological rules, there also exist national and international recommendations which provide implementation guidance, such as the rules stated by standards BS 7799, ISO 17799, ISO 27799 and ISO 27001. The standard BS 7799 that was created by the British standard institute (BSI) in the 90s, gives instructions of a good practice for the information system security. It has been adopted by ISO in 2000 so as to become ISO/IEC 1779 [88]. Since then, BSI has added a second part of the standard which is BS 7799-2. It focuses on how to implement an information system security management by referring to the structure of information security and to the identified controls. BS 7799-2 becomes ISO/IEC 27001 in November 2005 [88]. In healthcare, we have the ISO 27799 standard that has released in 2008. It addresses the information security management needs of the health sector and its unique operating environments [89]. The ISO 27799 (Health Informatics - Information Security Management in Health using ISO/IEC 27002) provides guidance to healthcare professionals or organizations on how best to ensure the security of health information. This concerns genetic data, as in some cases these data are considered as health information. Some other specific standards like those proposed by IHE (Integrating the Healthcare Enterprise) [90], can also used so as to complete security requirements for genetic data by defining specific security objectives. We introduce these security objectives before explaining how they can be assured in section 1.2.

- **Confidentiality** : Basically, confidentiality relates to information not being accessible or revealed to unauthorised individuals [91]. It can also be defined as status afforded to data or information indicating that it is sensitive for some reason. Therefore, it needs to be protected

against theft, disclosure, or both, and must be disseminated only to authorised individuals or organisations [92]. It is specially relevant in the case of nominative information such as medical data or genetic data. For example, an individual carrying genetic data related to genetic variants or genes that are known to increase the likelihood of a particular cancer or other genetic disease may be denied by the health insurance company for the coverage [93]. Thus, ensuring confidentiality of these data is needed during their storage, sharing or processing. We will see in section 1.2 some of many security mechanisms that have been developed in order to ensure data confidentiality.

- **Privacy**: Privacy consists on limiting access to an individual or identifying a person from his information [94]. In [95], authors proposed four categories of genomic privacy that are *i)* informational privacy which concerns the access to personal information; *ii)* physical privacy which corresponds to the access to persons and personal spaces; *iii)* decisional privacy that consists of governmental and other third-party interference with personal choices; and *iv)* proprietary privacy concerns which corresponds to the appropriation and ownership of interests in human personality. For all these categories the issue consists on the access on an individual through his/her data. As a simple example and as we have seen in 1.1.5, it has demonstrated that medical and demographic data used in Personal Genome Project, combined with genetic data can allow the identification of participants [73]. As we will see throughout this thesis, data privacy is a particularly important for genetic data, due to its biological nature.

- **Integrity**: Integrity verification is defined as a process of proving that a piece of information has not been modified by unauthorized users. For genetic data, integrity control corresponds to the protection of the accuracy and consistency of this data, avoiding unauthorized alterations or deletions. Data integrity can be compromised by many threats from accidental or malevolent data manipulations, erasures or transmission errors. For instance, as detailed in section 1.1.4, genetic data are being used in precision medicine the goal of which is to enable physicians to quickly, accurately and efficiently tailoring the right treatment according to the characteristics of each individual genome [96]. Thus, the integrity of this information is imperative in genomics/genetics or healthcare as illegally modifications of this data would affect physician decisions in diagnosing as well as individual health. In other words, incorrect information can result in hazardous events such as death of patients, or the prescription of the wrong medication for patients. Several solutions have been proposed for ensuring data integrity and we will come back to these ones in section 1.2.

- **Authenticity**: In general, data authenticity represents the fact that data proceeds from the source it is supposed to come from. This consists for example in asserting the origin of genetic sequences and its link to a given individual, or a sample of genetic variants that are associated to a certain disease and its link to a researcher or organization who works on that disease. Another example can be the authentication of data during genome sequencing. In fact, strains and their genome data are often mistakenly mislabelled during the process of genome sequencing, and this leads to wrong taxonomic interpretation. In part, this is because genome sequencing is carried out in central sequencing facilities where the chance

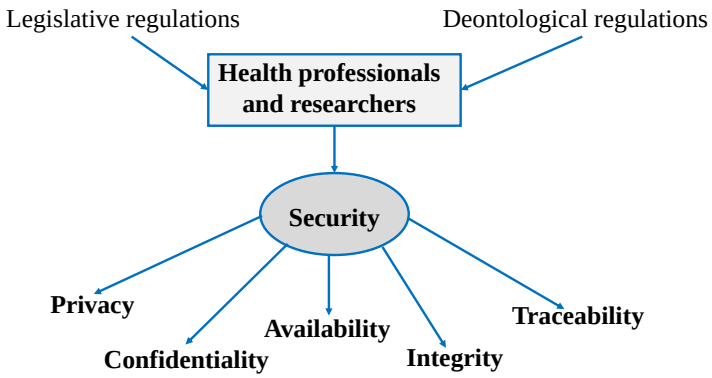

Figure 1.5: Security components for genetic data.

of mislabelling and contamination is relatively high. Thus, it is important to verify if a given genome sequence corresponds to the strain under investigation [97]. Both ensuring data integrity and data authenticity corresponds to the protection of the reliability of data. Thus, reliable genetic data can be used by researchers or health professionals in total trust. If it is possible to trace the data from its origin to its distribution (i.e., its existence), the concept of reliability becomes traceability.

- **Traceability**: Data traceability aims at identifying the persons or all the elements that have transferred, accessed, deleted or modified data from its origin to its final use or in a given period of time. When defining their security policies, health institutions or genomic research centers take special attention to this property as it serves to determine the responsible parts in case of negligence due to incorrect information manipulation [91].

- **Availability**: Each information system must be available to authorized users in order to be always useful. In case genetic data is used healthcare, genetic information must be accessible in any situation when needed and this availability is critical in case of emergency. This means that materials, software and communication channels that are needed for storing and accessing to this information must be fully operational taking into account the supported workload as well as the security mechanisms to use. Different threats may perturb the correct behaviour of an information system and most of them are non-malicious in nature and include unscheduled software downtime, errors, accidents, hardware failures and network bandwidth issues. Malevolent actions include various forms of sabotage can also be conducted with the intention of causing harm to an organization by denying users access to the information system. To counteract these security threats, different security mechanisms can be implemented as we will see in section 1.2.

In summary, as it is shown in Figure 1.5, legislative and deontological regulations impose ensuring genetic data security, and this consists of three essentials points that are: *i)* **confidentiality and Privacy** that consist on ensuring that only authorized users can access to genetic data as well as their owner; *ii)* **availability** that corresponds to the ability of an information system to be accessed by users at each time; and *iii)* **reliability** which consists on ensuring that data were

not illegally modified (i.e., its integrity) as well as the assurance of its origin and their owner (i.e., its authenticity). We recall that reliability becomes **traceability** if it is possible to trace data throughout its entire existence.

1.2 Implementing security in genetic data

The deployment of a security policy consists in exploiting mechanisms and devices that aim at securing information system and applying the rules defined by the security policy. This one specifies security rules and requirements that must be satisfied by an information system. These rules specify how information can and can not be accessed, all procedures of recovery management, new user registration and also how security services must be deployed, configured, parameterized, etc. Thus, security policy the deployment is a complex process, and as we have seen in section 1.1.6, different standards such as ISO 27001 or BS 7799 have been proposed in order to guide this process.

Each deployment of a security policy is started by a risk analysis which is conducted using different standards such as EBIOS [98], OCTAVE [99] or MEHARI [100]. The risk analysis process allows to measure the level of the risks (e.g., critical or not) and the identification of the objectives and security requirements for a given information system as well as for collected data that are stored, processed or shared through this system. After that, the identified risks can be countered or minimized using existing devices and security mechanisms. We can distinguish protection mechanisms into two categories: physical protection and logical protection. Physical protection mechanisms correspond to the materials that are used for counteracting unauthorized physical access, natural risks such as fire, robbery, flooding, etc. Thus, in order to protect, an information system, this one should be placed in a protected and isolated zone, where the access is well controlled. For instance, in order to counter physical access, badges or biometric authentication tools can be used. In addition, some cable locks that can be used to counter thieves should be deployed, and a regular maintenance ensures the proper functioning of the information system in terms of hardware and software. Notice that even though it depends of the security policy, as exposed above, it is usually provided by external service societies contractually linked to the health institution or genomic research center. These maintenance contracts take usually into account the constraints in terms of confidentiality, integrity and availability of genetic information.

Logical protection correspond to security mechanisms that are exploited at the software level. These are for instance user authentication methods (login, password, smart cards, etc.) [101]; access control using access control model such as OrBAC [102], cryptographic mechanisms (homomorphic encryption [103], secure multiparty computation [104], hash functions [105], etc.), certification management mechanisms that are used for distributing encryption keys (e.g., public key infrastructure (PKI)), network filtering mechanisms (e.g., Firewall), traceability mechanisms (e.g., message logging using SYSLOG, ODBC), intrusion detecting tools such as IDS or more recently, watermarking mechanisms [106]. In next section, we will describe some cryptographic and watermarking mechanisms as well as their respective limitations. As none of these mechanism can ensures all security objectives, these mechanisms can be combined in some cases, so as to ensure

more than one security objective (e.g., watermarking and encryption for protecting confidentiality and traceability).

1.2.1 Security mechanisms and their limitations

We distinguish here two categories of security mechanisms that are information system security mechanisms and data security mechanisms. Each mechanism was proposed for a specific security objective (confidentiality, integrity and authenticity, traceability and availability).

1.2.1.1 Information system security mechanisms

1. **User authentication**: It corresponds to a process that allows the verification by a device, the identify of a person who connects to a network resource, and this allows the protection of data from unauthorized access. Several schemes have been proposed in order to ensure user authentication, but the well-known is the strong authentication which is the combination of two different criteria: verifying the user's identity using passwords; and providing a proof of the user's identity using for instance smart cards. This solution can be associated with a token, which ensures an unique connection per user. Once the user connected, the token is assigned to the computer in which he is logged on. After that, no other connection will be allowed for the user somewhere else in the system. RSA SecurID [107] is an example of the strong authentication tool where two criteria are used to in order to ensure the protection of network resources. Herein, the authentication is based on a password or PIN and an authenticator. The latter is composed of a hardware token such as key job or a smart card, and a software token which is the RSA Authentication Manager Software [107]. Other mechanisms such as Windows Active Directory [108] or RADIUS [109] are also used for user authentication.

2. **Access control**: In previous section, we have seen that user authentication ensures the protection of data from non-authorized access [110]. Once the user has the green light for accessing to the information system, it is mandatory to control the activity that user can perform by defining his/her access rights. To respond to this issue, several access control models have been proposed. For instance, discretionary access control (DAC) model has been proposed for restricting the access to information system based on the identity of users or the groups to which they belong, or both; object ownership and permission delegation [111]. Other models such as the role-based access control (RBAC) [112], the organization based access control (OrBAC) [113], the attribute-based access control (ABAC) [114] or more recently smart contract-based access control [115] can also be used. Even though access control protects data, using only the authorization policy or the access policy can not counter all possible attacks, and it is sometimes possible for a user to bypass the mechanisms that implement this policy. Thus, it is suitable to reinforce data security using other mechanisms such as antivirus, cryptographic mechanisms, security audit, etc.

3. **Firewalls**: A firewall can defined as a collection of components that are interposed between two networks in order to filter traffic between these networks and according to a predefined

security policy [116]. If the information system is connected to another network, firewalls are used for protecting this system against intrusions. They make it possible to survey and restrict the access from the outside of information system such as internet to the inside (e.g., a local network, etc.) and vice et versa [117]. Thus, a firewall is one of mechanisms that are used for ensuring the access controls and as it is mentioned above, its main function is filtering the traffic by only letting packets from authorized addresses to pass. Notice that firewalls do not protect the confidentiality or integrity of the data circulating on the network, and so other mechanisms must be implemented in order to ensure these security objectives.

4. **Antiviruses**: A computer virus is defined as a computer program that can copy itself and infect a computer without the knowledge or the permission of the user. After being executed, a virus can modify other computer programs and inserting its own code [118]. For example, the conficker which is also known as downup virus had infected millions of computers all over the world and damages caused by this virus detected in 2008, are estimated at more than $ 9.1 billion [119]. Viruses are certainly one of the most important threats that face each information system. There exist different ways by which viruses can be inserted into an information system, this insertion can be conducted even if the information system is not connected to an open network. In addition, external data storage or sharing devices such as USB flash or hard disk drives can be infected. To counter these viruses, computer programs called antivirus (e.g., Avast Antivirus, McAfee, etc.) have been proposed in order to prevent, detect and isolate viruses, as well as the restoration of information system. The prevention consists of testing all memory units but also all network connections, databases and programs that can be imported. The virus detection corresponds to controlling the information system using one or several detection tools, and suspicious memory units should be isolated by disconnecting them. Regarding the information system restoration, viruses are first removed from the system using antivirus before reformatting the memory units and reinstalling the information system.

1.2.1.2 Data security mechanisms

A. Data encryption

In order to ensure the confidentiality of data, data encryption mechanisms are among the first security mechanisms that were proposed. We discuss this section how they work.

A.1 Principles of encryption

Data encryption is a process that transforms a clear message, known as plain-text, into an encrypted message known as cipher-text, that cannot be understood by anyone other than the person who created the message and the recipient. As illustrated in Figure 1.6, this process is conducted by means of an encryption algorithm parameterized by an encryption key K_p. Message decryption involves the transformation of the cipher-text into a clear message identical to the original one through a decryption algorithm and a decryption key K_s. There are two classes of encryption

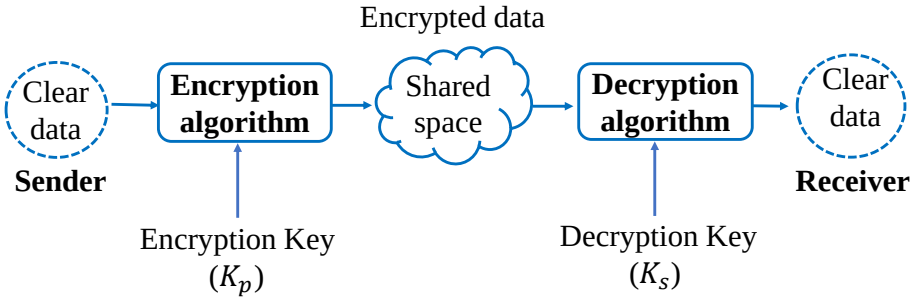

Figure 1.6: Main stages of a common encryption chain. The clear data is encrypted by a sender using an encryption key K_s. We consider that the encrypted data is shared (e.g., via the Internet) and then, decrypted by a receiver using a deciphering key K_p.

algorithms according to the dependence between the encryption and decryption keys: symmetric-key encryption or more simply, symmetric encryption and public-key encryption, i.e., asymmetric encryption.

Symmetric encryption is based on one key which is shared between the sender and the recipient. This means that the encryption key K_s and decryption key K_p are equals ($K_p = K_s$). Symmetric encryption is still highly used due to its rapidity, and until 1976, all the proposed encryption algorithms were symmetric. There exist two categories of symmetric encryption algorithms: stream encryption algorithms that work directly with data flows, for instance the RC4 [120], and block encryption algorithms that transform fixed-length strings (blocks) from the clear message into encrypted strings of the same length in encrypted message. There exist several block encryption algorithms including DES or triple DES (Data Encryption Standard), Blowfish, Serpent and AES (Advanced Encryption Standard) [121]. These algorithms are based on various block modes including Electronic Codebook (ECB), Ciphertext-Feedback (CFB), Cipher Block Chaining (CBC), Counter (CTR), or Output-Feedback (OFB) [122]. AES which was originally known as Rijndael, is the most commonly used symmetric algorithm [123]. This is due to the fact that AES is proven to be highly secure and fast. AES is the international standard set by the U.S. National Institute of Standards and Technology (NIST) in 2001 in order to be used for the data encryption [123]. It has a block size of 128 bits, but can have three different key lengths as shown with AES-128, AES-192 and AES-256 [124]. This standard replaced DES, which had been in use since 1977. In this thesis, in the case symmetric encryption is needed, we have opted for AES algorithm (see section 3.4 and section 4.3).

On the other hand, we have asymmetric encryption which is based on two different keys, the encryption key K_p and the decryption key K_s. These keys are distinct but still mathematically linked. However, theoretically, the knowledge of one of the keys does not allow obtaining the other one. K_p is public, and is accessible to all, while K_s is private and is only known to the recipient. The asymmetry comes from the fact that if a message is encrypted using K_p, it can only be decrypted using K_s. Therefore, for ensuring data confidentiality, it is necessary to encrypt the clear message with the recipient public key. Only him/her will be able to decrypt the encrypted message by means of his/her private key. On the other side, if a user encrypts a clear message

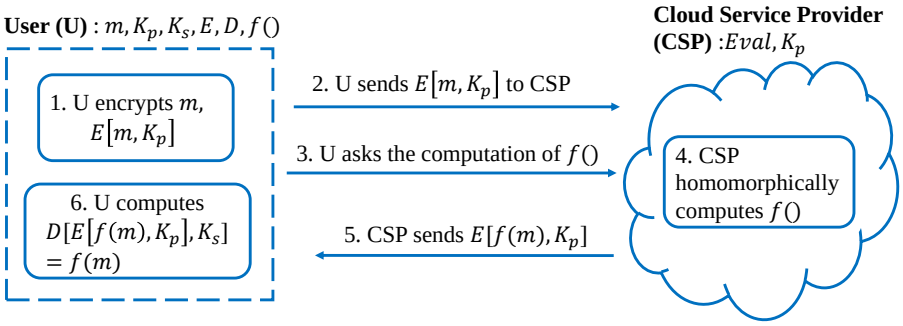

Figure 1.7: A simple example of HE use in cloud computing.

with his/her private key, everyone is able to decrypt it by means of the published public key and this reverse process is used for verifying the origin of the message. Thus, the user can not deny the emission of the message, and this ensures the non-repudiation and the authentication of the user. Different symmetric encryption algorithms have been proposed, and the most widely used algorithm is RSA (from its authors, Rivest, Shamir, Adleman) [125]. Recently, several authors have suggested the use of encryption, especially homomorphic encryption in order to ensure the privacy and confidentiality of genetic data during their storage and/or processing on the cloud for instance [36]. We detail in the sequel this mechanism

A.2 Homomorphic encryption

Homomorphic encryption (HE) is one of the most exciting new topics in cryptography research and is a promise for perfectly securing mechanism in cloud computing [14, 126]. It must allow a user to store his encrypted data on the cloud and user can ask the cloud to process these data without decrypting them. After processing, results are sent to user in encrypted form. From a historical perspective in cryptology, in order to perform operations on encrypted data with traditional encryption, there is no other solution than decrypting the data first. This is not the case of homomorphic encryption [127], the concept of which was introduced in 1978 by Rivest *et al.* [125] as a possible solution to the computing without decrypting problem.

By definition, each algebraic operation performed in the space of clear messages corresponds to another algebraic operation performed in the space of encrypted messages. To give a simple motivational HE example for a sample cloud application, as shown in Figure 1.7 let us consider the user U, first encrypts his or her sensitive data (Step 1), then sends the encrypted data to the cloud server providers (CSP) (Step 2). When the user wants to perform a function (i.e., query,), $f()$, over his or her own data, he or she sends the function to the CSP (Step 3). The CSP conducts a homomorphic operation over the encrypted data based on the function $Eval$ that allows the computation of the function $f()$ on encrypted data without accessing to the result (Step 4). This latter is sent to the user (Step 5) who will decrypt it using his or her own secret key in order to obtain $f(m)$ (Step 6). As seen in this simple example, the homomorphic operation, $Eval()$, at the CSP side does not require the private key of the user U. The function $Eval()$ is based on elementary operators (e.g., $\star$, $\circ$) that are defined in the space of clear messages $(\mathscr{M}, \circ)$ and

encrypted messages $(\mathscr{C}, \star)$. $\star$ and $\circ$ can both be the usual addition or/and the usual multiplication operators. Thus, if $m_1, m_2 \in \mathscr{M}$ are two clear messages, the homomorphic property is such that

$$D[[m_1, K_p] \star E[m_2, K_p], K_s] = m_1 \circ m_2 \tag{1.1}$$

where $D()$ and $E()$ are the decryption and encryption functions, respectively. $\star$ represents the algebraic operator performed in the space of encrypted messages while and $\circ$ is the algebraic operator conducted in the space of clear messages. We provide the definition of an HE cryptosystem and its properties as follows.

Definition 1 *Let $\mathscr{C}$ and $\mathscr{M}$ be the spaces of clear and encrypted messages, respectively. An homomorphic encryption(HE) cryptosystem $HE = (KeyGen, E, D, Eval)$ with $\mathscr{C}$ and $\mathscr{M}$ consists in four polynomial time algorithms:*

- $KeyGen[1^\lambda]$ the key generation algorithm that takes one input, a security parameter λ in order to output a key pair (K_p, K_s), where K_p is the public key used for encrypting data and K_s is the private key used for data decryption.
- $E[m, K_p]$ the encryption algorithm which with as inputs the encryption key K_p and a plain-text $m \in \mathscr{M}$ and as as output the cipher-text $c \in \mathscr{C}$.
- $Eval[h, c_1, \cdots, c_n, K_p]$ the evaluation algorithm the inputs of which are the public key K_p, an evaluation function h and a tuple of inputs that can be a mix of cipher-texts and previous evaluation results. It produces an evaluation output which can be decrypted to get access to the plain-text.
- $D[c, K_s]$ the decryption algorithm which produces a plain-text m based on the secret key K_s and a cipher-text.

The first attempts for defining a homomorphic cryptosystem [125, 128–139] have enabled either one type of operation or a limited number of operations on the encrypted data. In addition, some of theses methods are even limited over a specific type of set such as branching programs. One can distinguish three types of HE cryptosystems with respect to the number of operations that are allowed on the encrypted data as follows:

- **Partially Homomorphic Encryption (PHE)** allows performing only one type of operations with unlimited number of times, these operations being multiplication (e.g., ElGamal cryptosystem [130]) or addition (e.g., Damgård-Jurik [140]). Since the pioneer work by Rivest *et al* [125], several useful PHE cryptosystems have been proposed [128–139]. They are deployed in various applications including electronic voting [141] or Private Information Retrieval (PIR) [142], but these applications are restricted in terms of the types of homomorphic operations that are allowed. In other words, a PHE cryptosystem can only be used for a particular application, whose algorithm include only addition or multiplication operations. Each of these cryptosystems has improved the PHE in some way. We give some examples that are the basis for many other PHE cryptosystems.

- **RSA** (from its authors Rivest, Shamir and Adleman) is the first asymmetric cryptosystem and the first PHE that was proposed [125]. It was proposed shortly after the invention of public key cryptography by Diffie and Hellman [143] and it is the first feasible achievement of the asymmetric cryptosystem. Moreover, it is the first cryptosystem the homomorphic property of which has been introduced by Rivest *et al* [144]. It is a multiplicative PHE, i.e. the product of the RSA encrypted messages allows the computation of the product of the clear messages. Currently, RSA is considered one of the most solid PHE cryptosystems. This is why it is still being used for protecting critical data exchanges. Its security of the RSA is based on the hardness of the factorization problem of two large prime numbers.
- **Goldwasser-Micali (GM):** GM is the first probabilistic asymmetric encryption cryptosystem proposed by Goldwasser and Micali in 1982 [128]. The GM cryptosystem is based on the hardness of quadratic residuosity problem [145]. Notice that a number a is called quadratic residue modulo n if there exists an integer x such that $x^2 = a \mod n$. The quadratic residuosity problem decides whether a given number y is quadratic modulo n or not. The GM is the first cryptosystem that has introduced the semantic security. This means one clear message can have different encrypted messages. This property is important and is generally obtained by taking into account a random number in the encryption function E. The homomorphic property of this cryptosystem shows that encryption of the sum or XOR of two clears messages can be directly obtained by computing the product of corresponding encrypted values. In addition, as the clear message and encrypted message are binary messages, the operation is the same with XOR and GM is then homomorphic over only addition for binary numbers. This cryptosystem was extended in many cryptosystems such as Benaloh [136], Okamoto-Uchiyama [133] or Naccache-Stern [132] with the goal of increasing the computational efficiency.
- **ElGamal:** This cryptosystem was proposed by Taher Elgamal in 1985 [130], and it is the improved version of the original Diffie-Hellman Key Exchange algorithm [143]. The security of ElGamal is based on the hardness of the discrete logarithm problem [146]. It is mostly used in hybrid encryption systems for encrypting the secret key of a symmetric encryption cryptosystem. ElGamal is multiplicative PHE as the product of two encrypted messages allows to the computation of the product of these messages in clear form.
- **Paillier:** This is another novel probabilistic encryption cryptosystem that was proposed in 1999 by Pascal Paillier [135]. It is based on the composite residuosity problem [147], which is the generalisation of the quadratic residuosity problem [145] used in GM cryptosystem. Paillier cryptosystem has additive homomorphic properties, i.e., the product of two encrypted messages allows the computation of the sum of their corresponding clear messages. This cryptosystem was extended in the Damgård-Jurik cryptosystem [139].

Other PHE cryptosystems have been proposed, with the objective of improving previous cryptosystems and preserving their homomorphic properties, or using new techniques [137].

- **Somewhat Homomorphic Encryption (SWHE)** allows several types of operations in a limited number of times. Indeed, if these cryptosystems support for example addition and multiplication operations, the size of the encrypted data increases after each homomorphic operation, and this limits the maximum number of allowed homomorphic operations (e.g., BGN [136]). SWHE properties were observed for some PHE cryptosystems [129, 134, 148] but it was however difficult to use these schemes for encrypted data processing because the increase of size of the encrypted data. For instance, the scheme proposed by Fellows and Koblitz [148] allows both addition and multiplication operations over encrypted data. However, the size of the encrypted data grows exponentially with the homomorphic operations, and the multiplication operation is especially extremely expensive [148]. It was not until 2005 that the first SWHE scheme appeared. The first cryptosystem of this type is the BGN, named after its authors Boneh, Goh and Nissin [136]. This scheme supports an arbitrary number of addition operations but allows only a single multiplication by keeping the encrypted message size constant. The security of BGN is based on the subgroup decision problem [149] which consists in deciding whether a given element is a member of a subgroup G_s of the group G of composite order $n = pq$, where p and q are distinct prime numbers. This cryptosystem was the first significant step towards to an FHE scheme. Other SWHE schemes have been proposed [138, 150–152] with the same objective which consists of finding one day a FHE scheme. In general, during homomorphic evaluation, especially multiplicative evaluation, SWHE cryptosystems add noise in encrypted data. Once the noise exceeds a certain threshold, it will no longer be possible to correctly decrypt it. To overcome this issue, several techniques such as bootstrapping have been proposed [153]. Thus, a fully homomorphic encryption can be obtained.

- **Fully Homomorphic Encryption (FHE)** enables an unlimited number of operations for an unlimited number of times (e.e., BGV [154]). There have been several attempts to build such cryptosystems, but it was not until 2009 to see the first plausible construction of FHE scheme presented by Gentry in his Ph.D thesis [153]. It is based on ideal-lattices, and its use in practice is not feasible. The idea of gentry is to build a FHE cryptosystem from a SWHE cryptosystem by introducing some techniques that allow reducing the noise when it becomes important, during the homomorphic evaluation and one of these techniques is bootstrapping. Gentry's work leads not only to an FHE scheme based on ideal lattices, but also to a generalized theoretical and powerful framework for defining a FHE scheme. However, this solution has several limitations such as the very high computational complexity due to the fact that it is based on an ad hoc problem and a spare subset sum problem (SSSP) problems. Thus, this scheme cannot meet the requirements of practical applications but it give to researchers many ways that have permitted the designing of secure and practical FHE schemes after Gentry's work. There are four classes of FHE schemes:

  - The first class consists of the FHE schemes that are based on ideal lattices, i.e., based on the initial Gentry's scheme [155–158]. These schemes use smaller cipher-text and key sizes than Gentry's scheme without reducing the security. In addition, some of them have focused on the optimizations in the key generation algorithms in order to increase the FHE efficiently.

- The second class corresponds to FHE schemes that work on integers [159–163]. The security of these schemes is based on problems such as the Approximate-Greatest Common Divisor (AGCD) [164]. This problem consists on trying to find an integer p from a set of equations $pq_i + r_i$. These solutions are efficient compared to ideal lattice-based schemes. However, their security is based on weak problems.
- The last class consists the FHE schemes that are based on learning with errors (LWE) or ring learning with errors (RLWE) problems [154, 165, 166], these schemes show better performances compared to previous ones.

Today, solutions of the last class are the more efficient in terms of complexity, size of the encrypted messages and security. In this thesis, we were particularly interested in the BGV (from its authors Brkerski, Gentry and Vaikuntanathan) [154], a FHE scheme that was implemented by IBM in HElib library [167]. In this library, several significant optimizations such as re-linearization, bootstraping, squashing , batching, etc. have been considered. We will return to this scheme in chapter 2 where we have proposed BGV-based solution for protecting outsourced GWAS.

B. Secure Multiparty Computation

Even though homomorphic encryption allows the protection of data confidentiality as well as the encrypted data processing, it does not allow all operations to be carried out on encrypted data, in particular no-linear operations such as comparison and division. Secure multiparty computation (SMC) comes as a solution for performing these types of treatments. By definition, SMC allows a set of different parties or participants (at least a client and a server) to securely evaluate a function on their private data as inputs in such a way that no information other than an agreed upon output or result is available to the parties. This result which is known to everyone can be, for example, a Boolean, or the index of the closest element in the database can have various applications including privacy-preserving decision making on distributed genetic or financial data, online poker, private set intersections, privacy-preserving machine learning, etc. Proposed SMC solutions can be classified into two main categories accordingly the number of parties they support: *1)* secure two-party computation and *2)* secure multi-party computation. Different cryptographic techniques All these techniques can be used in order to realize a SMC scheme. Three common underlying techniques for these schemes are *a)* Oblivious Transfer, *b)* Yao's Protocol or garbled circuit evaluation, and *c)* Secret sharing.

- **Oblivious Transfer (OT):** This protocol introduced by Rabin 1981 [168] is among the fundamental tools for securing data, especially in cloud environments. OT can be introduced as follows. Considering that we have two parties a sender that knows two secrets S_1 and S_2, and a receiver who want to know one of these secrets, but he does not want the sender to know which one. More generally, let's consider an OT algorithm allows a receiver to obtain an element S_i in a set of T elements $S = \{S_1, S_2, \cdots, S_T\}$ from the sender; without knowing any other element in the set, and without revealing S_i to sender. To do so, the receiver choose an index i that corresponds to the element that he/she want in S, and this

index is used to retrieve S_i in S. At the end of the protocol, the receiver receives S_i, without knowing any other S_j, for $i \neq j$, and the sender does not know i.

- **Yao's Protocol (Garbled Circuit evaluation):** This protocol presented by Andrew Yao in 1986 [129] allows two parties or entities to collaborate and correctly compute a function before sharing its output, and without knowing the input of each entity to another. Yao introduces this protocol in order to give the response to the millionaires' problem. Herein, two millionaires want to determine who is the richest between them, without revealing their respective fortune. Yao modeled this problem as a series of binary gates that take as input encrypted data. Encryption operation can be conducted using classic symmetric encryption algorithms such as AES or 3-DES. Even though this solution is theoretically interesting, it remains useless due to its computation complexity. After Yao' scheme, several solutions have been proposed using in particular homomorphic encryption. The latest approaches model the function as a boolean circuit which is shared between the involved entities. At each gate of the circuit, input or output data is encrypted so that the entity which evaluate the function or part of the processing cannot extract any information about the inputs or the intermediate values.

- **Secret sharing:** Introduced by Shamir [169] in 1979, it represents the set of methods in which a secret is provided to several parties, so that the reconstitution of the secret requires the collaboration of a certain number of these parties. Any entity can not get access to the secret on its own. Formally, More formally, a secret sharing protocol between T participants or entities with threshold k such that: *i)* Any k participants or more, chosen in a set of T participants can always allow recovering the secret; and *ii)* any $t-1$ participants chosen a set of T participants can never allow the recovering of the secret. The solution proposed by Shamir is based on polynomials of degree k and Lagrange polynomial interpolation for distributing data (secret) to T participants. Assuming that the secret is the value s, a finite field is chosen so that the secret s is the size of an element of this finite field. For instance, a 64-bit secret gives the field $K = \mathbb{F}_2^{64}$. Thus, in order to retrieve the secret by at least k participants among T, one must choose a polynomial f on $K[x]$ of degree $k-1$ such that $f(0) = s$

$$f(X) = s + \sum_{i=0}^{k_1} a_i X^i, a_i \in \mathbb{F}_2^{64} \tag{1.2}$$

  Each value $f(b_i)$ is distributed to each participant, with b_i are distinct values and different to zero. Therefore, if k participants collaborate, then they are able to reconstruct the polynomial f of degree $k-1$ and recover the secret s, thanks to the Lagrange polynomial interpolation. If fewer than k participants collaborate, they will construct a polynomial but with different constant. As the secret corresponds the constant s from the polynomial of degree $k-1$, they cannot find any additional information on the secret. Secret sharing could have different applications such as the protection of the decryption key of a given cryptosystem, which requires the collaboration of many parties in order to conduct data decryption.

C. Cryptographic hash functions

A cryptographic hash function is defined as a cryptographic algorithm that takes as input an arbitrary amount of data, and produces a fixed length output called a hash value, or just "hash". This value can then be stored instead of the password itself, and later used for various applications including data integrity verification, pseudo random number generation, password verification or message authentication. An important property of such functions is that they are irreversible functions or one way functions. This means that it is infeasible to get an idea of the input of the function from its hash value. The hash computation can also be conducted using secret hash key which is associated to original data. For instance, in message authentication code (MAC) mechanisms [170], a message is concatenated to a secret hash key K_h. Thus, the secret hash value a_h of a given message m is given by

$$a_h = hash(m||K_h) \tag{1.3}$$

where $||$ is the concatenation operator, and *hash* represents a secure hash mechanism such as secure hash algorithm SHA1, MD5, SHA256 or SHA3 [105]. For each data of any size, these schemes provide a hash value encoded on l bits and the common choice of l is 160, 256, 256 and 512 bits. Each cryptographic hash function should be indiscernible from any random function with the same parameters and it should fulfil the following four properties:

1. **Efficiency** which means that it is easy for each message m, to compute the corresponding $h(m)$;
2. **Collision resistance** which means that it is extremely difficult to find two distinct messages m_1 and m_2 such that $h(m_1) = h(m_2)$, and a such possibility should requires at least $2^{n/2}$;
3. **Preimage resistance** which corresponds to the fact that, for a given a hash value a_h, it is hard to find a message m such that $h(m) = a_h$. The time complexity of a single preimage attack is at least $2^{n/2}$;
4. **Second preimage resistance** which means that if m_1 is given message, it is hard to find a second message m_2, such that $h(m_2) = h(m_1)$. The time complexity of a such second preimage attack is between $2^{n/2}$ and 2^n. For example, for any data of maximum 2^{64} bits, *SHA256* provides hash value encoded on 256 bits.

For any hash function mechanism, the probability two messages lead to the same hash value is $\frac{1}{2^{128}} \approx 2.9 \times 10^{-39}$. In this thesis, we will come back to the use of SHA256 in chapters 3, 4 and 5 where it is used in ensuring data confidentiality, databases partitioning during watermarking and computation of the watermark.

D. Watermarking as a complementary security mechanism

Previous mechanisms offer an *"a priori"* protection. On the contrary to these methods, watermarking provides an *"a posteriori"* protection, as it allows the access to the data while keeping

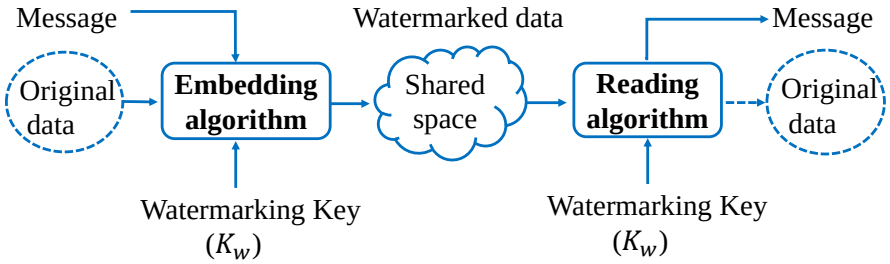

Figure 1.8: Main stages of a common watermarking chain. In this chain, we consider that the watermarked data is shared (e.g via the Internet) and it can be illegally modified or manipulated between the embedding and the reading stages. At the reading stage, the inserted message readed and/or extracted, and in the case of reversible watermarking, the original data can be fully recovered.

them protected by a message (watermark) intrinsically linked to it. In the sequel, we come back on general watermarking fundamentals, in particular database watermarking.

D.1 Definition

Digital watermarking is a technique that consists in the imperceptible embedding of an extra-information (watermark or message) within a digital contents (e.g., video, image, etc.), usually called host data, without perturbing its normal use or interpretation.

For instance, the watermark is embedded into an image by imperceptibly modifying its gray values. Watermarking uses the same principles as steganography, a discipline just as old as cryptography but with different objectives. With steganography, the host document data does not have any importance. The objective of the user is to conduct a secret communication using the host document as a covering channel. This is to say that the objective is to ensure the security of the inserted message, and this one must be completely imperceptible and undetectable. The objective of watermarking is the protection of the host document by using the embedded message, contrary to steganography. Watermarking was originally proposed in the early 90s in order to ensure the copyright protection of multimedia contents such as images [171]. To do so, a watermark that contains the owner identity is embedded into the image, and is used as copyright information. The buyer identity can also be inserted in order to ensure the traceability of the image. The inserted watermark should be in this context resistant to attempts of an attacker who want to erase or modify the watermark. Since then, watermarking was extended to several security objective such as copy protection, integrity verification, etc; and to may types of data such as databases.

D.2 Principles of watermarking

A classic watermarking chain deeply resembles a communication system and they have both the same objective which is the transmission of a message. In the case of digital watermarking, the noisy communication channel corresponds to the host content and the available bandwidth is rep-

resented by the number of bits of message one can embed. As illustrated in Figure 1.8, each watermarking chain is conducted based on two main processes that are message embedding and detection and/or extraction of the message.

- **Message embedding:** This process allows the insertion of a message (watermark) in the host content such as an image or a database. This insertion is performed by altering, modulating or modifying as imperceptible as possible of the host content under the principle of controlled distortion. For instance, an image watermarking is done by conducting the modification or modulation of gray levels of image pixels or of coefficients of a transform of this one such as DCT (Discrete Cosine Transform), TFD (Discrete Fourier Transform) or DWT (Discrete wavelet transform). For databases, watermarking can conducted by modifying database attribute values or by dealing with the order of tuples in the database. The message embedding process depends a secret watermarking key K_w which allows for instance the selection of attribute values to be watermarked, the construction of the watermark itself during the watermarking of a database or the selection of the pixels or the coefficients to be watermarked in the case of image watermarking.

- **Detection and/or extraction of the message:** This process depends on the secret watermarking key. The embedded watermark can completely be extracted or simply detected depending on the application the watermark. In addition, in some cases it is possible to invert the introduced modifications and to recover the original content. This latter is known as lossless or reversible watermarking. One distinguish a blind detection/extraction or not. A watermarking scheme is said to be blind if it does not require the presence of the original data to extract the message, semi-blind if it requires some information from original data, or no-blind otherwise. Notice that the watermarked data can be subject of authorized manipulations or attacks (innocent and malevolent) in between the embedding side and the reader that could erase, weaken or modify the inserted message. The former is referred as innocent attacks, while the latter is malevolent. The capability of a watermarking scheme to resist such an attack corresponds to the concept of robustness, and the length of the message that one can insert in host data corresponds to the watermarking capacity. We will come back to these properties in the next section.

D.3 Properties of watermarking systems

Each watermarking scheme should be characterized by different properties regarding the given application. However, it is difficult to satisfy all these properties and a compromise has to be established between them. In this section, we describe the existing main properties:

- **Robustness:** A watermarking scheme is called robust if after processing operations or maliciously attacking on the watermarked data, the inserted watermark is still accessible [106]. For instance, for images, many processing including data compression, color correction, noisy transmission, addition of captions or geometric modifications can be conducted. These operations are known as innocent attacks. On the other hand, during their transmission or

distribution across the internet, watermarked data may face several attacks the purpose of which is to remove/alter the watermark for the illegal use of the watermarked data. As we will see in chapter 5, we have proposed a robust watermarking scheme that must allows different genetic processing such as genome-wide association studies without compromising test results.

- **Watermarking capacity:** The watermarking capacity is defined by the maximum amount of information that can be embedded within a specific content. In the case of genetic data watermarking, it can be measured in bpn, that is to say in number of bits that can be inserted in one nucleotide base or bits per codon (bits/codon) [12]. Under the condition of imperceptibility as well as the requirements of robustness, the watermarking capacity relies on the size of the original data. The more original patterns are attainable, more information is able to be inserted. However, embedding as much watermark information as possible is a more difficult task in digital watermarking. In addition, depending on the application of watermarking scheme, capacity can be less considered.

- **Imperceptibility:** It is an essential property for digital watermarking as the visual similarity between the watermarked version of data and original one, and the perceptual quality of the original data should be transformed imperceptibly by the embedding process. Even though by definition the watermark must be imperceptible or invisible, sometimes watermarking schemes embed a visible watermark. This could be for example the embedding of related to ownership into the original content (e.g., image) in a perceptible manner, so visible watermarking can perform copyright protection in more direct and immediate manner than invisible watermarking [172]. This type of solution is at the limit of watermarking but is considered as such because of the degradation of the host data. There are two main reasons why it is important to keep the imperceptibility of the host data after the watermark embedding. Firstly, the absence or the presence of a watermark cannot be distinguished from the primary purpose of the original data, if the watermarked data is so badly distorted that its value is lost. Additionally, suspicious perceptible artifacts may introduce a watermark in existence, and perhaps its precise location being detected from host data. This information may help the attacker to access to the watermark and perform different illegal operations such as substitution or removal of the watermark. Therefore, the information embedded in it may no longer be available.

- **Reversibility:** Introduced in 1997 by Mintzer *et al.* [173] for image watermarking, this property allows the extraction of the watermark and the restoration of the original host data from their watermarked version by inverting back modifications induced during the watermarking process. This property is often desired in different applications such as healthcare or genomics/genetics where the quality of data is a strong constraint. For instance, several schemes have demonstrated that this property can be used in integrity control of data where, one can insert a digital signature computed for the whole host document or data [174].

- **Complexity:** This property corresponds to the indication of the required computation time for the watermark embedding and watermark detection/extraction processes. In some applications such as video on demand (VOD), the insertion is not needed in real time, but at the

reading stage, the no detection or a delay in detection may cut off the broadcasting process. Thus, embedding and detection/extraction complexity constraints can also be determinant in some application frameworks.

- **Security:** This property can be defined as making it very difficult for attackers to extract or remove the watermark and its content as well as its modification (falsification), or to embed a new one so as to hide the original watermark. Thus, for each watermarking scheme, the access to the embedded watermark must be restricted, generally by means of a secret watermarking key K_w that allows only authorized users to extract the embedded information. As we have seen in previous sections K_w can also used for generating the watermark. Without this key, it should not be possible to find the watermark or generating a valid watermark. For some watermarking methods, the message is encrypted before being embedded in order to improve its security [175]. Thus, even though an attacker can extract the watermark information, it will still be difficult for him/her to get access to the watermark without knowing the decryption [176]. It is important to note that this property is directly related to the notion of robustness as the suppression of the watermark resulting the useless of watermarked data.

Nowadays, it is not possible to offer all of these properties simultaneously, and there is no watermarking scheme that can ensure all of them. However, in practice, the requirement relating to each of these properties varies according to the application context (e.g., integrity control, copyright protection, tracking of illegal copies, etc.). A watermarking method will be chosen according to the compromise that it establishes between these different properties.

D.4 Applications of watermarking schemes

Digital watermarking is potentially useful in many applications depending on the relationship between the host data and the embedded watermark, or the document or data to watermark. There are several applications of watermarking including: **copyright protection**, **traitor tracing** and **integrity verification**.

The first proposed and most-studied application of digital watermarking is **copyright protection**. It was initial proposed for multimedia contents before being extended to other types of data such as databases. Copyright protection relies on the embedding of an identifier which associates the host document or data to its owner (creator or buyer) [177]. This identifier that corresponds to the watermark should be imperceptible and resistant to any operations, especially those conducted by attackers in order to damage or remove the watermark. For instance, in association with Adobe, Digimac developed a tool which is available in Photoshop software and allows the copyright protection control for images. In fact, when the this tool recognizes a watermarked image, it refers to a centralized database which is accessible online, and uses the inserted watermark as a key message in order to find the identity of the owner of the image [178]. In many cases, the copyright assertion is conducted using two steps: the detection step that consists of verifying the presence of the watermark and the extraction step that allows the identification of the owner. Thus, watermarking offers more practical and autonomous solutions than classic solutions that are based on the registration of the document (e.g., image) to a trusted third party who keeps a coy of the

original document. In addition, it is not easy to implement this solutions in case of databases or software because of their important size which can cause the storage complexity overhead in when keeping their copies. Notice that the first database watermarking scheme, proposed by Agrawal and Kiernan [179], focused on copyright protection. It is also possible to use the watermarking for identifying the recipient of the one content or for tracing the historical of its possible illegal distribution. This is referred to as content **traitor tracing** or fingerprinting [180]. To do so, data owner embeds different watermarks in each distributed copy of the content using an identifier or fingerprint which uniquely identifies an individual. If one of the receivers decides to illegally reroute or redistribute the content, it becomes possible to identify him or her [181]. These solutions are designed in the way that they must be resistant to collusion attacks. In these types of attacks several users owning copies of the same content cooperate together in order to obtain the original version of the content [182]. Tardos codes offer an interesting compromise between the length of fingerprinting (in bits) and the detection efficiency of at least one attacker from such a coalition [183]. Traitor tracing solutions can also be used for identifying a dishonest user who is the origin of data leakage. As previously exposed, the user identifier is inserted when he/she accesses the data. If the information is retrieved elsewhere on internet, it will be possible to identify the the individual which is responsible of this diffusion by extracting the watermark. Contrary to the previous problem, collusion attacks are of less concerns as such data leaks are usually the result of one user.

The last but not the least application of digital watermarking is **integrity verification**. Indeed, it is essential to ensure data integrity, especially when they acquire a legal value or if they contribute to sensitive decision making. That is especially the case of the genomic domain where genetic data are very sensitives. For instance, genetic variants are highly used in different genetic analysis, and their illegal modifications may have several consequences to an individual health, their relatives, etc. Thus, their integrity must be preserved, and watermarking comes as solution [184]. As previously said, it possible to control the integrity using reversible watermarking. To do so, the integrity of a content such as database is controlled by computing the digital signature of the content which is then embedded in the content. In the detection process, the signature is extracted from the content and is compared to the one computed on the recovered content. Any difference between them will indicate if the content have been illegal modified or not. Other watermarking solutions that are used are the so called solutions fragile or semi-fragile schemes. In opposition to robustness, the fragility of the watermark to contents' manipulations can herein be useful. During the verification, the absence or the incorrect detection of a watermark will indicate a data integrity loss. Depending on the application context, the watermark can be designed for resisting to some specific manipulations but not to all. If all modifications can be detected, we will talk about fragile watermarking [185]. Such solutions are usually very sensitive, like a digital signature or message authentication code, and in some cases, they can indicate which parts of the content that have been illegal modified [186]. On the contrary, a semi-fragile watermark are designed to be robust to some innocent manipulations, that are allowed but fragile to malicious manipulations [174].

D.5 Database Watermarking

Digital watermarking has been initial proposed for multimedia data [187] but since the 2000s, it has gain an, interest in protecting databases. In fact, databases represent today great economical and strategic concerns for both enterprises and public institutions. In that context, data leakage, robbery as well as innocent or even hostile data degradation represent a real danger, and watermarking comes as an interesting mechanism for databases [188]. Herein, we give an overview of watermarking schemes that have been proposed for protecting relational databases.

By definition, a database is a structured set of data, stored on media which accessible through a computer in order to satisfy several users simultaneously. A relational database is a database organized accordingly to the relational model which is based on the notion of relationship as the mathematical representation of a set of data. Formally, a relational database DB is a finite set which is composed by a list of T tables or relations $\{R_i\}_{i=1,\cdots,T}$. Each relation is made of N tuples $\{t_1, t_2, \cdots, t_N\}$ and one tuple corresponds to M attributes $\{a_1, a_2, \cdots, a_M\}$. An attribute a_j takes its values in a specific domain which can be categorical or numerical. In a database, the value $t_i.a_j$ represents the j^{th} attribute of the i^{th} tuple in the relation, and in one relation, the value $t_i.PK$ represents the unique identifier of attribute values of the tuple t_i. This value is called a primary key. Although watermarking emerges as a promising complementary mechanism for database security, the use of existing methods that have been proposed for multimedia data (e.g., image, video, etc.) is not a straightforward process. This is due to the fact that relational databases differ from multimedia data in several aspects, and these differences must be taken into account during the conception of a new database watermarking scheme. For instance, in a multimedia document such as image are sorted into a specific order, in a temporal and/or spatial domain (e.g., pixels of an image), and this gives a sense of the document itself to the user. Contrary to the multimedia data, data stored in relational databases are independent elements within a common structure. Thus, tuples or records in one relation can be stored without any specific order and can be reorganized in many ways in a relation without impacting the database information. Thus, a database watermarking must consider all their particularities compared to multimedia data.

Several methods have been proposed for watermarking databases, and these methods can be classified according to many criteria. First, methods are classified based on the fact that the database to watermark is encrypted or not. The second level of classification is based on their robustness against attacks. they are robust methods that are developed for traitor tracing applications and fragile/semi-fragile methods most of them were designed for integrity control applications. Another criteria of classification can based on the way how these methods deal with data distortion. Thus, there are methods without or with distortion control, distortion free methods and lossless or reversible methods. Notice that all database watermarking methods exploit either numerical or categorical data.

Regarding database watermarking methods in clear, a pioneering work was proposed by Agrawal *et al* [179]. In this robust method, watermark embedding is conducted using bit substitution in the least significant bits (LSB) of attribute values. Database elements to be modified (tuples, attributes and bits) are secretly selected by means of a hash function. In this method, the embedded watermark depends of the database content and it is not known by the user, i.e., it corresponds to a database identifier. During the detection process, if the database has been watermarked, the

expected number of bit correspondences in secretly selected positions should be near to 100%. In the case the database has not been watermarked, this number logically falls down to 50 %. Since then, many other methods have been developed with as interest traitor tracing, fingerprinting or copyright protection applications through the insertion of watermarks that are robust to database modifications, these ones being illegal or not [106, 189–192]. As introduced before, these methods can also be classified depending on how distortion is controlled. Thus, there are distortion-based methods where a watermark is embedded in the database by modifying database elements [179] or introducing "fake" tuples in the database [193], and these modifications may satisfy distortions constraints or not; distortion control-based methods where a watermark can be inserted in the database by using database statistics, and the distortion is controlled [192]; or reversible methods where a watermark is embedded by modifying database attribute values [189] or by spaces between database contents [194], with the constraint that it is possible to reverse the modification operations and recover the original database from their watermarked version. For instance, the method presented by Gupta and Pieprzyk [195] a lossless watermarking method where a meaningless pattern is inserted into secretly chosen tuples. To do so, a LSB of an integer part of a numerical attribute value is secretly chosen and is replaced by another bit which is pseudo-randomly generated. The original value is then inserted into the space left by right shifting the LSB representation of the fractional part. The presence of this pattern is checked by the detector, indicating if the database has been watermarked or not.

In parallel, fragile methods have been designed [174, 185, 186, 196–205]. Contrary to robust ones, these methods allow the embedding of a "fragile" watermark which will be damaged even by minor database modification is performed. They have been especially proposed for integrity verification. Some of them allow the localization of the database elements (e.g., tuples) that have been modified [197]. There are two categories of fragile schemes: distortion-free schemes and lossless or reversible schemes. Instead of modifying database elements, distortion-free methods encode the watermark into new data, such as in some "virtual" attributes' values [186, 198], or by dealing with the ordering of the database elements (i.e., tuples or attributes [196], [185]). The first distortion-free method was presented by Li *et al.* in 2004 [196], and it does not modify the attribute values. In fact, in this method, the database is first divided into several groups. Then, tuples are grouped and ordered in each group accordingly the value of a hash function which is computed on the attribute values concatenated with the primary key and the secret watermarking key of the owner. The watermark to embed for a group G_i is a sequence W_i of length $l_i = \frac{N_i}{2}$ with N_i the number of tuples in the group. The watermark embedding consists in altering the order of pairs of tuples in the group depending on the bit to insert. During the detection process, if the same order of tuples is not obtained, the database is considered as illegally modified. In [198], in order to insert a watermark, one or several virtual attributes of NULL values are added to the database before dividing the database into groups of tuples. Then, the watermark embedding works as follows. In a group, the values of one of the virtual attributes are substituted by the aggregate values (e.g., the sum, the median or the mean value) of some other chosen numerical attribute values. For one tuple, the checksum [206] of each attribute is computed and concatenated to the virtual attribute value. The detection process and integrity control follows the same procedure. The integrity of the database is only verified when the recomputed checksums correspond to the extracted

ones. Regarding fragile and lossless methods, they are well adapted for ensuring integrity control of databases. In general, they embed a digital signature of the database into itself. During the verification stage, the digital signature is extracted and compared to the one computed from the reconstructed database. This kind of methods relies on the difference expansion watermarking modulation [200] or on histogram shifting [174]. They work on numerical attributes [200] or on categorical attributes [174]. For example, the method proposed by Coatrieux *et al.* [174] is the first fragile and lossless watermarking method that works on categorical data. This method is an adaptation of the histogram shifting modulation to categorical data. In order to embed the watermark, database tuples are divided into many groups and each group is partitioned into two sub-groups $SG1$ and $SG2$. The number of appearances of the attribute values of the in the sub-group $SG1$ are used in order to construct a virtual dynamic, i.e., an order relation between different values that the attribute can take. The elements of the sub-group $SG2$ serve to the insertion and the histogram shifting modulation is applied considering the virtual dynamic constructed from $SG1$. The elements that belong to the class with the highest cardinality are considered as carrier elements. The others are shifted to the right in order to create a free bin. Carrier elements are then shifted or let unchanged depending on the bit to insert, '1' or '0', respectively. The inserted watermark can be a signature of the database that can be used for integrity verification.

Regarding watermarking methods for encrypted databases, few methods have been proposed [207, 208]. The idea is to protect data confidentiality using encryption mechanisms and ensuring while other security services such as integrity control using watermarking. For instance, the method proposed in [207] focuses on the protection of outsourced databases by ensuring the confidentiality of databases using Order Preserving Encryption (OPE) [209] and integrity verification of encrypted databases using watermarking.

All above solutions have several limitations. First, all of them work on databases that are not updatable i.e., static databases. In general, these methods consider database updates as unauthorized modifications. Thus, in the case the database one user adds, deletes or updates some tuples, the whole database has to be re-watermarked. This is also the case of proposed methods for encrypted databases. Regarding illegal modification, distortion-free methods can localize altered elements but without a really good precision (i.e., tuple level at best). Lossless methods allow us to know if the database has been modified but that is all. In addition, for encrypted database watermarking [207], encryption operations are conducted using an OPE encryption that is known for its security limitations due to some of its deterministic properties [210]. In this thesis, we overcomes these limitations by proposing in chapter 3, a dynamic database watermarking solution that allows the watermarking of homomorphically encrypted data. Contrary to these solutions, it allows database watermarking while making possible update operations such as addition, suppression or modification of database elements. This solution was particularly proposed for the protection of outsourced genetic data.

1.2.2 Privacy-preserving of genetic data

In this section, we give an overview of security solutions that have been proposed for ensuring privacy and security of genetic data, during their sharing, storage and computation. They are

based on different security mechanisms including homomorphic encryption (HE), secure multi-party computation (SMC), differential privacy (DP) and secure cryptographic hardware (SCH). We classify these solutions four categories depending their application: methods that were proposed for securing count queries on genetic data (QGD); methods which are used for securing genetic sequence comparison (GSC) and matching; methods that were developed for ensuring the security of personal genetic testing (PGT); and methods that are used for protecting genome-wide association studies (GWAS) and statistical analysis.

1.2.2.1 Secure count queries on genetic data

Several querying operations (e.g., SNP, allele or frequency counts) on large genomic databases are among fundamental building blocks for genetic analysis such as GWAS, personal genetic testing, etc. For example, a disease susceptibility analysis is usually done by querying a patient's genome against a list of known variations and then predicting this disease susceptibility. In addition, many genomic databases such as gnomAD (the genome aggregation database) were constituted in order to respond researcher's queries about variants or frequencies about individuals from different sources [211]. Thus, these operations present an opportunity for genomic databases whose data are from multiple sources and different jurisdictions, which otherwise cannot be publicly shared due to security issues presented in section 1.1.6.

Different solutions have been proposed for securing count queries on genetic data (QGD). One of the earlier attempts to securely compute count queries on outsourced genomic databases is a solution proposed by Kantarcioglu *et al.* [212]. Authors proposed a framework which involves two third parties. One party is responsible for integrating homomorphically encrypted data coming from different data sources and then executing queries on behalf of a researcher on those data. Then, the result of the query is transmitted to another party, a key holder site who is responsible for encryption key management. This key holder site conduct the decryption of the result and produce the final result and send it to the researcher. This method has several limitations such as the query execution time which is quite large. As stated in their paper, it takes around 30 mins to execute a count query over 40 SNP variants in a database of 5000 tuples. Thus, this method may not be suitable for big databases that contain millions of tuples. The use of homomorphic encryption produce large encrypted databases which require large storage spaces. To improve the efficiency of this solution in terms of communication, computational and storage complexity, Canim *et al.* [213] presented a new method that makes use of a symmetric encryption combined with cryptographic hardware. Indeed, their framework combines a secure cryptographic coprocessors (SCP) and Advanced Encryption Standard (AES) [123] so as to perform count queries on joint genetic databases securely.

Other solutions that secure count querying have been proposed [214–216], and their objective as to securely perform count queries while minimising communication, computational and storage complexity. Table 1.3 represents the comparison of these methods.

1.2.2.2 Secure genetic sequence comparison and matching

In bioinformatics, sequence comparison is one of the most fundamental techniques that are used for analyzing similarities or homologies in between DNA sequences. It is used for motif finding, gene finding or sequence alignment which is used for evaluating the optimal cost of insertions, deletions and substitutions of bases (A, C, G and T).

Different methods have been proposed for performing secure genetic sequence comparison (GSC), and most of them are based on homomorphic encryption and secure multiparty computation. Proposed methods protect the well-known sequence comparison algorithms such as dynamic programming methods (e.g., Smith–Waterman algorithm), word methods (e.g., BLAST) and their variant. For instance, Atallah and Li [217] proposed a privacy-preserving method that allows the computation of the edit distance between two DNA sequences based on dynamic programming. This method requires two non-colluding servers, each of them possessing one input sequence, to engage an interactive process. In order to exchange the results of computation from the servers in each iteration, a secure look-up table is introduced. As the number of iterations is the product of the two input sequence lengths, computation and communication complexity is overhead. The method presented in [218] improves the computation efficiency of [217] allows two different parties the computation of the edit distance between two DNA sequences such that neither party learns anything about the private DNA sequence of the other party except the comparison result. However, the communication complexity is still the same.

Notice that other methods have been proposed in order to secure DNA sequences comparison [219–222], and sequence comparison is amongst the widely covered areas in the implementation of privacy-preserving genetic methods. However, most of them are based on SMC and HE algorithms the complexity of which is no negligible.

1.2.2.3 Secure personal genetic testing

Genetic testing consists the examination of variations in chromosomes, genes and proteins between an individual's genome in order to conduct disease susceptibility, identity, paternity, genealogical or compatibility test. Compatibility test enables a pair of individuals to evaluate the risk of conceiving an unhealthy or healthy baby. In this case, methods based on private set intersection have been proposed [223] so as to conduct the computation of genetic compatibility, where one individual submits the fingerprint for his or her genome-based diseases, while the other individual submits her or his entire genome. By doing so, the couple learns their genetic compatibility without revealing their entire genomes. Another test that can be conducted is paternity testing which determines whether a male individual is the father of another individual. This test is based on the high similarities between the genome of the father and his or her child (99.9%) comparing to two unrelated individuals (99.5%). It is not known exactly which 0.5% of the human genome is different between two individuals, but a properly chosen 1% sample of the genome can determine paternity with high accuracy [224]. These tests are usually conducted using SNPs, haplotypes or short tandem repeats (STRs). The method presented in [225] allows two parties to securely conduct paternity, ancestry and identity tests based on Paillier HE cryptosystem. These tests are

conducted by matching two DNA profiles (STRs) from these parties. To do so, different polynomials over the input STRs are constructed and secretly compared, and yields zero if there is match between them. Another example is a SMC-based method proposed by Blanton *et al.* [226]. It allows two parties to conduct paternity tests for individual sand their supposed children using STRs, and genetic compatibility tests between partners in order to evaluate the risk of having unhealthy babies. Used STRs are kept private from involved parties but the genetic compatibility test still leaks the information about the tested disease. To conduct susceptibility test, methods such as [227–229] have been proposed and all focusing on detecting the presence of mutations and rare variants that can be implicated in a disease. In [228], the test is securely performed using sharing HE and oblivious transfer in an interactive protocol to ensure that both the query and the genome data were kept private. Methods [227] and [229] are also SMC-based focusing on private computation of monogenic disorders and HIV-related cases susceptibilities, respectively.

These methods have many limitations due to the use of cryptographic mechanisms such as SMC and HE that are known for their overheads in computation, communication and storage. This complicates their practical use in the case of large scale genetic data. To overcome these issues, different solutions such as [230] have been proposed. This method allows the efficient and secure outsourcing of storage and genetic testing in cloud environment. To to so, they proposed to combine a secure cryptographic hardware such as Intel Software Guard Extensions (SGX) and asymmetric encryption such as AES. Notice that SGX provides a secure computation unit called enclave where computation like the genetic testing functions are executed in a secure manner. Even though SGX-based methods are more efficient than SMC and HE based methods, they are limited storage capacity and it has recently demonstrated that they are sensitives to in-memory and side-channel attacks [231]. However, the consequences of these attacks and their possible remedies are still open research problems.

1.2.2.4 Secure GWAS and statistical analysis

As explained in 1.1.3 genome-wide association studies are widely used by researchers in order to evaluate the correlation between genetic data such as variants (e.g., SNPs), and diseases. In these studies, genetic data are compared between cases affected by the disease of interest and unaffected controls. The genetic data compared consist on common variants that are tested individually or rare variants within a gene that are considered together. We discuss in this section solutions that were proposed for securing shared or outsourced GWAS.

Securing shared or externalized genetic association studies does not simply consists the protection of genetic data storage and transmission [212, 214]. Indeed, parties involved in such studies may not want that the other parties access their data, the objective and the conclusions of the study, these ones being highly valuable assets. At the same time, the trust one can have in a cloud service provider is quite relative. Thus, it is the data analysis algorithm itself and the way it is shared between parties that have to be protected. Different methods have been proposed in order to conduct privacy-preserving GWAS, especially for common variants. These methods are based on different cryptographic mechanisms including Differential Privacy (DP), Homomorphic Encryption (HE), Secure Multiparty Computation (SMC) and Secure cryptographic Hardware (SCH).

Many privacy-preserving GWAS are based on differential privacy [232, 233] due to the ineffectiveness of data anonymization methods like k-anonymity [234, 235] or l-diversity [236] as it has shown in [237]. Basically, DP adds a random noise to real data in order to ensure individuals' privacy. In [238], the proposed solution allows researchers to conduct exploratory analysis in a differentially private way, including the computation of: i) the number and location of the most significant SNPs to a disease, ii) the *p-values* of a statistical test between a SNP and a disease, iii) any correlation between two SNPs, and iv) the block structure of correlated SNPs. Uhlerop *et al* [239] propose a differentially private release of aggregate GWAS data. They provide DP versions of the χ^2-statistic test and of the minor allele frequencies (MAFs) test. Simmons *et al* [240] introduce a computational GWAS framework that adapts DP principles to protect private phenotype information (e.g., disease status), while correcting for population stratification at the same time. The authors of [241] developed a new statistic tests for private hypothesis testing. These statistics are designed specifically so that their asymptotic distributions, after accounting for the noise added for privacy concerns, match the distributions of the classical (nonprivate) χ^2 statistic test. Similar methods: RandChi and RandChiDist, have been proposed in [242]. In a more general way and as pointed out in [242], it is inherently challenging to use DP techniques for GWAS. The noise added to the original data reduces the utility of data and makes accurate statistical analysis much harder. The level of noise depends on the dataset and on the study's objective and also has to be refined when more data are added.

HE algorithms are other mechanisms that have been used for protecting genetic data. Many solutions to conduct privacy-preserving computation of GWAS using homomorphic encryption have been proposed [16–18]. For instance, the method proposed by Zhang *et al* [17] allows the computation of χ^2-statistic in the homomorphic domain. This method improves the solutions presented in [16] and [18] by proposing a technique which allows the computation of nonlinear operations such as division. To do so, they construct a lookup table which links the division result to the nominator and denominator of the corresponding simplified fraction. This table is encrypted and only known by an authorized party. This one receives the encrypted versions of the fraction numbers and decrypts the results of the division based on the table without the knowledge of the secret decryption key. Even though the proposed strategy performs well, it does not scale enough to treat large-scale data. In [19], Lu *et al* perform GWAS on homomorphically encrypted genotype and phenotype data. In this method, they use a packing technique for the frequency table to improve the efficiency of their method in terms of communication complexity compared to previous ones. Nevertheless, this method is still limited to a small number of variants. Recently, Bonte *et al* [20] proposed two solutions to perform secure GWAS: (1) a somewhat homomorphic encryption (HE) approach, and (2) a secure multiparty computation (SMC) approach. These approaches aim at preventing data breaches when calculating the χ^2-statistic with the idea of not revealing any information other than whether the statistic is significant or not (binary response). Their approach perform better than previous ones taking advantage of a data masking technique so as to perform secure comparison of data between two parties. Unfortunately, while being secure, these methods are most suited for GWAS based on frequencies. Indeed, HE is limited when it comes to statistical analysis processes that are already of great complexity when applied over unencrypted data. To sum up, today, homomorphic encryption based privacy-preserving GWAS are limited in terms of

practical use.

Several other SMC-based methods for securing GWAS have been proposed [243–247]. Kamm *et al* [243] present a data collection and computation system where genetic data are distributed among several parties based on additive secret sharing. Constable *et al* [244] present a privacy-preserving GWAS framework on federated genomic datasets. They secure the χ^2-statistic test on top of SMC systems based on garbled circuit. However, this scheme cannot be generalized to more than two participants. Zhang *et al* [245] propose a secret sharing based SMC approach to secure the χ^2-statistic test, MAF and Hamming distance (HD) computations. Contrarily to [244], this one can be scaled to more than two parties. Hyunghoon *et al* [246] describe a protocol for large-scale genome-wide analysis using multiparty computation techniques. The GWAS method they focus on is a method that enables the identification and the correction for population stratification biases before computing CATT statistics. Bloom [247] proposed a distributed algorithm based on SMC in order to secure a linear regression. SMC-based methods show better performances that HE-based ones, but they still have an important overhead in terms of communication complexity compared to the same computation in a centralized nonencrypted environment. Thus, this higher complexity hinders practical adoption of SMC solutions over the large-scale genomic/genetic data.

To overcome these issues, a few numbers of methods based on the combination of encryption and secure cryptographic hardware (SCH) have been developed. As stated above, the idea is to isolate sensitive data in a protected computation unit (enclave) that allows secure computation. For instance, Chen *et al* [230] present a method based on AES encryption and Intel's Software Guard Extensions (SGX). Data are encrypted with AES before being sent to SGX, where data are decrypted before being securely processed. In [248], authors propose a hybrid framework where several algorithms used in GWAS such as Linkage Disequilibrium (LD) computation, Hardy-Weinberg Equilibrium (HWE) test, CATT and Fisher's Exact Test (FET) can be securely performed on federated genomic datasets. They exploit homomorphic encryption and SGX due to the fact that HE allows to compute linear operation over encrypted data in a secure way, especially, the sum of all entities frequencies tables in secure way. Moreover, HE allows to achieve randomness in encrypted data thanks to its probabilistic properties. However, as mentioned in previous section, SCH-based methods are sensitive to many attacks such as side-channel attacks [231,249,250], and the consequence of these attacks and their possible remedies is an open research problem.

Table 1.3 sums-up all the above methods accordingly the genetic algorithm they have been applied to, as well as their respective applications.

1.3 Conclusion

As we have seen in this chapter, genetic data are widely collected, shared and externalized in open environments in order to allow different institutions, individuals or researchers to access data for various purposes. However, genetic data sharing and/or outsourcing come with several security issues due to the fact that these data are sensitives. As a consequence, strict legislative and ethical rules have been defined and impose ensuring the security of these data in terms of several security objectives such as *i) privacy* and *confidentiality*; *ii) availability*; *iii) integrity*; and *iv) traceability*.

We have seen that different security tools (physical or logical) must be defined in order to respond to these security objectives.

Implementing the security of genomic data is a complex process due to the nature of human genome, and must depend on a specific security policy. In addition, proposed solutions must be complementary and consistent in order to achieve a high level of security. Consequently, a compromise has to be found in order to ensure an acceptable security level while not perturbing medical services (in the genomic are used in healthcare) or research results (in the case these data are used in genomic research). Several mechanisms such as homomorphic encryption, secure cryptographic hardware or watermarking mechanisms have been proposed. However, any of these mechanisms can fulfill all security objectives and many of them such as SMC and HE based solutions are suffering from their overhead complexity which makes them no practical in real life. In addition, some important statistical algorithms that are used in GWAS are still no protected.

It is in this context that we propose in chapters 2 and 4, two solutions that allows the privacy-preserving genetic association studies in cloud environments. The first one which is based on fully homomorphic encryption ans secure multiparty computation is the first solution that secures collapsing method using a logistic regression model. However, like other techniques that are based on these mechanisms, our solution is limited in use due to its computational complexity overheard. The second solution secure WSS and is practical in real life use. On the other hand, all solutions that have proposed for genetic data watermarking are for cellular DNA, and for various purposes including integrity control and copyright protection. However, They can not be used for watermarking genetic data that are used in GWAS studies. This work responds this issue by developing in chapter 5 an adapted watermarking mechanism for genetic data used in outsourced GWAS. In the next chapter will describe our FHE-based solution that allows to securely computation of collapsing method based a logistic regression model.

Table 1.1: A synthetic overview of existing secure and privacy-preserving schemes for genetic data

Scheme	**Genetic computation**	**Considered Architecture**	**Statistical algorithm**	**Security mechanisms**	**Type of considered genetic data**
Tramer *et al* [233]	GWAS	Secure outsourcing	χ^2-statistic	DP	Common variants
Kifer *et al* [241]	GWAS	Secure outsourcing	χ^2-statistic, GOF	DP	Common variants
Asharov *et al.* [219]	GSC	Secure collaboration	Approximate edit distance	Garbled circuit, Oblivious transfer, Secret sharing	Genetic sequences
Souza *et al.* [220]	GSC	Secure outsourcing	Queries on VCF	PIR, AES, Hash functions, FV cryptosystem	All types of genetic variants
Sei *et al* [242]	GWAS	Secure outsourcing	χ^2-statistic	DP	Common variants
Kamm *et al* [243]	GWAS	Secure outsourcing	χ^2-statistic	Secret sharing	Common variants
Constable *et al* [244]	GWAS	Secure outsourcing	χ^2-statistic, MAFs	Garbled circuit	Common variants
Zhang *et al* [245]	GWAS, GSC	Secure outsourcing	χ^2-statistic, MAFs and Hamming distance	Lightweight computational footprints, Secret sharing	Common variants (Start from VCF files (same entry format as for sequencing data) Association study is only performed with common variants but consider also rare variants in sequence comparison)
Cho *et al* [246]	GWAS	Secure outsourcing	CATT, Possible application to logistic regression	Secret sharing	Common variants
Bloom *et al* [247]	GWAS	Secure outsourcing	Linear regression	Secret sharing	Common variants

Continued on next page

Table 1.1 – *Continued from previous page*

Scheme	**Genetic computation**	**Considered Architecture**	**Statistical algorithm**	**Security mechanisms**	**Type of considered genetic data**
Wang *et al* [251]	GWAS	Secure outsourcing	Exact logistic regression	BGV cryptosystem	Rare and common variants (But each SNP is tested individually (one at a time))
Lauter *et al* [16]	GWAS	Secure outsourcing	HWE, PGOF, χ^2-statistic, CATT, Linear regression, LD	SHE cryptosystem (Not published yet)	Common variants
Kim *et al* [18]	GWAS	Secure outsourcing	MAFs, χ^2-statistic	YASHE and BGV cryptosystems	Common variants (Similar to [245] : start from a VCF file only for sequence comparison and do not consider rare variant association tests)
Zhang *et al* [17]	GWAS	Secure outsourcing	χ^2-statistic	BGV cryptosystem	Common variants
Lu *et al* [19]	GWAS	Secure outsourcing	χ^2-statistic, LD, HWE	BGV cryptosystem	Common variants
Bonte *et al* [20]	GWAS	Secure outsourcing	χ^2-statistic	Secret sharing, Blinding, FV cryptosystem	Common variants
Chen *et al* [230]	GWAS, QGD	Secure outsourcing	Queries on VCF, many possible computation algorithms	AES-GCM cryptosystem, SGX, Hash functions	None (They do not propose association test but solutions to query data on VCF files)
Chen *et al* [252]	GWAS	Secure outsourcing	Transmission Disequilibrium Test (TDT)	AES-GCM cryptosystem, SGX	Common variants

Continued on next page

Table 1.1 – *Continued from previous page*

Scheme	**Genetic computation**	**Considered Architecture**	**Statistical algorithm**	**Security mechanisms**	**Type of considered genetic data**
Sadat *et al* [248]	GWAS	Secure outsourcing	HWE, CATT, FET and LD but the proposed method can be used for other algorithms such as TDT, EIGENSTRAT and Linear mixed model [240]	Paillier cryptosystem, SGX	Common variants
Kantarcioglu *et al.* [212]	QGD	Secure outsourcing	-	Paillier cryptosystem	Common variants, Genetic sequences
Canim *et al.* [213]	QGD	secure outsourcing	-	SCP, AES cryptosystem	Common variant, Genetic sequences
Ghasemi *et al.* [214]	QGD	secure outsourcing	-	Paillier cryptosystem	Common variants, Genome sequences
Nassar *et al.* [215]	QGD	secure outsourcing	STR-based matching	Paillier cryptosystem	Common variants, Genome sequences
Hasan *et al.* [216]	QGD	secure outsourcing	Tree-based indexing	Garbled circuit, AES and Paillier cryptosystems	Common variants, Genomic databases
Atallah and Li [217]	GSC	Secure outsourcing	Edt distance, Dynamic programming (Smith-Waterman)	SMC, OT, HE	Genetic sequences

Continued on next page

Table 1.1 – *Continued from previous page*

Scheme	**Genetic computation**	**Considered Architecture**	**Statistical algorithm**	**Security mechanisms**	**Type of considered genetic data**
Jha et al. [218]	GSC	secure collaboration	Edit distance	Oblivious transfer, Garbled circuit	Genetic sequences
Wang et al [253]	GSC	Secure collaboration	Multiple algorithms including Edit distance and Multiple alignment	Garbled circuit	Common variants
Wang et al [222]	GSC	Secure outsourcing	Pattern matching	Predicate encryption	Genetic sequences
Troncoso-Pastoriza *et al.* [228]	PGT	Secure collaboration	Finite Automata, Levenshtein distance	Secret sharing, Oblivious transfer, Paillier cryptosystem	Genomic sequences
McLaren *et al.* [229]	PGT	Secure outsourcing	-	Proxy re-encryption, AES-CCM and Paillier cryptosystems	Common variants
Johnson *et al* [238]	GWAS	Secure Outsourcing	χ^2-statistic, FET, Logistic regression	DP	Common variants
Uhlerop *et al* [239]	GWAS	Secure Outsourcing	χ^2-statistic, MAFs	DP	Common variants
Jagadeesh *et al.* [227]	PGT	Secure outsourcing, Secure collaboration	Boolean operations	Garbled circuit	Rare variants
Blanton *et al.* [226]	PGT	Secure collaboration	STR-based matching	Garbled circuit, Oblivious transfer, Hash functions	Common variants
Bruekers *et al.* [225]	PGT	Secure collaboration	STR-based Matching	SMC, Hash functions, Paillier cryptosystem	Genetic sequences

CHAPTER 2

Privacy-preserving GWAS using fully homomorphic encryption

As defined in chapter 1, the main objective of GWAS is the identification of genetic variants that are associated with some diseases. These studies are mainly based on the statistical analysis of genetic data shared between different entities such as a genomic research unit (GRU) who possesses genetic variants from affected individuals (cases) and a genomic research center (GRC) who has genetic variants from unaffected individuals (controls). GWAS take advantage of cloud computing capabilities which allow users the storage and processing of large amount of data. However, As we have seen in the previous chapter, shared and/or outsourced genetic data present several security needs in terms of privacy, confidentiality, integrity, etc., that are derived from strict ethics and legislative rules.

Indeed, during outsourcing, data owner loses the control over his data. Even if in some cases a service level agreement has been signed between the cloud service provider (CSP) and the user has not actually any other choice than trusting the CSP. In addition, as introduced in chapter 1, the human genome is very sensitive in the sense that it is unique to its owner and can be linked to individual sensitive information or about his or her relatives, from a clinical and behavioral point of view. Thus, there is a need for protecting genetic data during their storage, sharing or/and processing in the cloud. Existing security mechanisms that have been proposed for performing privacy-preserving genome-wide association studies (GWAS) in cloud environments, such as differential Privacy, homomorphic encryption, secure multiparty computation and secure cryptographic hardware present some weaknesses especially in terms of type of users' data that can be externalized during association study. In addition, there are many statistical algorithms used in GWAS that are not yet protected. This was the case of collapsing method. In this context, new appropriate methods must be defined for securing these GWAS.

In this chapter, we are interested in securing the collapsing method [13], a case-control GWAS, with the objective to test whether the proportions of individuals with rare variants in cases and controls differ. More clearly, we present a privacy-preserving GWAS method that allows to securely compute collapsing method based on the logistic regression model. To do so, our solution takes advantage of fully homomorphic encryption, and especially of the BGV cryptosystem which

allows encrypting and process blocks of data, in combination with secure multiparty computation. Contrarily to the previous schemes in the state of the art, it considers that all user data are outsourced and CSP only returns to the users whether the test is significant or not, making our scheme more secure. In addition, in order to make our solution more efficient, we proposed an original data packing strategy that allows the reduction of communication and computation complexities as it allows processing data in parallel.

2.1 Overview on genome-wide association studies and security mechanisms

In this section we present an overview on a genome-wide association study that we are interested in through this chapter, as well as the security mechanisms that we exploited during the protection of this study.

2.1.1 Collapsing method based on logistic regression

Methods used to test for association with common variants are underpowered to test for association with rare variants. To overcome this issue several powerful methods that allow studies on rare variants have been proposed and one of them is collapsing method [13]. This method involves collapsing genotypes across variants in a specific gene and applying a statistical test such as logistic regression. It is a powerful tool for analyzing rare variants. To describe collapsing method, let us consider a sample of N individuals constituted of N_1 cases and N_2 controls. Let Y_j be the disease status where Y_j equal 1 if an individual j is affected by disease/"case" or 0 if this individual is unaffected/"control". To perform the association test, k sites of the studied gene where the variants of interest exist are chosen by GRC and GRU. After this selection, a variable X_i is defined for each individual such that

$$X_i = \begin{cases} 1 & \text{if the individual } i \text{ has at least one variant allele on any of the } k \text{ sites} \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

The detection of the association between disease and the gene is conducted by testing if the proportion of individuals in cases and controls differ. To do so, several statistical methods such as logistic regression model [254] can be used where outcome variables are $\{Y_j\}_{j=1,2,\cdots,N}$ and the predictor variables are $\{X_j\}_{j=1,2,\cdots,N}$. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. More clearly, it consists in studying the dependence between a binary variable Y to be explained (qualitative variable with two modalities) and one or several predictor variables $X_1, X_2, \cdots, X_l$ which are also qualitative. Notice that in the case these variables are quantitative, the statistical model is called linear regression. As the variable Y is a binary variable, the $logit$ transformation is applied so as to define logistic regression model such as

$$logit(p) = ln(\frac{p}{1-p}) = \beta_0 + \beta_1 X_1 + \cdots + \beta_l X_l \tag{2.2}$$

Table 2.1: Distribution of frequencies for cases and controls

	$X=0$	$X=1$
$Y=0$	N_{00}	N_{01}
$Y=1$	N_{10}	N_{11}

where p is the probability that Y_j occurs and $\{\beta_l\}_{l=0,1,\cdots,l}$ are the regression coefficients, and are estimated through the realization vectors of $(Y, X_1, \cdots, X_l)$ that are $(y_i, x_{i2}, x_{i3}, ..., x_{il})$ for $0 < i < N$ from a sample of N individuals. Regression coefficients are computed such that the probability of observing the realizations of this sample is maximum and are estimated using the maximum likelihood function such that

$$L(\beta) = \prod_{i=1}^{N} p_i^{y_i}(1-p_i)^{1-y_i} \tag{2.3}$$

where $\beta = (\beta_1, \beta_2, \cdots, \beta_l)$ and p_i is such that

$$p_i = \frac{exp(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_l x_{il})}{1 + exp(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_l x_{il})} \tag{2.4}$$

Herein, the values of $\{\beta_l\}_{l=0,1,\cdots,l}$ we want to calculate are the ones that maximize the function L. These values can also be obtained by using the ln function of L which does not change the result. Thus, maximizing L is equivalent to maximizing $ln(L)$ such that

$$g(\beta) = ln(L(\beta)) = ln(\prod_{i=1}^{N} p_i^{y_i}(1-p_i)^{1-y_i}) = \sum_{i=1}^{N_1+N_2} y_i ln(p_i) + (1-y_i) ln(1-p_i))] \tag{2.5}$$

To conduct the collapsing method, two hypothesis tests are considered such that

$$\begin{cases} H_0: & \text{the studied gene is not associated to the disease} \\ H_1: & \text{the studied gene is associated to the disease} \end{cases} \tag{2.6}$$

The null hypothesis (H_0) states that the studied gene is not associated to the disease. In that case, $\{\beta_l\}_{l=1,\cdots,l}$ are equal to 0 and $logit(p) = \beta_0$. The alternative hypothesis (H_1) informs that there is an association between the disease and the gene. In this case, at least one value of $\{\beta_l\}_{l=1,\cdots,l}$ is not null and the $logit$ model remains $logit(p) = \beta_0 + \beta_1 X_1 + \cdots + \beta_l X_l$. From here on and for sake of simplicity, we will consider the model $logit(p) = \beta_0 + \beta_1 X_1$ and (2.5) becomes,

$$g(\beta) = \sum_{i=1}^{N_1+N_2} y_i(\beta_0 + \beta_1 x_i) - \sum_{i=1}^{N_1+N_2} ln(1 + exp(\beta_0 + \beta_1 x_i)) \tag{2.7}$$

As presented in table 2.1, our sample of N individuals is composed of four frequencies that are N_{01}, N_{11}, N_{00} and N_{10}. They represent the number of individuals that: have at least one variant allele on any of the k sites in controls; have at least one variant allele on any of the k sites in cases; have non variant allele on the k sites in controls and have non variant allele on the k sites in cases, respectively. In that case, (2.7) is written as follows

$$\begin{aligned} g(\beta) = {} & N_{10}\beta_1 + N_{11}(\beta_1 + \beta_2) - [(N_{01} + N_{11}) ln(1 + exp(\beta_1 + \beta_2))] \\ & - [(N_{00} + N_{10}) ln(1 + exp(\beta_1))] \end{aligned} \tag{2.8}$$

This function has a global maximum only if the values of N_{00}, N_{10}, N_{01} and N_{11} are different from 0. If one of these values is 0, it does not mean that the gene is not associated with the disease (nor that the gene is associated with the disease). It means that during association test, we can not apply logistic regression. For example, if $N_{01} = 0$, $N_{11} = N_1$, $N_{00} = N_2$ and $N_{10} = 0$, it means that all individuals that have a given disease such as diabetes (controls) have a variant allele on any of the chosen k sites, and none healthy individuals have a variant allele on any of the chosen k sites. Thus, in this case the studied gene is associated with diabetes. In the following, we consider that all frequencies are different from 0.

After the maximization of (2.8) under H_1, the estimation of regression coefficients is such that

$$\hat{\beta}_{H_1} = (\hat{\beta}_0, \hat{\beta}_1) = (ln(\frac{N_{10}}{N_{00}}), ln(\frac{N_{11} * N_{00}}{N_{01} * N_{10}})) \tag{2.9}$$

and under H_0, the maximization of (2.8) leads to

$$\hat{\beta}_{H_0} = \hat{\beta}_0 = ln(\frac{N_{11} + N_{10}}{N_{01} + N_{00}}). \tag{2.10}$$

After estimating all regression coefficients, the statistic test result is calculated as $Stat$ such that

$$\begin{aligned} Stat &= 2(g(\hat{\beta}_{H_1}) - g(\hat{\beta}_{H_0})) \\ &= 2[N_{10}(ln(N_{10}) - ln(N_{00} + N_{10})) + N_{11}(ln(N_{11}) - ln(N_{01} + N_{11})) \\ &+ N_{01}(ln(N_{01}) - ln(N_{01} + N_{11})) + N_{00}(ln(N_{00}) - ln(N_{00} + N_{10})) \\ &- ((N_{01} + N_{00})(ln(N_{11} + N_{10}) - ln(N_{01} + N_{00})) - (ln(N_{01} \\ &+ N_{11} + N_{00} + N_{10}) - ln(N_{01} + N_{00}))(N_{01} + N_{11} + N_{00} + N_{10}))] \end{aligned} \tag{2.11}$$

This value being distributed according to the χ^2 distribution with degree of freedom *df* ($\chi^2(df)$), if $Stat > \chi^2(df)$, H_0 is rejected and this means that the studied gene is associated with the disease, otherwise, we can not decide whether the gene is associated to disease or not. Notice that the degree of freedom corresponds to the difference between the number of predictor variables in H_1, and the number of predictor variables in H_0. In our case, $df = 1$ and using this value combined with the threshold value $\alpha = 0.05$, we obtain $\chi^2(1) = 3.841$ from χ^2 distribution table.

2.1.2 Security mechanisms: fully homomorphic encryption

As we have seen in chapter 1 homomorphic encryption is one of powerful security mechanisms that are used in the protection of shared and/or outsourced genetic data. It allows performing linear operations over encrypted data. In this work, we opted for BGV cryptosystem, a fully homomorphic encryption scheme which is based on the ring learning with errors (RLWE) [255]. The parameters of this cryptosystem are described as follows: we select a ring $R = \mathbb{Z}[x]/f(x)$ where $f(x) = x^d + 1$ is a cyclotomic polynomial and d is power of 2. $R_q = \mathbb{Z}_q[x]/f(x)$, q is a prime number verifies $q = 1 \mod 2d$. The element in R_q can be viewed as d degree polynomial over Z_q. The computation in R_q are the addition and multiplication on polynomials, result reduces modulo $f(x)$ with coefficient in $]-q/2, q/2]$. We take a discrete Gaussian error distribution $\mathcal{N} = N(0, \sigma)$, the parameter σ is standard deviation over R. The plain-text space is

$R_t = \mathbb{Z}_t/f(x)$. The selection of the parameter t, q, d and σ is to guarantee that the homomorphic encryption and decryption are correct and the scheme is secure. This cryptosystem is conducted using these following algorithms:

Key generation samples ring elements are $e, s \longleftarrow \mathcal{N}$ and $a_1 \longleftarrow R_q$. From these elements, a key pair (K_p, K_s) is generated, where $K_p = (a_0 = (a_1 s + te), a_1)$ is the public key used for data encryption, and $K_s = s$ is the secret key used for data decryption. During the encryption process, the message $m \in R_t$ is encoded as a degree polynomial with coefficients in $\mathbb{Z}_t$. Thus, given a public key $K_p = (a_0, a_1)$ and the encryption algorithm samples $u, f, d \longleftarrow \mathcal{N}$, the cipher-text c is computed as

$$c = (c_0, c_1) = (a_0 u + tg + m, a_1 u + tf) = E[m, u, g, t, f] \tag{2.12}$$

To decrypt the cipher-text $c = (c_0, c_1)$, we use the secret key $K_s = s$ and the decryption function D. Thus, the plain-text m is obtained such that

$$D[c, K_s] = m = (c_0 + sc_1 \mod q) \mod t \tag{2.13}$$

As said at the beginning of this section, this cryptosystem is a fully homomorphic encryption algorithm and has multiplicative and additive homomorphic properties. Considering two plain-texts m and m', and two cipher-texts $c = (c_0, c_1) = (a_0 u + tg + m, a_1 u + tf) = E[m, u, g, t, f]$ and $c' = (c'_0, c'_1) = (a_0 u' + tg' + m', a_1 u' + tf') = E[m', u', g', t, f']$, the homomorphic addition c_{add} on these two cipher-texts is delivered as

$$\begin{aligned} c_{add} &= c + c' = E[m + m', t, (u + u'), (g + g'), (f + f')] \\ &= (c_0 + c'_0, c_1 + c'_1) = (a_0(u + u') + t(g + g') + (m + m'), a1(u + u') + t(f + f')) \end{aligned} \tag{2.14}$$

Regarding, the homomorphic multiplicative c_{mul} of c and c' is given by

$$c_{mul} = c_0 c'_0, c_0 c'_1 + c_1 c'_0, c_1 c'_1 \tag{2.15}$$

In the next sections, we discuss how this cryptosystem, combined with SMC techniques that have seen in chapter 1, can be used in order to protect outsourced collapsing method.

2.2 Overview on existing privacy-preserving GWAS methods based on fully homomorphic encryption

As we have presented in chapter 1, fully homomorphic encryption (FHE) allows the computation of both addition and multiplication operations on encrypted data without decrypting them.

Up to now, many solutions have been proposed for conducting privacy-preserving computation of GWAS using fully homomorphic encryption [16, 18, 251, 256–258]. Yasuda *et al.* [256] gave a practical solution for conducting computation of multiple Hamming distance values using the LNV scheme [21] on homomorphically encrypted data, so as to find the locations where a pattern occurs in a text. Some solutions such as [257,258] applied homomorphic encryption to machine learning,

and described how to secure conducting predictive analysis based on an encrypted learned model. Lauter *et al.* [16] proposed a method that allows secure computation of basic statistic algorithms which are commonly used in genetic association studies such as Hardy-Weinberg Equilibrium (HWE), Pearson Goodness-Of-Fit (PGOF), Linkage Disequilibrium (LD), linear Regression, χ^2-statistic and Cochran-Armitage Test for Trend (CATT). However, this method is not practical due do its storage and computation complexities. Wang *et al* [251] adopted homomorphic encryption on rare variants to perform exact logistic regression. Kim *et al* [18] proposed a scheme that allows secure computation of MAFs, and the χ^2-statistic using homomorphic encryption. Even though they use a specific encoding technique to improve the work presented in [16], they only homomorphically compute the allele counts, and execute other operations on decrypted data. Other methods such as [17, 19, 20, 259] have been proposed.

All previous methods were proposed for protecting several statistical algorithms but collapsing method based on logistic regression was not secured. In this chapter, we present the first scheme that allows the secure computation of collapsing method based on the logistic regression model by combining fully homomorphic encryption and secure multiparty computation. Contrarily to the previous schemes, our solution considers that all user data are outsourced and only returns to the users whether the test is significant or not making our solution more secure.

2.3 Privacy-preserving GWAS: Collapsing method

In this section, we first introduce the outsourcing framework we consider before presenting how collapsing method can can securely conducted using fully homomorphic encryption and secure multiparty computation.

2.3.1 Considered data outsourcing scenario

As shown in Figure 2.1, the scenario we consider in our framework is composed by three entities: a Genomic Research Unit (GRU) who owns data from individuals with disease (cases); a Genomic Research Center (GRC) who has data from healthy individuals (controls) and a Cloud Service Provider (CSP). GRU and GRC outsource their data for storage or in order to be used for performing genetic association studies. In the later case, GRC performs on the cloud a case-control association test where the objective consists on determining whether a gene is associated with disease, through a statistical method such as collapsing method based on logistic regression model. As described in chapter 1, this data externalization can causes many security threats and must be protected during their storage or processing on the cloud. The objective of this chapter is to set up a scenario where genetic data is shared, processed or stored on the cloud in a protected manner. The association test is performed on the cloud and GRU receives the result of the processing without compromising the confidentiality of the data, and this by considering a passive attacker model, and without including a trusted third party. We describe the details of proposed solution in the next section.

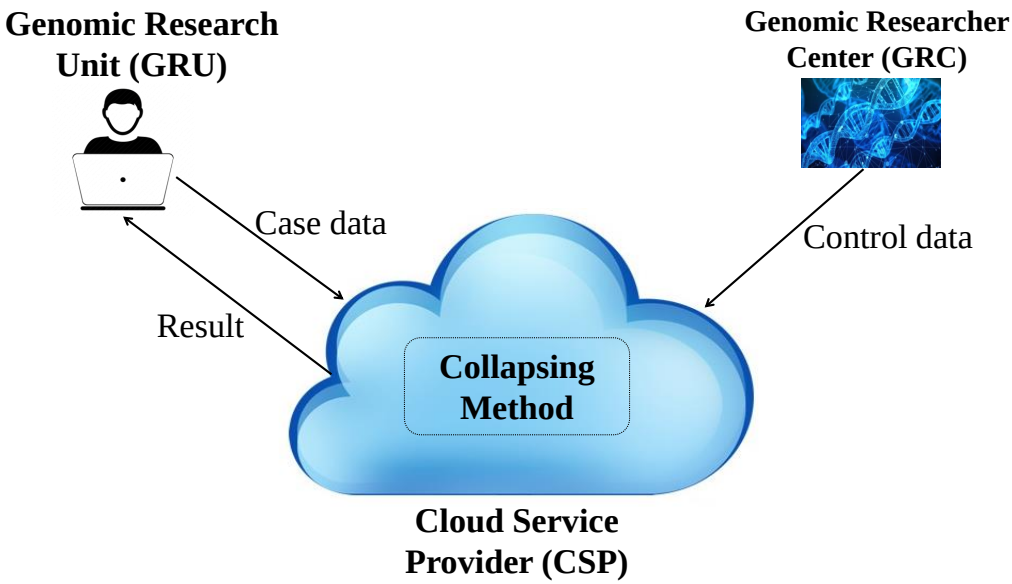

Figure 2.1: Considered genetic data outsourcing scenario

2.3.2 Proposed scheme

We want to implement a secure scenario in which a cloud service provider (CSP) stores or processes protected data that are outsourced by two entities: a genomic research center (GRC) and a genomic research unit (GRU). Processing and result sent to GRU are conducted without revealing any information that can be used by GRC in order to get access to GRU's data or vice-versa, in particular frequencies. Indeed, GRU knows the frequencies N_{11} and N_{10} (these are calculated from his/her data) and these ones can be combined with the processing results from CSP and the overall size of the simple N in order to extract some information about GRC's data (N_{01} and N_{00}). Therefore, the association test must be conducted without revealing any information to GRU. This is possible if GRU does not know sites of interest that have been "collapsed". Similarly, even of GRC does not directly receive association test results from CSP, he/she may get access to data when for example GRU publish theme on internet. Thus, GRC should not know the chosen sites of interest.

As introduced at the beginning of this section, our framework considers three entities: GRU who owns N_1 cases; GRC who has N_2 controls and CSP with the computing power. We assume that all data are encrypted using homomorphic asymmetric cryptosystems such as BGV, and stored on the cloud by CSP, GRU and GRC possess their own pair of keys, respectively i.e., (K_p^U, K_s^U) and (K_p^C, K_s^C) where K_p^U and K_p^C are public keys while K_s^U and K_s^C are the private keys. In addition, GRU and GRC ask CSP to perform collapsing method on their data and send the result to GRU, and CSP is considered as "honest but curious". More clearly, it follows all processing steps but may try to infer information about GRU and GRC data. In this chapter, we are focusing on protecting data confidentiality and individual privacy. Other data threats such as data integrity or traceability will be the subject of next chapters. On the CSP side, some operations such as the computation of N_{00}, N_{01}, N_{10} and N_{11} are conducted based on data encrypted by K_p^U and K_p^C as

Table 2.2: Comparison test between $reg(N_{00}, N_{01}, N_{10}, N_{11})$ and $reg(N_{00}+1, N_{01}+1, N_{10}+1, N_{11}+1)$, executed 10000 times.

Sample $(N_{00}+N_{01}+N_{10}+N_{11})$	100	1000
Error (E_{rr})	0.7%	0.1%

they are carried out separately on the data of each entity. Because the collapsing method is based on some nonlinear operations (e.g., logarithmic and division operations) that cannot be achieved with BGV, a third party entity (TPE) is introduced. It has also a pair of key, a public key K_p^T and a secret key K_s^T. We made the choice of no encrypting data with the same key K_p^T from TPE because we did not want to build our framework on a single TPE, which, in the case of a collusion with the CSP, will compromise the data. Even though the framework we propose does not prevent the association between the CSP and the TPE, but at any time GRU and GRC can decide to no longer participate in the association test and this will protect their respective data. Finally, TPE will intervene at the end of the scenario for decrypting results of the processing and send it to GRU. Case data from GRU and control data from GRC correspond to VCF files (see chapter 1), one VCF file contains a table the lines of which correspond to the variants and the columns of individuals. Each table element contains binary values. To benefit of the BGV batching property and of the fact it allows binary operations, one line is encrypted as a single message. For a test of M variants, we have two sets of encrypted vectors $\{E[v_n^{cas}, K_p^U]\}_{n=1...M}$ and $\{E[v_p^{con}, K_p^C]\}_{p=1...M}$. It is difficult for CSP to differentiate case and control data in encrypted form. To simplify the processing on encrypted data, case or control data will be indicated to CSP, and this will not cause any security breach as long as individual genotypes are encrypted.

During the computation of frequencies N_{00}, N_{01}, N_{10} and N_{11}, in some cases one or more values of these frequencies can equal to 0. Since the CSP is working on encrypted data, it will continue the computation without knowing it and send the significant results to GRU but it is not always the correct result. To overcome this issue, CSP will add 1 on these frequencies so that the logistic regression can be conducted and therefore, instead of working on a sample of N_1+N_2 individuals, the CSP will work on a sample of N_1+N_2+4 individuals. With this new sample, we have four possibilities: one case with a variant, one case without a variant, one control with variant and one control without a variant) and this will not change the final result of the association test. Indeed, we have implemented a function $reg(N_{00}, N_{01}, N_{10}, N_{11})$, which returns the result of the processing (rejection of the hypothesis H_0 or not) for N_{00}, N_{01}, N_{10} and N_{11} different from 0. As illustrated in table 2.2, by comparing $reg(N_{00}, N_{01}, N_{10}, N_{11})$ with $reg(N_{00}+1, N_{01}+1, N_{10}+1, N_{11}+1)$, we have less number of cases where outputs of our function reg are different. In addition, let E_{rr} be the percentage of cases where $reg(N_{00}, N_{01}, N_{10}, N_{11}) \neq reg(N_{00}+1, N_{01}+1, N_{10}+1, N_{11}+1)$, this value continue to decrease with the increase of individuals in the sample.

After analyzing the impact of null values on our scenario we describe how our secure collapsing method is conducted as shown in the Figure 2.2. We recall that GRU and GRC data are encrypted by their respective keys before being outsourced. Our method is detailed in 9 steps as follows.

1. GRU sends a request to the CSP and GRC to initialize the association test, specifying the

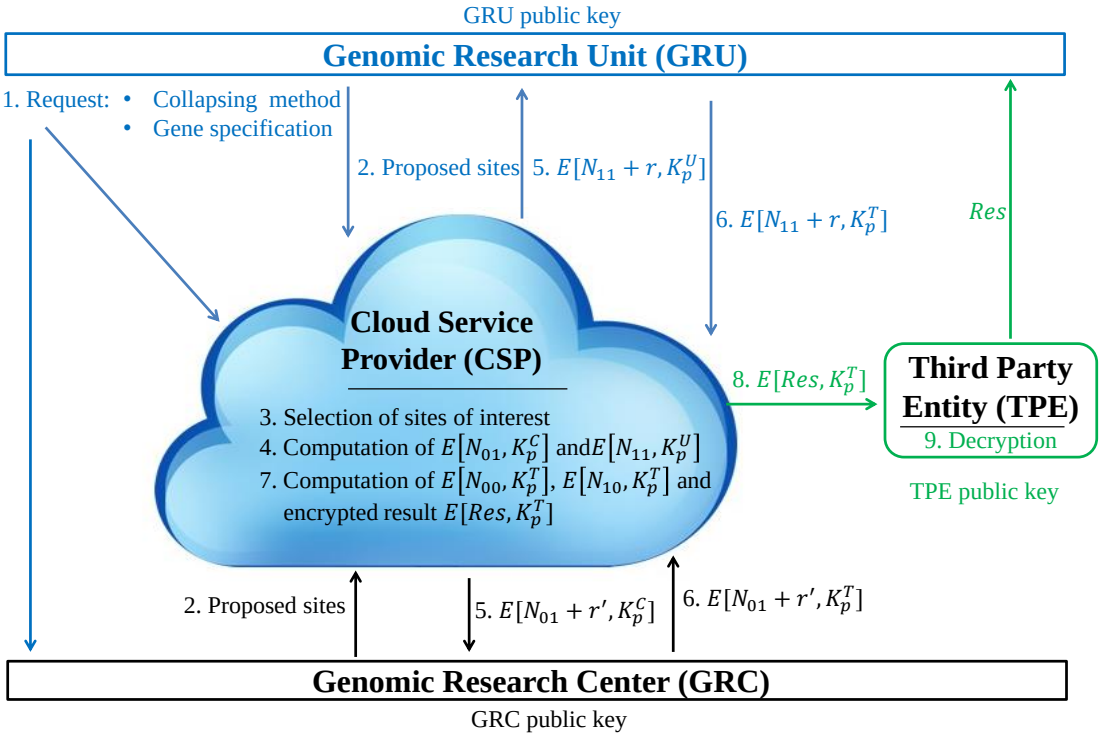

Figure 2.2: Different steps of our secure collapsing method

gene it is interested in. The positions and the corresponding genotypes in the gene of interest are not necessarily the same for GRU and GRC. Thus, in the studied gene, each side must indicate to the CSP the chosen positions.

2. GRU and GRC choose their respective sites of interest and send them to CSP. They correspond to positions into the vectors $\{E[v_n^{cas}, K_p^U]\}_{n=1...M}$ and $\{E[v_p^{con}, K_p^C]\}_{p=1...M}$.

3. CSP selects the sites of interest in the stored samples of GRC and GRU and constructs two encrypted vectors of binary values $E[C_U, K_p^U]$ and $E[C_C, K_p^C]$ from $\{E[v_n^{cas}, K_p^U]\}_{n=1...M}$ and $\{E[v_p^{con}, K_p^C]\}_{p=1...M}$, respectively.

4. CSP computes $E[N_{01}, K_p^C] = E[\sum_k C_C(k), K_p^C]$ and $E[N_{11}, K_p^U] = E[\sum_k C_U(k), K_p^U]$.

5. CSP selects two random values r and r', computes $E[r, K_p^C]$ and $E[r', K_p^U]$ and sends $E[N_{01}+r, K_p^C] = E[N_{01}, K_p^C]+E[r, K_p^C]$ to GRC and $E[N_{11}+r', K_p^U] = E[N_{11}, K_p^U]+E[r', K_p^U]$ to GRU. This process corresponds to an additive data masking operation.

6. GRU decrypts $E[N_{01}+r, K_p^C]$ and GRC decrypts $E[b+r', K_p^U]$. Both of them re-encrypt these values using K_p^T and send the results to CSP.

7. CSP computes $E[N_1, K_p^T]$ and $E[N_2, K_p^T]$. With $E[N_{01}, K_p^T]$, $E[N_{11}, K_p^T]$, it calculates $E[N_{00}, K_p^T] = E[N_1, K_p^T] - E[N_{01}, K_p^T]$ and $E[N_{10}, K_p^T] = E[N_2, K_p^T] - E[N_{11}, K_p^T]$.

Then, 1 is added to each value such that

$$\begin{aligned} E[N_{00}, K_p^T] &\longleftarrow E[N_{00}, K_p^T] + E[1, K_p^T] \\ E[N_{01}, K_p^T] &\longleftarrow E[N_{01}, K_p^T] + E[1, K_p^T] \\ E[N_{10}, K_p^T] &\longleftarrow E[N_{10}, K_p^T] + E[1, K_p^T] \\ E[N_{11}, K_p^T] &\longleftarrow E[N_{11}, K_p^T] + E[1, K_p^T] \end{aligned} \tag{2.16}$$

After these computations, CSP interacts with TPE to SMC computes $E[Stat, K_p^T]$ (see eq. 2.11, section 2.1).

8. Thanks to BGV, CSP computes the encrypted sign $E[Res, K_p^T]$ of the encrypted difference $E[Stat - \chi^2(1), K_p^T]$ and sends it to TPE.

9. TPE decrypts Res and sends the decrypted value to the GRU.

As stated in section 2.1, the computation of $Stat$ requires the computation of $ln(.)$, a non-linear function. To secure it, our solution combines homomorphic encryption with a multiplicative data masking. It adds a noise that can be removed thanks to ln property: $ln(ab) = ln(a) + ln(b)$. As result, GRU helps in computation of $ln(N_{11})$ and $ln(N_{10})$, while GRC helps in calculating $ln(N_{01})$ and $ln(N_{00})$. As with $ln(.)$ function we cannot compute $ln(N_{01} + N_{11})$ from $ln(N_{01})$ and $ln(N_{11})$, we have also performed the same multiplicative data masking between CSP and TPE so as to compute $ln(N_{01} + N_{11})$ and $ln(N_{00} + N_{10})$. As shown in Figure 2.3, our multiplicative data masking is conducted as follows: CSP randomly choose six integers $r_1, r_1^{'}, r_2, r_2^{'}, r_3, r_3^{'}$ and computes $E[ln(a), K_p^T]$ where $a \in N_{00}, N_{01}, N_{10}, N_{11}, N_{01} + N_{11}, N_{00} + N_{10}$, based on the encryption value of $ln(ar)$ ($E[ln(ar), K_p^T]$) where $r \in r_1, r_1^{'}, r_2, r_2^{'}, r_3, r_3^{'}$ such that

$$E[ln(a), K_p^T] = E[ln(ar), K_p^T] - E[ln(r), K_p^T] \tag{2.17}$$

All these multiplicative data masking operations are conducted after adding 1 to the frequencies (see eq. 2.16), since we cannot compute $ln(a + 1)$ from $ln(a)$. Moreover, we could not have conducted multiplicative data masking if these frequencies could take the value 0. The next section will focus on experimental results and discussion.

2.4 Experimentation and results

In this section we experimentally verify the above solution on a real genomic database and using BGV cryptosystem. This latter is implemented in HElib, an homomorphic encryption library which is written in C ++ and uses the GMP and NTL libraries. We have chosen BGV because it allows us to optimize the size of encrypted data that can be stored in the cloud thanks to batching. By definition, batching consists on encrypting several messages in one single encrypted message while keeping homomorphic properties of the cryptosystem. This is due to the fact that the polynomial $f(x)$ used in section 2.1.2 is factorized into irreducible polynomials $F_1, F_2, \cdots, F_s \mod q$, and therefore a message $M \in Z_q[X]/f(X)$ can be represented as a set of s messages $(m_1, m_2, \cdots, m_s)$ each corresponding to a polynomial F_i ($m_i = M \mod F_i$).

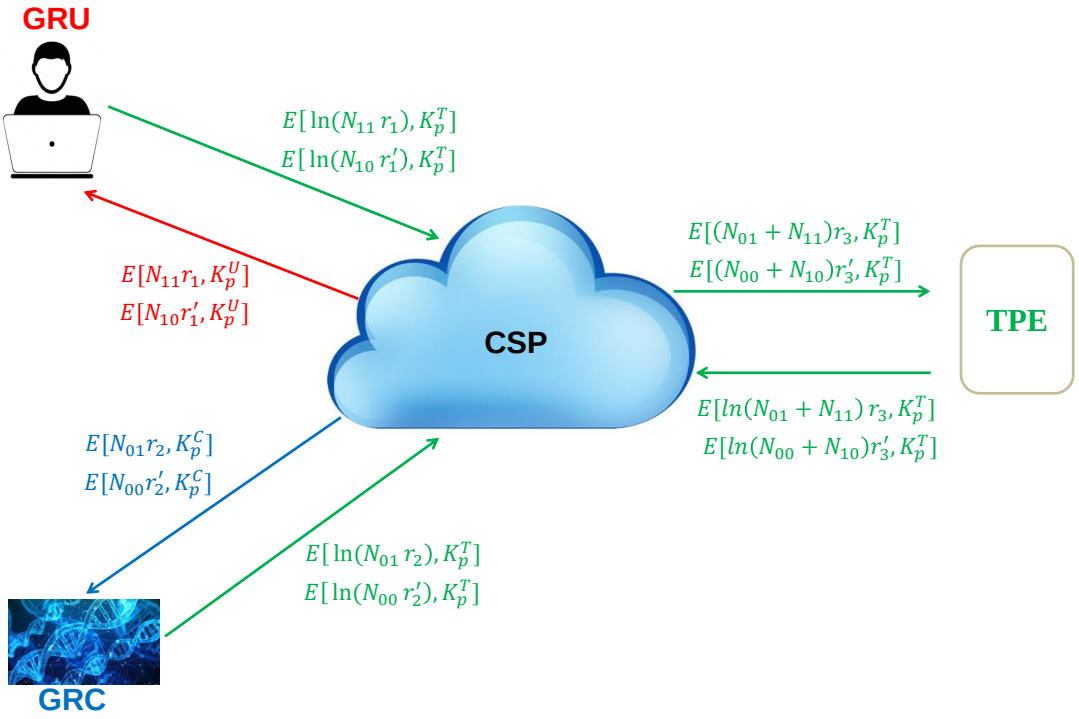

Figure 2.3: Different exchanges between entities during the computation of $ln(.)$

As a result, we can see the space of clear messages as a set of polynomials $(m_1, m_2, \cdots, m_s)$ of $(Z_q[X]/F_1 \times Z_q[X]/F_1 \times \cdots \times Z_q[X]/F_s)$. In addition, with HElib, it is possible to use either the leveled homomorphic encryption or fully homomorphic encryption based on bootstrapping. We will come back to this property in next sections.

2.4.1 Description of HElib library

HElib is a software library that implements homomorphic encryption (HE), specifically the BGV scheme. It uses the variant of BGV which has $Z_p^r[X]/f(X)$ as the space of clear messages where p is a prime number. With batching introduced at the beginning of this section, in HElib, we can declare a vector $[m_1, m_2, \cdots, ms]$ containing several polynomials or integers seen as polynomials, and encrypt them as a single encrypted message. The number of slots or of polynomials that we can put in the same vectors is s and it depends on m, p and r. HElib also offers the possibility of directly encrypting a polynomial of $Z_p^r[X]/f(X)$, but batching is more interesting as it is very useful in the case of encryption of several messages.

In HElib, before generating encryption and decryption keys, we must first instantiate the context in which several parameters are defined. Among them, m, p and r define the space of clear messages which will be used $Z_p^r[X]/f(X)$. It is strongly recommended to choose the smallest value of p^r in order to minimize the sizes of the encrypted messages and computation time. With a fully homomorphic encryption scheme such as BGV, conducting operations on encrypted data increase the noise in this data. To control this issue, in HElib an encrypted message is characterized by

the level and and an estimator of the noise. If noise estimator exceeds a certain threshold, HElib reduces this noise by performing a modulus switching. The level L an encrypted message can have is one of parameters that must be defined in the context. Thereby, before performing any operation on an encrypted message, this one has the level L. If some operations are conducted on this message, its level decreases until it reaches 1 and at this level no modulus switching can be performed. Thus, if we want to conduct other operations on this message, there is an increase of the noise without having any way to control it and the message will no longer decrypted. To overcome this issue, a bootstrapping is applied. This one allows us to evaluate an arbitrary circuit and is a homomorphic evaluation of the decryption circuit in order to refresh an encrypted message for more computations.

Bootstrapping is not applied to any context, the polynomial ψ should satisfy some constraints, HElib offers a set of contexts which allow bootstrapping [260]. If we want to be able to apply bootstrapping on our encrypted data, we must indicate this during the definition of the context and specify if we want to use fully homomorphic version of HElib. Therefore, additional parameters must be defined. Note that when bootstrapping is included, the size of public and private keys is significantly increase, since the encryption of the public key will be added to the public key.

HElib has become a benchmark for evaluating homomorphic encryption because it is the first library that implements a fully homomorphic scheme and it includes efficient optimizations such as batching and bootstrapping. It is now used in various domains such as privacy-preserving GWAS [18]. In this chapter HElib is used in order to implement the proposed solution which consists in securing collapsing method based on logistic regression model.

2.4.2 Encoding and computation on encrypted data

Experiments were conducted on a genetic sample of 57 individuals (20 cases and 37 controls) and these data are extracted from a real genetic database. We have seen in the previous section that data are encrypted using the BGV cryptosystem implemented in HElib library. The data that we encrypt corresponds to individual genotypes that are stored in VCF files (see chapter 1).

Before conducting encryption operations of our genetic data and uploading them on the cloud, a pre-processing step must be conducted. It consists in modifying individual genotypes by replacing all values that are greater than 1 by 1, and the "." which corresponds to a missing variant or position that has not been sequenced by 0. This will be useful as all the data will be expressed in binary form and the processing will be expressed directly in corresponding Boolean circuits. This modification will hide some genetic information such as the identification of alternative allele for a given variant or unsequenced positions but data will still be used for some genetic processing which require only the knowledge about the existence of the reference allele or of an alternative allele in individual genotype. This is the case of collapsing method. Moreover, this transformation will allow us to choose a HElib context where $p = 2$ and $r = 1$.

To optimise the size of the encrypted data, we will use batching for data encryption. To explain this optimization, let us consider for example a sample of case data which contains N_1 individuals but computations are conducted in the same way for control data. The genotype of individual i can

be represented in two vectors A_{1ij}/A_{2ij}, i and j are going from 1 to N_2 and 1 to N_g (the number of positions in the studied gene), respectively. We had two options: *i)* Encrypting data according to individuals and the size of each vector corresponds to the number of positions in the sample. These vectors or slots are filled by individual modified genotypes; *ii)* Encrypting data according to positions and vectors or slots correspond to alleles at these positions. In our method, we opted for the second option. Thus, if s is the size of slot, we have two encrypted vectors for each position p such that

$$\begin{aligned} C_{1p} &= Enc[\cdots, A_{1N_2j}, \cdots, A_{14p}, A_{13p}, A_{12p}, A_{11p}] \\ C_{2p} &= Enc[\cdots, A_{2N_2j}, \cdots, A_{24p}, A_{23p}, A_{22p}, A_{21p}] \end{aligned} \tag{2.18}$$

We conduct these computations for each position and the number of encrypted vectors is $2 \times N_g$. This way of data encryption allow us to compute the encrypted values of observations x_i for all individuals at once and this optimizes the computational cost. We explain this optimisation in next section.

2.4.3 Extraction of observations x_i

In the context we have chosen, we are working with $p = 2$ and $r = 1$. In this case, additions correspond to XOR ($\oplus$) while multiplications correspond to bitwise multiplication ($\otimes$). Therefore, the circuit which allows the extraction of encrypted values of x_i from our encrypted vectors is as follows.

$$\prod_{p=1}^{N_g} [C_{1p} \oplus C_{2p} \oplus E[1, \cdots, 1, K_p^U] \oplus C_{1p} \otimes C_{2p}] = E[\cdots, x_{N_1}, \cdots, x_1, K_p^U] \tag{2.19}$$

From these values, a simple addition on encrypted values allows us the computation of frequencies N_{00}, N_{10}, N_{01} and N_{11}. In order to continue the processing by conducting data masking, the encrypted values of N_{00}, N_{10}, N_{01} and N_{11} must have specific representations. After that, we will continue with the representation of elements from eq.2.11, in order to compute $Stat$. Thus, for masking N_{01} and N_{11} we evaluate a circuit which allows to pass from $E[0, \cdots, 0, x_{N_1}, \cdots, x_1, K_p^U]$ to $E[\cdots, x_{b_m}, \cdots, b_0, K_p^U]$, such that

$$N_{11} = \sum_{i=1}^{N_1} x_i = \sum_{j=1}^{[log_2(N_{11})]} b_j 2^j \tag{2.20}$$

Thus, all the following computations and data masking operations are performed on binary values. Optimizations conducted on the addition, subtraction and multiplication operations in binary form on encrypted data are conducted using algorithms presented in [261] and they help us to reduce the time required to perform the bootstrapping.

2.4.4 Computational results

As explained in previous section, our solution was experimented on a genetic database that contains 57 individuals among them 20 cases and 37 controls considering a gene with 100 positions.

Data are encrypted using the BGV cryptosystem implemented with HElib library and this one optimizes the size of encrypted data to store in the cloud thanks to batching. We ran the proposed solution on a machine equipped with 4 GB RAM, Intel Core i5-5200U, 2.7GHz, running on Ubuntu 18.04 LTS. The public and private keys of GRU, GRC and TPE are generated using the same context. To give an idea about computation time, let us choose a context where security parameter is less than 80. In this context if we chose $L = 20$, the bootstrapping time in this context is $3s$ and this context allowed us to have 60 slots. We randomized by small masks $1 < r < 8$ and the execution time is about 15 minutes with no errors in the test results compared to the same tests conducted on clear data.

Bootstrapping is the operation which consumes much time (it is estimated at 95% [262]). In the context a security parameter is greater than 80 and $L = 20$, the bootstrapping is estimated at 600 seconds. The computation time for the same sample (20 cases and 37 controls) and with the same masks, $1 < r < 8$, is about 50 hours.

2.5 Conclusion

In this chapter we have focused on the privacy-preserving genome-wide association studies. We have proposed a privacy-preserving collapsing method using a logistic regression model. It takes advantage of fully homomorphic encryption, secure multiparty computation and multiplicative data masking in order to allow two entities a genomic research unity and a genomic research center to compute this association test on encrypted data without the need to decrypt them. This solution is secure under the honest but curious adversarial model. Because our solution makes no approximations, it achieves exactly the same results as working on clear data. The computation of some operations such as $ln(.)$ function on encrypted data with only addition and multiplication operations has complicated the task for us. The multiplicative data masking solution we have proposed requires a higher computation time because of the passage through binary representation of encrypted values.

During the implementation of our solution, we were limited to the bootstrapping method implemented in HElib, but more recently several improvements which can reduce the computation time [263].

CHAPTER 3

Watermarking of updatable homomorphically encrypted genetic data

As exposed in the previous chapters, genetic data outsourcing induces many critical security issues for data owners especially in terms of individual privacy, data confidentiality and integrity. To protect data confidentiality and privacy from unauthorized users as well as from the cloud service provider (CSP), we have seen in chapter 2 that one common solution consists in encrypting data before their outsourcing. Homomorphic encryption is widely used in this cases, as it allows the protection of data and still allows processing on these data without need of decryption. Beyond data confidentiality, data integrity is another major concern as it can be compromised by several threats such as transmission errors, unauthorized modifications by attackers or by sub-contracted service providers. This is the objective of this chapter where we have proposed a dynamic database watermarking method which allows the protection of integrity of homomorphically encrypted data.

In that context, in this chapter we are interested in taking the point of view of the service cloud provider who may also want to protect data that are under his/her responsibility in order to ensure that these data are not illegal modified by attackers or by malevolent sub-contracted cloud service providers. In addition, as exposed in chapter 1, section 1.2.1.2, existing solutions that allow the protection of outsourced data integrity are all static in the sense that if any modification is occurred, whole database is re-watermarked so as to update the watermark [207].

In this chapter we propose a solution that gives cloud service providers, the capacity of verifying the integrity of homomorphically encrypted databases that are outsourced and maintained at distance by their owners. To do so, we propose to use watermarking in association with homomorphic encryption with the idea of being also able to detect illegally modified data. The method we propose allows verifying the integrity of outsourced databases all along their lifecycle, in a dynamic fashion. This means that it should be possible to perform update operations (tuple additions, tuple suppression or attribute value modifications) without having to re-watermark the whole database. In addition, there is a need for database watermarking scheme that provides a good localization performance comparing to the existing literature. Moreover, our method should be able to work

with homomorphically encrypted data.

In the first time, we will explain how to watermark a static homomorphically encrypted database before describing the complete dynamic crypto-watermarking scheme we propose. In addition, we have conducted a simulation of different possible attacks such as tuple suppression, tuple addition and encrypted attribute value modification, so as to analyse the performance of our solution.

3.1 Overview on crypto-watermarking methods

As we have seen in chapter 1, different methods have been proposed for securing the integrity of outsourced data by mean of watermarking. Most of them are focusing on integrity control by data owners. Initial, crypto-watermarking methods have been proposed for securing multimedia. The objective of these methods is to ensure data confidentiality using encryption while giving access to watermarking based security services such as copyright protection, traitor tracing [264–266] or ensuring integrity control from decrypted/encrypted data [267–269]. Crypto-watermarking schemes can be classified according to the domain where the embedded watermark is available. It can be in the encryption domain [270], in the clear domain [271], [272], or in both domains [11].

Up to now, few methods that combine watermarking and encryption have been proposed so as to protect outsourced databases [207, 208]. The one proposed in [207] focuses on the protection of outsourced databases from the cloud service provider point of view. To do so, the user encrypts the database elements using Order Preserving Encryption (OPE) [209] before being uploaded on the cloud. The cloud service provider (CSP) can embed a watermark into the encrypted data so as to protect the database integrity. To do so, the encrypted database is partitioned into several groups, and the Discrete Cosine Transform (DCT) of each group is computed giving access to DC and AC coefficients. AC coefficients are used for generating the watermark bits based on a cryptographic hash function; watermark bits that are next inserted into the DC coefficients by using the well-known quantization index modulation (QIM) [273]. To get access to the encrypted and watermarked database, the inverse DCT is applied. During the verification stage, database integrity relies on the comparison of the extracted watermark with the recomputed one. This solution allows the verification of the integrity of encrypted database, but does not consider the possibility to update the database. In addition, encryption operations are conducted using an OPE cryptosystem that is known for its security limitations due to some of its deterministic properties [210].

In this chapter, we propose a dynamic database watermarking that allows a cloud service provider to protect and verify the integrity of a homomorphically encrypted database externalized by its owner, even if this one updates his or her data. Our solution allows watermarking of any database which is homomorphically encrypted using any semantically homomorphic cryptosystem (additive, multiplicative or fully). The main objective of our solution is the detection and localization of unauthorized database modifications; such authorized modifications being thus: tuple insertion, tuple suppression or attribute value modifications conducted by the database owner. To conduct watermark embedding in encrypted databases without altering clear data, we take advantage of the semantic security properties of homomorphic encryption cryptosystems. And as we will see, integrity verification is achieved by making possible the watermark extraction from cryptographic

hashes of subsets of homomorphically encrypted attribute values, and the CSP will be able to detect and identify which database elements have been modified.

3.2 Homomorphic encryption cryptosystems

As explained in the chapter 1, homomorphic encryption (HE) [127] allows computation on encrypted data, producing an encrypted result which, when decrypted, corresponds to the one computed on the clear data. Let us recall that if $\mathscr{M}$, $\mathscr{C}$ and $\mathscr{R}$ are the spaces of clear messages, encrypted messages and random integers, respectively. The encrypted version of a message $m \in \mathscr{M}$ is as such as

$$\begin{aligned} E\colon \mathscr{M} \times \mathscr{R} &\mapsto \mathscr{C} \\ (m, r) &\mapsto E[m, r] = E[m, K_p] = c \end{aligned} \tag{3.1}$$

where r is a random integer selected in $\mathscr{R}$. As we will see in section 3.3, we will take advantage of homomorphic and semantic security properties for the insertion of a watermark into encrypted pieces of data. The database watermarking method we propose in this chapter was implemented using the Damgård-Jurik (D-J) cryptosystem and the ElGamal cryptosystem but it can be implemented with all semantic homomorphic cryptosystems in general. We discuss these two cryptosystems in the sequel.

3.2.1 Damgård-Jurik Cryptosystem

The Damgård-Jurik (D-J) cryptosystem [140] is a generalization of the Paillier cryptosystem [135] and its principles are as follow. Let $((g, K_p), K_s)$ be the public key and the private key, respectively, such that

$$K_p = pq \quad \text{and} \quad K_s = LCM((p-1), (q-1)) \tag{3.2}$$

where p and q are two large prime numbers and LCM is the least common multiple function. Let $\mathbb{Z}_{K_p^n} = \{0, 1, \cdots, K_p^n - 1\}$, $\mathbb{Z}^*_{K_p^n}$ denote the set of integers $\in \mathbb{Z}_{K_p^n}$ that have multiplicative inverses modulo K_p^n where $n \in \mathbb{N}^*$. A fast implementation of this cryptosystem, without reducing its security [140], is obtained by choosing:

$$g = 1 + K_p \tag{3.3}$$

Let $m \in \mathbb{Z}_{K_p^n}$ be the message to be encrypted, its cipher-text $c \in \mathbb{Z}^*_{K_p^{n+1}}$ is such that

$$c = E[m, r] = g^m r^{K_p^n} \mod K_p^{n+1} \tag{3.4}$$

where $r \in \mathbb{Z}^*_{K_p}$ is a random integer and it makes the D-J cryptosystem semantically secure. To get access to the message m from c^{K_s}, the recipient has to calculate $K_s m$. To do so, authors of [140] have proposed an iterative procedure to find m from $(1+K_p)^m \mod K_p^{n+1}$. This procedure takes advantage of the Binomial theorem and a function $L(\cdot)$ defined such as $L(b) = \frac{b-1}{K_p}$, function that is applied repeatedly as follow. Taking as input the quantity $a = (1 + K_p)^m \mod K_p^{n+1}$ this

Algorithm 1 Damgard-Jurik algorithm

procedure $F(a)$
  $m \leftarrow 0$
  for $j \leftarrow 1, n$ **do** ▷ $m = m_{j-1}$
    $t_1 \leftarrow L(a \mod K_p^{j+1})$
    $t_2 \leftarrow m$
    for $k \leftarrow 2, j$ **do** ▷ $t_2 = m(m-1)...(m-k+2)$
      $m \leftarrow m - 1$
      $t_2 \leftarrow t_2 * m \mod K_p^j$
      $t_1 \leftarrow t_1 - \frac{t_2 * K_p^{k-1}}{k!} \bmod K_p^j$ ▷ $t_1 = t_1 - C_k^i K_p^{k-1}$
    end for
    $m \leftarrow t_1$
  end for
  return $m \mod K_p^n$
end procedure

algorithm first compute $L(a)$ which gives access to $m_1 = L(a \mod K_p^2) = m \mod K_p$ (using Binomial theorem). Then, by iteratively calculating from $j = 0$ to n, m_j is given by:

$$L(a \bmod K_p^{j+1}) - (C_2^{m_{j-1}} K_p + ... + C_j^{m_{j-1}} K_p^{j-1}) \bmod K_p^j$$

The algorithm achieves $m_n = m \mod K_p^n$ where ($C_k^j = \frac{j!}{(j-k)!k!}$). This procedure we note as the function $F(\cdot)$ is given in Algorithm 1.

The decryption of the cipher-text c into m such as

$$m = F(c^{K_s}) K_s^{-1} \mod K_p^n$$

This cryptosystem has an additive homomorphic property. Considering two plain-texts m_1 and m_2, the homomorphic properties of the D-J cryptosystem are the following ones:

$$E[m_1, r_1] E[m_2, r_2] = E[m_1 + m_2, r_1 r_2] \tag{3.5}$$

$$E[m_1, r_1]^{m_2} = E[m_1 m_2, r_1^{m_2}] \tag{3.6}$$

3.2.2 ElGamal Cryptosystem

The ElGamal cryptosystem was developed by Taher ElGamal in 1984 [130]. It is based on the hardness of the discrete logarithm problem [146] and its principles are as follows. Let G be a cyclic group with n and g as order and generator, respectively. We recall that it is possible to generate all cyclic group elements from the powers of its generator g. For a random integer x chosen in $\mathbb{Z}_n^*$, we calculate $y = g^x$. Then, the public key K_p and the private key K_s are given by $K_p = (G, n, g, y)$ and $K_s = x$, respectively. The cipher-text of the plain-text $m \in \mathbb{Z}_n$ is given by

$$c = E[m, r] = (g^r, my^r) = (g^r, mg^{rx}) = (c_1, c_2) \tag{3.7}$$

where r is an integer randomly selected in $\mathbb{Z}_n$ making the ElGamal cryptosystem semantically secure. To decrypt the cipher-text c, we use K_s and the decryption function D. Then, the plain-text m is obtained such that

$$m = D[c, K_s] = c_2 (c_1^x)^{-1} = mg^{rx} (g^{rx})^{-1} \tag{3.8}$$

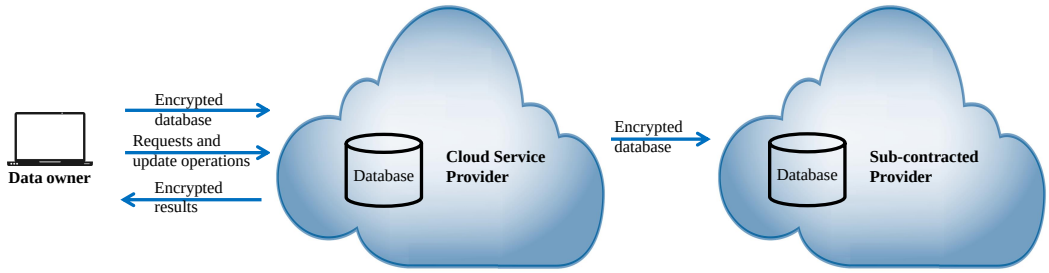

Figure 3.1: Considered encrypted database outsourcing framework.

This cryptosystem has a multiplicative homomorphic property. Considering two plain-texts m_1 and m_2, we have

$$E[m_1, r_1]E[m_2, r_2] = E[m_1 m_2, r_1 + r_2] \tag{3.9}$$

In the next sections, we discuss how semantic security property of homomorphic encryption cryptosystems can be used in order to embed a watermark into encrypted databases with the objective of protecting their integrity.

3.3 Watermarking of homomorphically encrypted databases

In this section, we first introduce the encrypted database outsourcing framework we consider before presenting how to dynamically watermark an encrypted database along its lifecycle.

3.3.1 Database outsourcing framework

As shown in Fig.3.1, in our framework a user or data owner securely outsources his database into the cloud, maintained by a cloud service provider (CSP). The database elements are independently encrypted using a HE cryptosystem before being uploaded to the cloud. This encryption task can be made with the help of one of the HE cryptosystems depicted above (fully or partially being additive or multiplicative). By doing so, the user can ask CSP to conduct some data treatments or analysis on his data while preserving their confidentiality.

Different security issues have to be considered in such a context. In a first time, it is common to assume that CSP is honest but curious. That is to say, it honestly stores and follows all data processing or updating operations requested by the data owner but may try to infer user's data. In order to ensure data confidentiality, we assume that all data stored in the cloud have been encrypted homomorphically by their owners. To tackle the problem of integrity of data which is the objective of this chapter. Herein, CSP is authorized to conduct storage and/or processing of databases outsourced by their owners, even with the help of sub-contracted service providers. From the point of view of CSP, data may face many attacks from external attackers as well as from malicious or not well secured sub-contracted clouds. There is thus an interest for CSP to protect homomorphically encrypted data that are under his responsibility in terms of integrity. To do so, we propose a crypto-watermarking scheme which combines watermarking and encryption so as to

allow the integrity protection of encrypted database under the constraints: *i)* users can update their data during time; *ii)* users' data are not modified by the watermarking process as it usually does. Thus, contrarily to common database watermarking schemes, additions, deletions or modifications of tuples and/or attribute values are stated as authorized modifications while illegal modifications caused by attackers or system errors (e.g., storage or transmission errors) should be detected. As we will see, our solution responds these constraints by taking advantage of the homomorphic and the semantic security properties of HE cryptosystems.

3.3.2 Outsourced HE encrypted database

As stated above, in our framework, encrypted outsourced data are supposed to be stored in relational databases. As defined in section 1.2.1.2, we note DB as relational database composed of a list of T tables. If $t_i.a_j$ is the j^{th} attribute of the i^{th} tuple in the database, the encrypted version DB_e of the database DB is obtained by independently encrypting the values $\{t_i.a_j\}_{i=1,\cdots,N;j=1,\cdots,M}$ using an HE cryptosystem as follows.

$$c_{ij} = E[t_i.a_j, r_{ij}] \tag{3.10}$$

where $r_{ij} \in \mathscr{R}$ is a random integer associated to $t_i.a_j$. Notice that in the case the Damgård-Jurik cryptosystem is used, r_{ij} is taken in $\mathbb{Z}^*_{K_p}$ and (3.10) becomes

$$c_{ij} = E[t_i.a_j, r_{ij}] = g^{t_i.a_j} r_{ij}^{K_p^n} \mod K_p^{n+1} \tag{3.11}$$

In the case where the ElGamal cryptosystem is exploited, r_{ij} is taken in $\mathbb{Z}_n$ and (3.10) is such as

$$c_{ij} = E[t_i.a_j, r_{ij}] = (g^{r_{ij}}, t_i.a_j g^{r_{ij}K_s}) \tag{3.12}$$

In the sequel, we first explain how to watermark such a static encrypted database before introducing our complete dynamic crypto-watermarking scheme.

3.3.3 Static database watermarking for homomorphically encrypted data

The solution we propose allows the embedding into an encrypted database DB_e of a watermark W, a proof of integrity, that will be available in the encrypted domain. As shown in Fig. 3.2, its architecture relies on two main procedures: database protection and integrity verification of the database. The protection stage, see Fig. 3.2a, is performed into three steps: *i)* a secret database reorganization step where tuples of DB_e are rearranged into the database DB_e^r based on the secret watermarking key K_w; *ii)* a watermark embedding step which consists in embedding W into DB_e^r in order to produce the database DB_e^{wr}; *iii)* a back database reorganization step in which DB_e^{wr} is reorganized in order to get access to the watermarked and encrypted database DB_e^w. The verification stage is conducted in a similar way (see Fig. 3.2b). Let $\widehat{DB}_e^w$ be a protected database, to verify the integrity of this database, we first perform its secret reorganization based on K_w. Then, the watermark $\widehat{W}$ is extracted and compared to the watermark W. If $\widehat{W}$ and W are different, the original database was illegally modified. In addition to this, the verification stage allows us the identification of the encrypted attribute values or element of $\widehat{DB}_e^w$ that have been altered. In the sequel, we enter into the details about these different stages.

3.3.3.1 Database protection

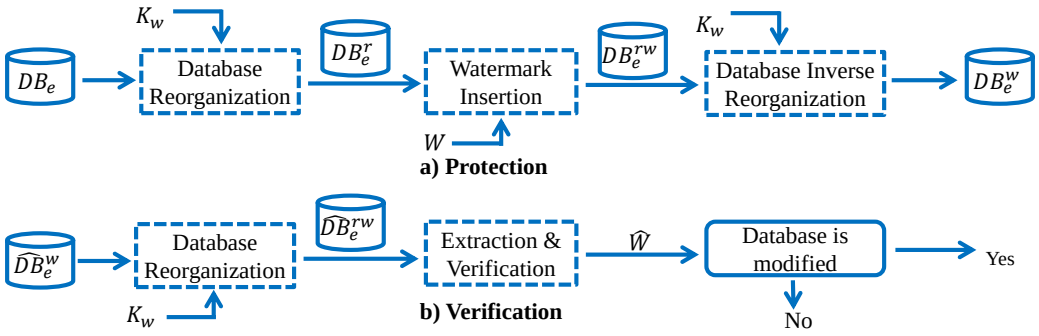

Figure 3.2: System architecture of the proposed method. K_w, W, $\hat{W}$ represent the secret watermarking key, the inserted watermark and the recovered watermark, respectively.

In case of a static database, this procedure is constituted of three main steps (see Fig. 3.2):

1. **Secret database reorganization**: The objective of this step is to ensure that an unauthorized user cannot access to the watermark W. The basic principle of this step is to secret reorganize the database DB_e into DB_e^r using the secret watermarking key K_w. To do so, each database tuple is associated to secret cryptographic hash such that

$$h_i = hash(t_i) = hash(K_w || E[t_i.PK, r_{iPK}]) \tag{3.13}$$

   where: h_i is the hash of the tuple t_i and $t_i.PK$ its primary key, "$||$" is the concatenation operator, and $hash$ a cryptographic hash function such as Secure Hash Algorithm 2 (SHA-2) [274]. Tuples are simply reorganized in the ascending order of their hash values. The security of this step relies on the diffusion and collision properties [275] of the cryptographic hash function that is used, as well as on the knowledge of the watermarking key K_w.

2. **Watermark insertion into the reorganized database**: In this step, a binary watermark W is inserted into the reorganized database DB_e^r. More clearly, one bit of the watermark is embedded into the hash value of a subset constituted of homomorphically encrypted attribute values of DB_e^r. Let us consider a reorganized encrypted database constituted of k subsets and as watermark W, a binary sequence of k bits uniformly distributed ($W = \{b_l\}_{l=1,\cdots,k}, b_l \in \{0,1\}$) and secretly generated using the watermarking key K_w with the help of a random number generator. The verification of the integrity of the database will relies on the correct extraction of W from the hash values of the attribute subsets. The interest in working with subsets rather that with the whole database, is that it becomes possible to localize and identify which database parts or attribute values have been illegally modified. This watermark embedding step relies on the two following sub-steps:

   - *Database partitioning into attribute value subsets* - The secretly encrypted and reorganized database DB_e^r is divided into k overlapping "subsets" $\{B_l\}_{l=1,\cdots,k}$. We conduct this database partitioning with the guarantee that each subset has at least one element that is shared with other subset. Fig. 3.3 gives an example of such a table partitioning in the case of subsets of 3×3 elements. It can be seen that for one subset or

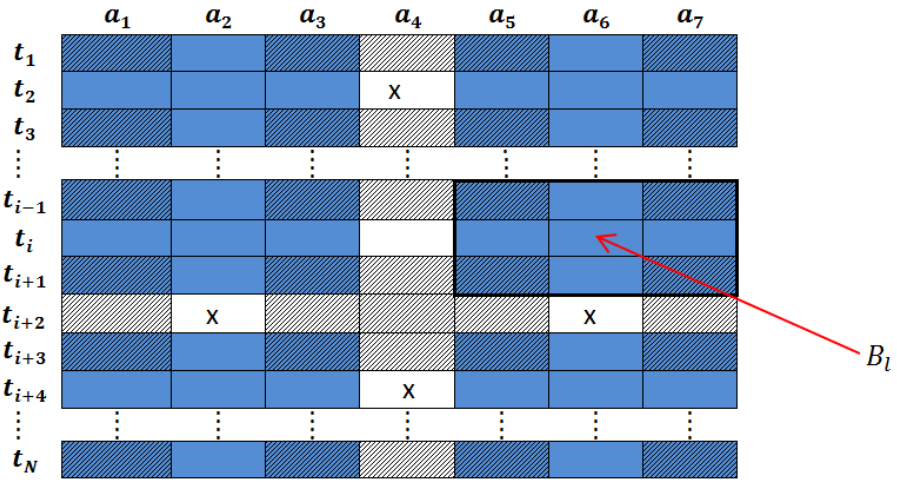

Figure 3.3: Partitioning of an encrypted and reorganized database DB_e^r into overlapping and non-overlapping subsets or blocks of 3×3 encrypted attributes values. Blue and dashed areas represent overlapping subsets. B_l is one subset and $E[t_i.A_j, r_{ij}]$ is its center element. Standalone encrypted attribute values, identified by black crosses are regrouped into independent and non-overlapping subsets.

"block" its center element is not shared with the other overlapping subsets. In general, this partitioning can be made in different ways. However, as we will see in section 2.4, it strongly impacts our scheme performance in terms of detection and localization precision.

- *Embedding of one watermark bit into one attribute subset* - Each subset B_l is then watermarked into B_l^w by inserting one bit b_l of W in B_l such that

$$b_l = hash(B_l^w)_v = s_v \tag{3.14}$$

where s_v represents the v^{th} bit of the cryptographic hash S of the subset B_l^w, i.e., $S = hash(B_l^w)$. The value of v is chosen based on the secret watermarking K_w. As it is extremely difficult to predict the output of a cryptographic hash function (e.g., SHA-2) for a given input, an iterative procedure is used so as to watermark the subset B_l into B_l^w. It is the center attribute value of the subset B_l (i.e., $E[t_i.a_j, r_{ij}]$, see Fig. 3.3) that is modified for bit insertion in the subset using the function f defined as

$$\begin{gathered} f : \mathscr{C} \times \mathscr{M} \mapsto \mathscr{C} \\ (E[t_i.a_j, r_{ij}], e) \mapsto f(E[t_i.a_j, r_{ij}], e) = E[t_i.a_j, r_{ij}] \otimes E[e, r] = E[t_i.a_j, r_{ij} \oplus r] \end{gathered} \tag{3.15}$$

where $\mathscr{C}$, $\mathscr{M}$ are the spaces of the encrypted data and clear, data respectively, and r is random integer taken in $\mathscr{R}$. We recall that the operators $\otimes$ and $\oplus$ depend on the exploited HE cryptosystem. Moreover, when E is an additive homomorphic encryption function (resp. a multiplicative homomorphic encryption function), then $e = 0$ (resp. $e = 1$). More clearly, we take advantage of the semantic property of HE cryptosystems to modify the encrypted value of an attribute without modifying the clear value of the attribute. Our iterative procedure used to modify the center element of each subset

Algorithm 2 Iterative procedure for the modification of the center element of an encrypted subset B_l for embedding of one watermark bit.

1: **INPUT**: A subset B_l, A watermark bit b_l, HE cryptosystem with E its encryption function
2: **procedure** SUBSETWATERMARKING(B_l, b_l)
3: $B_l^w \leftarrow B_l$
4: **while** $b_l \neq s_v = hash(B_l^w)_v$ **do**
5: $\alpha = rand(.)$ $\%$ $rand(.)$ is a uniform random function in R
6: $f(E[t_i.a_j, r_{ij}], e) = E[t_i.a_j, r_{ij}] \otimes E[e, \alpha]$ $\%$ $E[t_i.a_j, r_{ij}]$ is the center element of B_l^w
7: $E[t_i.a_j, r_{ij}] \leftarrow f(E[t_i.a_j, r_{ij}], e)$
8: **end while**
9: **return** B_l^w
10: **end procedure**

is illustrated in Algorithm 2. In this chapter, the secure hash algorithm 2 (SHA-2) is used as cryptographic hash function. Due to its "strength", there is one chance in two to insert one bit b_l at each iteration, i.e., to have s_v equal to b_l (see (3.14)). Algorithm 2 can be refined depending on the cryptosystem used to encrypt the database. In the case of the Damgård-Jurik cryptosystem, the value of $f(E[t_i.a_j, r_{ij}], e)$ in step 7 is such as

$$f(E[t_i.a_j, r_{ij}], e) = E[t_i.a_j, r_{ij}]E[0, \alpha] = E[t_i.a_j, r_{ij}\alpha] = g^{t_i.a_j}(r_{ij}\alpha)^{K_p^n} \mod K_p^{n+1} \tag{3.16}$$

where $\alpha \in \mathbb{Z}_{K_p}^*$ is a random number. On the other hand, if the database has been ElGamal encrypted, step 7 of Algorithm 1 becomes

$$f(E[t_i.a_j, r_{ij}], e) = E[t_i.a_j, r_{ij}]E[1, \alpha] = E[t_i.a_j, r_{ij}+\alpha] = (g^{r_{ij}+\alpha}, t_i.a_j g^{(r_{ij}+\alpha)K_s}) \tag{3.17}$$

where $\alpha \in \mathbb{Z}_n^*$ is a random integer.

3. **Back reorganization of encrypted and watermarked database**: Once all subsets of the database DB_e^{wr} have been watermarked, DB_e^{wr} is reorganized back in order to obtain the encrypted and watermarked database DB_e^w.

3.3.3.2 Extraction of the watermark and integrity verification of the database

Watermark extraction for controlling the integrity of a protected database is performed in a similar way as in the protection procedure. Therefore, let $\widehat{DB}_e^w$ be a suspicious database, its integrity verification is conducted accordingly following two steps:

1. The database $\widehat{DB}_e^w$ is first reorganized into $\widehat{DB}_e^{rw}$ using the secret watermarking key K_w. After that, $\widehat{DB}_e^{rw}$ is divided into multiple subsets.

2. The cryptographic hash values of all subsets are computed and one watermark bit is extracted from each of them using (3.14), extracted bits correspond to the watermark $\widehat{W}$.

Once $\widehat{W}$ obtained, it is compared to the *a priori* known watermark W, i.e., the watermark that has been originally embedded. Any differences will indicate if the database has been illegally modified. In addition, it is possible to identify and localize altered subsets. Beyond, in general and as we will see in section 2.4, the protection we proposed allows the detection of different malicious attacks [191] such as

- *Tuple addition attack* − it corresponds to the unauthorized introduction of tuples the attribute values of which are encrypted based on the public key of the database owner.
- *Encrypted attribute value modification attack* − herein, an attacker performs homomorphic operations in order to falsify or damage some database element values.
- *Tuple suppression attack* − where some tuples are illegally removed from the database.

3.3.4 Dynamic database watermarking for updatable encrypted data

During the database lifecycle, tuples or attribute values of the database can be remotely added, removed or modified. These tasks are conducted by CSP based on data owner's requests. As stated previously, these requests are considered as authorized. Unauthorized modifications we want to detect are of same nature but conducted by malicious entities (e.g. malevolent data storage subcontractors, badly securely data storage provided by CSP subcontractor) or may result from errors of storage or communications.

With the previous static database watermarking scheme, it is necessary to re-watermark the whole database if any database element is updated. Such complete re-watermarking has several limitations such as computation overhead, etc. In this work, we thus propose a dynamic database watermarking solution that allows the protection of the database integrity on the fly while still making possible to localize illegal database modifications. To achieve this goal, the challenging issue is to maintain a coherent watermark at each update operation. To do so, our dynamic watermarking scheme while having verification and protection procedures quite similar to the previous static scheme, takes advantage of a secure journal table J_t that contains historical details about all suppressed or added tuples.

To make more clear how our proposal works, let us consider an already protected database DB_e^w along with its secure journal table J_t. As illustrated in table 3.1, record in J_t corresponds to one update of one tuple in DB_e^w (i.e., addition and suppression). It contains: the update order of the tuple; the identifier of the added or suppressed tuple, this identifier can for instance be the encrypted primary key $E[t_i.PK, r_{iPK}]$; the executed action the tuple undergone (addition (A) or tuple suppression (S)); and, the watermark bits w that were embedded into the tuple.

As we will see the update of the attribute values of existing tuples does not require the addition of specific information in J_t. Anyway, the journal table J_t is organized depending on the chronological order of the database updates and will be very helpful for secretly reorganizing database element and, moreover for maintaining watermark coherence, allowing watermarking on the fly and verifying database integrity (detection and localization). Being a sensitive element, the journal

Table 3.1: Example of a journal table J_t with some records, where A and S indicate tuple addition and tuple suppression, Id_i is the identifier of the i^{th} tuple concerned by the action, w corresponds to the bits of the watermark embedded after the suppression or addition of the i^{th} tuple in the database.

Update Order (UO)	Executed action (EA)	Tuple identifier (Id)	Inserted watermark (w)
1	A	Id_1	w_1
2	A	Id_2	w_2
5	S	Id_5	w_5

content is secret and should only be known from CSP. To do so, J_t record elements are encrypted. We will discuss more about the security of the journal table in section 3.4.4.

In the sequel, we detail our solution by presenting in a first time how it works when: *i)* new tuples are added; *ii)* some tuples are suppressed; *iii)* authorized encrypted attribute value modifications are conducted.

3.3.4.1 Database watermarking on the fly in the case of new tuple addition

Let DB_e^w be an encrypted and watermarked database that only contains two tuples t_1 and t_2 as illustrated in Fig. 3.4. When a data owner wants to add one tuple in the database, he or she homomorphically encrypts it before sending it to CSP. Let us assume that the new homomorphically encrypted tuple CSP receives is t_i. CSP adds it to DB_e^w while performing the watermark update as follows.

1. CSP performs the decryption of the journal table J_t where the tuples are organized in their chronological order.
2. Following our example with subsets of 3×3 elements, CSP uses J_t to identify the two last tuples or lines that were previously added to DB_e^w (see Fig. 3.4a) and places the new tuple at the last position.
3. CSP computes the corresponding attribute subset partitioning as illustrated in Fig. 3.4b. Still working with subsets of 3×3 elements, it is possible to find the partition associated to t_i based on its index i (this value corresponds to the update order information in the journal J_t) and the index j that corresponds to the position of attribute values of t_i:
   - If $i = 0$ or $= 1 \mod 4$, then the subsets associated to t_i are centered on the encrypted attribute values $t_i.a_j$ such as $j = 0 \mod 4$.
   - If $i = 2$ or $3 \mod 4$, the subsets associated to t_i are centered on the elements $t_i.a_j$ that correspond to $j = 2 \mod 4$.

   Let us recall that it is the subset centered element we modify in the encrypted domain for message embedding.
4. In the last step, CSP watermarks the previous set of tuples based on the two following substeps:

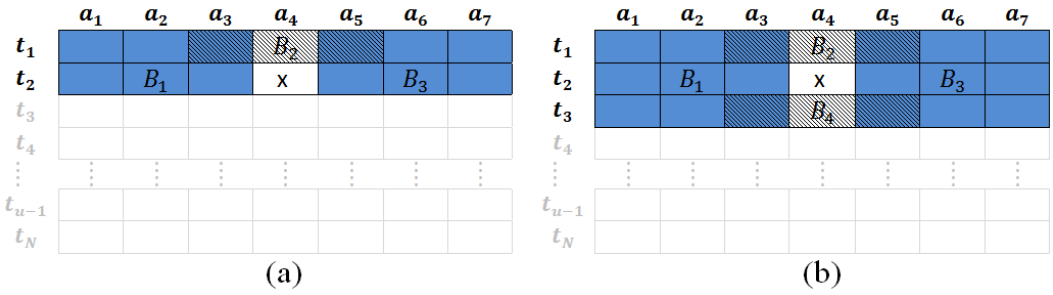

Figure 3.4: **(a)** Protected database initialized with two tuples, the elements of which are encrypted independently, and where subsets are constituted of 3×3 elements. Blue areas, B_1 and B_3 correspond to incomplete subsets while hashed grey areas correspond to elements of the subset B_2. Empty areas, where new tuples will be added, correspond to the tuples t_3, t_4 and t_5. **(b)** Protected database after the addition of the new tuple t_3 into the database. In this situation, B_1 and B_3 in blue correspond to complete subsets while B_4 is a new subset of three elements only. In both cases, black crosses represent the standalone encrypted attribute values.

a) For already existing subsets - CSP extracts the already watermarked bits from pre-existing and incomplete subsets, i.e., as for instance from B_1 in Fig. 3.4a before tuple addition, and re-insert them into the new subsets after the addition of new tuple (see B_1 in Fig. 3.4b).

b) For newly created subsets such as B_4 in Fig. 3.4b, CSP uses the secret watermarking key K_w so as to generate a sub-watermark w, a sequence of bits, and insert one bit per new subset.

c) Finally, CSP adds to J_t the record R_{t_i} such that $R_{t_i} =< 1, A, Id_i = E[t_i.PK, r_{iPK}], w_i >$, where w_i corresponds to the newly embedded watermark bits in the database, and re-encrypts J_t.

3.3.4.2 Database protection on the fly when one tuple is suppressed

In the case a data owner proceeds to the suppression of one tuple t_i from the database DB_e^w, our dynamic database watermarking scheme works as follows:

1. CSP first performs the decryption of J_t and looks for the position of t_i in the database as well as of its two neighbors t_{i+1} and t_{i-1} in the case of subsets of 3×3 elements.

2. CSP then computes the subset partition as for tuple addition (see previous section) and extracts pre-existing watermark bits from these database subsets.

3. CSP substitutes the suppressed tuple by a "virtual tuple" that is to say an empty record and re-inserts extracted watermark bits in the subsets by modifying one of their elements. Two distinct situations should be considered when conducting their re-watermarking process:

   a) In the cases the suppressed tuple t_i includes the center attribute values of some database subsets, as illustrated in Fig. 3.5a, with the subsets B_l^w and B_{l+1}^w, the corresponding watermark bits are re-inserted by modifying one of the encrypted attribute values

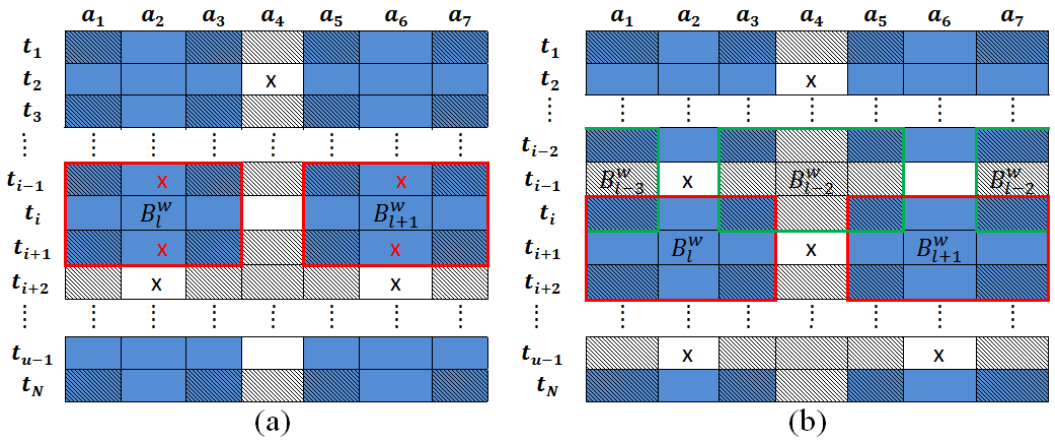

Figure 3.5: **(a)** Protection of the database when the suppressed tuple contains center elements of subsets. Outlines in red indicate database subsets that are concerned by the suppression of the tuple t_i, while red crosses represent database element values that will be modified by the re-watermarking of the data subsets. **(b)** Protection of the database if the suppressed tuple does not include center elements of subsets. Green and red outlines correspond to the subsets that are concerned by the suppression of the tuple t_i while black crosses represent single encrypted attribute values.

of the tuples t_{i-1} and t_{i+1} that are out of the intersection of two subsets (see elements marked by red cross in Fig. 3.5a for the subsets B_l^w and B_{l+1}^w). This encrypted attribute value modification is commonly performed using the Algorithm 1 presented in section 3.3.3.

b) In the case the suppressed tuple t_i does not include center elements of subsets, as depicted in Fig. 3.5b, watermark bits are normally re-inserted into subsets.

4. Finally, CSP constitutes the record R_{t_i} indicating that t_i has been suppressed and this adds it to J_t. J_t is then encrypted.

3.3.4.3 Protecting database on the fly when encrypted attribute values are modified

Let us consider the following scenario where one element, e.g., $E[t_i.a_j, r_{ij}]$, of the protected database DB_e^w is modified by its owner user. To update the watermark on the fly, CSP conducts the following steps:

1. It first decrypts J_t and identifies the position of the tuple t_i in the database as well as of its two neighbors (t_{i-1} and t_{i+1}) or four neighbors (t_{i-2}, t_{i-1}, t_{i+1} and t_{i+2}) depending on its position.

2. It then identifies the subsets associated to the attribute value that is modified by the user.

3. CSP extracts the pre-existing watermark bits that were inserted into the subsets to which $E[t_i.a_j, r_{ij}]$ belongs. It then updates the database element and re-embeds the watermark bits into the subsets using algorithm 2, as normally.

It is important to notice that because this kind of modification does not add or suppress a tuple in the database, the journal table J_t is not updated.

3.3.4.4 Watermark extraction and verification of database integrity

Let us consider CSP want to verify the integrity of an encrypted and watermarked database $\widehat{DB}_e^w$ so as to detect if this one has been tampered or not. It is important to recall that this verification process is conducted at the level of database element subsets and not directly at the level of a tuple or of a database element. To do so, while still considering our example with 3×3 subsets, CSP performs the following steps:

1. The journal table J_t is first decrypted.
2. Integrity verification starts by verifying the latest updated tuple, i.e., added or suppressed, and continues with the previous one and so on, going back in the history of the database. For each tuple t_i, CSP conducts the integrity verification taking into account its associated record R_{t_i} in J_t as follows:
   a) in the case the action EA performed to t_i is suppression, then:
      i. CSP adds an empty tuple at the position of t_i in the database;
      ii. to verify the integrity of the subset around t_i CSP identifies from J_t the neighbors of t_i in the database while distinguishing two situations depending on the subset partition around t_i:
         - If the suppressed tuple t_i includes element centers of subsets as in Fig. 3.5a, CSP retrieves the tuples t_{i-1} and t_{i+1} so as to re-build subsets that were sharing elements with t_i.
         - If the suppressed tuple does not contain subset element centers, then CSP has to retrieve the tuples t_{i-2}, t_{i-1}, t_{i+1} and t_{i+2} so as to rebuild subsets as in Fig. 3.5b.

      Once the subsets identified, watermarked bits are extracted using (3.14) and compared with the ones reported in J_t i.e., w_i. Any difference will indicate a loss of integrity in this sub-region of the database and will help to localize the position of the tamper accordingly to the subset partitioning.
   b) in the case the tuple t_i has been added, then:
      i. CSP retrieves from J_t the two first neighbors of t_i, that is to say t_{i-1}, t_{i+1}.
      ii. CSP computes the subset partition around t_i and extracts from each subset one watermark bit. Extracted bits are next compared to the ones stored in J_t, i.e., w, in order to detect if some subsets have been illegally altered.

From this standpoint, one can consider that the integrity of a tuple t_i is verified if all the subsets it belongs to are detected as un-modified with the above procedure. The same reasoning can be followed from the database point of view that is to say that the database integrity is considered as broken if even one of its subsets is detected as tampered.

Table 3.2: Some tuples from our genetic database. One record or tuple contains information about one variant at a given position in the genome and attributes I_k represent the individuals.

$chrom$	pos	ref	alt	$gene$	I_1	I_2	$\cdots$	I_k	$\cdots$	I_{55}
1	861261	G	A	SAMD11	0	1	$\cdots$	2	$\cdots$	1
1	871334	G	T	SAMD11	2	1	$\cdots$	1	$\cdots$	0
9	135140020	A	G	SETX	0	0	$\cdots$	0	$\cdots$	1
21	16335402	C	T	NRIP1	1	0	$\cdots$	0	$\cdots$	1

It is important to notice that the previous detection and verification procedures allow CSP to detect unauthorized modifications of homomorphically encrypted attribute values. Other unauthorized modifications, such as tuple suppression or tuple addition, will be detected based on J_t. Indeed, illegally added tuples will appear as extra data for CSP, while illegally suppressed tuples will not be found in the database $\widehat{DB}_e^w$ by CSP.

3.4 Experimental results and performance analysis

In this section, we evaluate the performance of our dynamic watermarking in terms computation complexity, watermarking capacity, tampering detection and localization considering one real genetic database.

3.4.1 Test database

Our dynamic database watermarking scheme was experimented using a real genetic relational database constituted of one relation of 4000 tuples. This one contains information about genetic variants of 55 individuals. Such a database is used by geneticists in genome-wide association studies (GWAS) [6] so as to establish the relationships in-between genetic variants and diseases. One tuple corresponds to one variant and is constituted of 60 attributes. The five first ones give information about the genetic variant and correspond to: the name of the chromosome ($chrom$) to which belongs the variant; the position of the variant (pos) in the chromosome; the reference allele (ref); the alternative allele (alt); and, the name of the gene ($gene$) in which the variant belongs. For one individual and one variant, the genotype is an integer value stating if the individual allele equals the reference allele (value "0") or not. In the case it is different the genotype takes its value in the range $\{1, \cdots, U\}$ in case of U possible alternative alleles. In the following, we consider the encrypted form of the attribute pos as tuple primary key $t_i.PK$ because it uniquely identifies each database tuple.

Two homomorphic encryption cryptosystems were experimented: the Damgård-Jurik cryptosystem which is an additive HE cryptosystem and ElGamal as multiplicative HE cryptosystem. For both cryptosystems, we choose public and private keys of 1024 bits in order to ensure a high security level. In the following experiments, we still consider subsets of 3×3 encrypted attribute values or elements. Thus, in the case of a static database, 33447 subsets can be defined on the above test database of 4000×60 elements. In the case of dynamic watermarking J_t is encrypted

	$a_1 = PK$	a_2	a_3	a_4	a_5	a_6
t_1	15168	52405	59972	51875	2703	52916
t_2	22049	10898	34196	20342	42372	44076
t_3	1604	14573	41918	30450	62456	32342
t_4	19744	47492	42201	23564	53753	27252
t_5	14744	40014	21014	74121	24114	47447
t_6	4709	52027	46237	61211	44925	21038

Figure 3.6: A simple example of a tuple addition and a tuple suppression attacks. Red tuple is supposed to be suppressed by an attacker while blue tuple represents illegally addition by an attacker

using Advanced Encryption Standard (AES) [276]. In the following experiments, our static and dynamic watermarking schemes were implemented using C/C++ with GMP library on a computer equipped with 8 GB RAM running on Ubuntu 18.04 LTS. The cryptographic hash function used in all the following experiments is SHA-2 [274].

3.4.2 Database watermarking attacks

As introduced in section 3.3.3.2, three types of database watermarking attacks are considered so as to evaluate the efficiency of our dynamic watermarking schemes: the *"tuple suppression attack"*, the *"tuple addition attack"* and the *"Encrypted attribute value modification attack"*. These attacks can be caused by data transmission errors or attacks in the system. Let us recall that modifications requested by the database owner are authorized

3.4.2.1 Tuple addition and tuple suppression attacks

In these types of attacks, an intruder adds or deletes some random tuples in the database. From here on and for sake of simplicity, let us assume that an attacker deletes or adds one tuple in the protected database DB_e^w. As described in section 3.3.4, the verification of database integrity is conducted using pieces of information that are reported in the journal table J_t. If an attacker has added one tuple in the database, the tuple identifier will not be found in J_t by CSP, during the verification procedure. More clearly, at the end of the procedure, CSP will identify an extra tuple illegally added. To give a simple example, let us consider the protected database presented in Fig. 3.7. If an attacker illegally adds the tuple t_5 (see Fig. 3.6), this addition will impact all subsets around this tuple, and this modification is reported during the verification due to the fact that its identifier 14744 is not found in J_t in Fig. 3.8. The situation where one tuple has been illegitimately deleted from a protected database DB_e^w is quite similar. The suppressed tuple, while existing in J_t record, will not be retrieved by CSP in DB_e^w. In order to continue the verification of the integrity of the whole database, an empty or virtual tuple is just added in $\widehat{DB}_e^w$ by CSP (i.e., the attacked version of DB_e^w). Notice that one consequence of such a deletion is that all subsets to which belongs the suppressed tuple will be considered as unauthentic. As an example, let consider the protected database depicted in Fig. 3.7. If an attacker deletes the tuple t_3 from the database, the verification stage will give us an error as the corresponding identifier 1604 is no longer associated to any tuple (see Fig. 3.8). Using the journal table J_t, especially the tuple identifiers that it stores,

	$a_1 = PK$	a_2	a_3	a_4	a_5	a_6
t_1	15168	52405	59972	51875	2703	52916
t_2	22049	10898	34196	20342	42372	44076
t_3	1604	14573	41918	30450	62456	32342
t_4	19744	47492	42201	23564	53753	27252
t_5	4709	52027	46237	61211	44925	21038

Figure 3.7: A simple example of a protected database before its attack.

UO	EA	Id	w
1	A	1604	001
2	A	19744	101
3	A	4709	101

Figure 3.8: A simple example of journal table.

we have a detection rate of 100% for tuple addition and tuple suppression attacks.

3.4.2.2 Attribute value alteration attack

In this kind of attack, the attacker performs some homomorphic operations on some database elements so as to illegally modify the database. Indeed, as the database is homomorphically encrypted using the database owner's public key, and that this key is assumed to be known from everyone, an attacker can make some operations in order to modify encrypted attribute values of the database. In the integrity verification procedure, as stated in section 3.3.4, one can distinguish three levels of control: the subset level, the tuple level and the database level. In a first time, let us consider the modification of only one database element. At the subset level, based on the considered database partitioning (see Fig. 3.3), if the modified encrypted attribute value is not at the intersection of two overlapping subsets, the probability the modification is not detected is $1/2$. This is due to fact there is one in two chances that the watermark bit embedded in the cryptographic hash value of subset changes (see section 3.3.3). On the other hand, if the modification consists in the alteration of an encrypted attribute value in the intersection of two subsets, then the non-detection probability equals to $1/4$. Consequently, the probability that the alteration is not detected in one subset for any of the two previous cases is bounded by $1/2$. Regarding the database level, if the alteration consists of z subsets of the database DB_e^w, the detection rate is bounded by

$$P = 1 - (\frac{1}{2})^z \tag{3.18}$$

Notice that this probability converges rapidly to 1 with the increase of z. In order to verify the previous results, we have randomly modified a given percentage of encrypted attribute values in DB_e^w: 0.0004% (or equivalently only one element), 0.001% (or equivalently three elements), 0.004% (ten elements), 12.5%, 25%, 50%, 75%, and 100%. To do so, a random number b is selected and its encrypted version is multiplied by the original element to modify in the database. More clearly, let us assume that the element $E[t_i.a_j, r_{ij}]$ has been selected. Its attacked version is given by

$$E[t_i.a_j b, r_{ij} + r_{b_e}] = E[t_i.a_j, r_{ij}]E[b, r_{b_e}] \tag{3.19}$$

Table 3.3: Experimental and theoretical global detection rates for attribute value alteration attack. DJEDR, EEDR, TDR and PE represent the Damgård-Jurik experimental detection rate, the ElGamal experimental detection rate, the theoretical detection rate and the percentage of elements modified by an attacker. All experimental detection rates are given in average after 25 trials.

EP	0.0004%	0.001%	0.004%	12.5%	25%	50%	75%	100%
DJEDR	64%	88%	100%	100%	100%	100%	100%	100%
EEDR	68%	88%	100%	100%	100%	100%	100%	100%
TDR	50%	87.5%	99.9%	100%	100%	100%	100%	100%

for ElGamal HE cryptosystem, where r_{b_e} is chosen in $\mathbb{Z}_n$. For Damgård-Jurik HE cryptosystem, this value is given by

$$E[t_i.a_j + b, r_{ij} r_{b_d}] = E[t_i.a_j, r_{ij}] E[b, r_{b_d}] \tag{3.20}$$

where r_{b_d} is randomly selected in $\mathbb{Z}^*_{K_p}$. Table 3.3 gives theoretical and experimental detection rates. These results are given in average after 25 trials for both HE cryptosystems ElGamal and Damgård-Jurik. It can be seen in the Figure 3.9 that, the successful detection depends on the quantity of altered encrypted attribute values. If an attacker modifies only one encrypted attribute value in the database, the detection rate is of 64% for Damgård-Jurik HE cryptosystem, and 68% for ElGamal HE cryptosystem. The greater the number of modified elements, the higher the detection rate is. It can be also seen that experimental results confirm the theoretical ones.

3.4.2.3 Comparison of our method with other approaches

In this section, we focus on the performance of our scheme compared to database watermarking methods from the state of the art, that have been developed for the purpose of verifying the integrity of databases. Before entering in the details of this discussion, let us remind that the main objective of our watermarking scheme is to allow CSP to protect in terms of integrity encrypted databases that are remotely outsourced and updated by their owners. It allows detection and localization of unauthorized modifications of database elements such as tuple insertion, tuple suppression or encrypted attribute value modification.

As stated in chapter 1, Section 1.2.1.2 many watermarking methods, especially fragile watermarking schemes, have been proposed for securing relational databases in terms of integrity [174,185,186,196–205]. Some of them like methods [199], [185], [174], [202], [205] only work at the database level. [196], [203] and [186] can detect and localize database modifications but their localization precision is limited to group of tuples. Furthermore, in the case of the suppression of tuples, these schemes are going to detect it, but with no capacity to identify the exact number of suppressed tuples. [200] overcomes this issue but its localization remains limited to group of tuples. Our scheme can work on subpart of tuples. A few schemes, like [197], [198], [204] and [201] allows detecting any modification and localize them up to the attribute tuple level but they only work on static databases and not with encrypted databases. We have also seen in section 3.1 that, to the best of our knowledge, the method proposed in [207] is the only one that combines encryption and watermarking so as to protect outsourced databases. Even though it can be used for

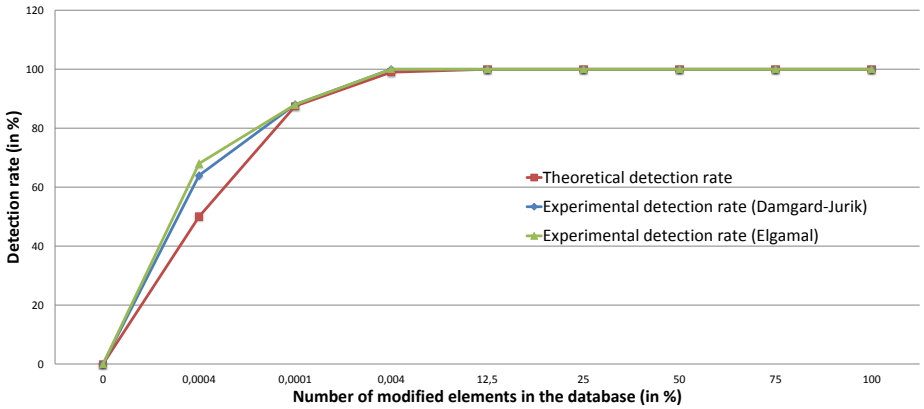

Figure 3.9: A graphic comparison of theoretical and practical detection rates for Damgård-Jurik and ElGamal cryptosystems

verifying the integrity of group of tuples in an encrypted database, it does not work in a dynamic way. The database has to be entirely re-watermarked at each update. In addition, it is based on the order preserving encryption cryptosystems, cryptosystems that are known to have several limitations in terms of data security due to the fact that they are deterministic [277]. Contrarily to all of the above methods, our method can properly detect and localize any unauthorized modification of subsets of encrypted attribute values in a static or dynamic database. In the latter case, watermarking and integrity verification processes are conducted along the database lifecycle and on the fly without having to re-watermark the whole database. That is not the case of the previous methods. Our scheme watermarks homomorphically encrypted databases taking advantage of the inherent properties of HE cryptosystems without altering user's data. As mentioned it can be deployed with partially homomorphic encryption algorithms, additive or multiplicative. It can obviously be implemented with fully homomorphic encryption cryptosystems, at the price however of a computation complexity increase [154]. It is important to notice that the performance of our scheme strongly depends on the computation complexity of such cryptosystems. Even if the data integrity checking is fast, the watermarking process is time consuming. We thus recommend protecting block of tuples at once rather than working tuple by tuple. In the case of a large database, we also suggest to divide it into small tables, associating consequently one journal to each table.

At least, our scheme is used by CSP in order to protect the data of his/her clients while other schemes from the literature are under the control of the database owner to protect the integrity of the data he/she externalized. Nevertheless, our scheme can be used by a data owner to protect his data locally, i.e., on his own server/computer.

3.4.3 Computation complexity and watermarking capacity performance analysis

In both our static and dynamic watermarking scheme, the complexity of watermark embedding mainly depends on Algorithm 2 (see section 3.3.3). At each of its iterations, one watermark bit is embedded into one subset by multiplying the subset center element (an encrypted attribute value) by the encrypted value of e while making varying the HE cryptosystem random value until the v^{th} bit of the cryptographic hash subset value matches the bit value to be embedded. Let

Table 3.4: Computation time for watermark protection and integrity verification of a test database of 4000×60 elements (4000 tuples of 60 attributes) as well as for encrypting with Damgård-Jurik or ElGamal. SWHED, WUHED, DJCT and ECT represent the static watermarking of an homomorphically encrypted database, watermarking of an updatable homomorphically encrypted database, Damgård-Jurik computation time and ElGamal computation time.

Watermarking method	Computation stage	DJCT	ECT
SWHED	Database encryption	33 min 23 s	4 min 58 s
	Database watermark embedding	9 min 44 s	2 min 47 s
	Database integrity verification	16 s	31 s
WUHED	Insertion of a Watermark in one added tuple	0.31 s	0.06 s
	Watermark update if one tuple is suppressed	0.37 s	0.06 s
	Integrity verification if one tuple is added	0.0033	0.006 s
	Integrity verification for one suppressed tuple	0.0032 s	0.007 s

us recall that e is equal to 1 or 0 depending on the type of HE cryptosystem, i.e., additive or multiplicative, respectively. As one encryption operation of the value of e is of higher complexity than one multiplication, the watermarking computation complexity for one subset is bounded by $\mathcal{O}(L)$ encryptions where L is the number of iterations. Since there is one in two chances that the bit of the cryptographic hash value of a subset equals the watermark bit at each iteration, there are $L = 2$ iterations in average. As a consequence, the subset watermarking complexity is bounded by $\mathcal{O}(2)$ encryptions. Beyond, if we have a dynamic or a static database constituted of n subsets, the computation complexity is thus bounded by $\mathcal{O}(2n)$ encryptions. The computations complexity of the integrity verification procedure depends on the computation of the subset cryptographic hash values of subsets. Compared to homomorphic encryption operations, the complexity of these computations is insignificant. We provide in table 3.4, the computation time for protecting our test database and for verifying its integrity when using Damgård-Jurik and ElGamal cryptosystems. It can be seen that most computation costs are related to HE operations, especially when protecting the complete database at once. In the dynamic case, the watermarking of one tuple is greater than in static case. This is due to the fact that the system has to access and update the journal J_t. In both static and dynamic cases, the integrity verification process is quite fast. Indeed, its complexity mainly depends on the cryptographic hash computations. It can also be seen that in terms of complexity the advantage is given to the ElGamal cryptosystem. Such results confirm that the complexity of our watermarking scheme depends on the used HE cryptosystem.

In terms of watermarking capacity, our method embeds one bit of watermark per encrypted attribute subset. Based on the database partitioning algorithm given in section 3.3.3 (see Fig. 3.4) and for a given table of $N \times M$ elements (N tuples of M attributes), such a capacity K in bits can be approximated by:

$$K \approx \left\lfloor s + t + \frac{s + t + \frac{M}{4} + \frac{N}{4}}{9} \right\rfloor \text{ bits} \tag{3.21}$$

where $\lfloor . \rfloor$ is the floor function, and s and t are such that

$$s = \left\lfloor \frac{M-1}{4} + 1 \right\rfloor \times \left\lfloor \frac{N-1}{4} + 1 \right\rfloor \tag{3.22}$$

$$t = \left\lfloor \frac{M-3}{4} + 1 \right\rfloor \times \left\lfloor \frac{N-3}{4} + 1 \right\rfloor \tag{3.23}$$

Notice that it is possible to increase the capacity by using more bits of the subset hash value in order to encode several bits of the watermark. If l bits are used, the non-detection probability of a subset modification will obviously decrease (it will be $(\frac{1}{2})^l$). However, this will greatly increase the computation complexity, Algorithm 2 will have to conduct in average 2^l iterations to make such embedding possible.

3.4.4 Security analysis

The database watermarking method we propose in this chapter allows CSP to conduct the integrity verification of homomorphically encrypted databases accessed and updated remotely by their owners. Its security depends on various primitives and on the knowledge of the scheme parameters.

Encryption operations are performed using an additive or multiplicative HE cryptosystems, the security analysis of which have been well investigated in [103]. In our watermarking scheme, we do not intrinsically modify these cryptosystems as only exploit their homomorphic and semantic security properties. More clearly, there is no access to HE cryptosystems' private parameters such as the private keys and user's clear data. Therefore, the level of confidentiality they offer is still ensured. In addition, even if the attacker has access to secret watermarking parameters such as: the secret watermarking key or the database partition, there is no other information he or she can get from these parameters about private parameters of HE cryptosystem. Also, database watermarking does not compromise the database decryption as watermark embedding does not modify clear data thanks to homomorphic encryption.

Regarding database integrity, different attacks can be conducted by an attacker in order to compromise the integrity of a protected database. In the case of a static database, both database protection and integrity verification procedures depend on the secret watermarking key K_w. Without knowing K_w, an attacker can not conduct database reorganization, its partitioning into subsets as well as the cryptographic hash bits into which the watermark is embedded. Our dynamic watermarking scheme being derive from the static one, its security is the same. As stated in section 3.3.4, the journal table J_t is encrypted using AES and decrypted when necessary by CSP for a database update. As for HE cryptosystems, the security of the AES cryptosystem has been intensively investigated [276]. Notice that AES is nowadays widely used in many applications. As a consequence, an attacker must have access to the AES encryption key of the journal table so as to break the confidentiality of J_t.

3.5 Conclusion

In this chapter, we have addressed the control of the integrity by the cloud service providers, for encrypted outsourced databases. This is an important issue as illegal modifications of these

data can come from several sources such as transmission errors, unauthorized modifications from sub-contracted service providers or attackers.

We have proposed a database watermarking scheme, the purpose of which is to allow the cloud services providers to conduct the verification of integrity for homomorphically encrypted databases. It takes advantage of the semantic security property of homomorphic encryption cryptosystems in order to embed a watermark, a binary message into homomorphically encrypted databases. By making use of this property, the proposed method embeds a binary message or watermark in encrypted databases without altering user's clear data. It can be deployed with partially homomorphic encryption cryptosystems (additive or multiplicative) as well as with fully homomorphic encryption cryptosystems. Beyond, our watermarking method is dynamic, i.e., it is possible to protect encrypted databases while allowing data owners to conduct update operations such as tuple suppressions, tuple additions or encrypted attribute value modifications.

By using two cryptosystems (the Damgård-Jurik cryptosystem which is HE additive and the ElGamal cryptosystem which is multiplicative), we have experimentally shown that our solution provides high detection and localization performance capabilities. In addition, obtained results show that the proposed method has a better localization performance for illegal modifications than database watermarking schemes proposed for clear data. However, its performance depends on the computation complexity of homomorphic encryption cryptosystems. We have also given a detailed theoretical performance analysis of our method in terms of watermarking capacity, and we have proposed how the watermarking capacity can be increased by using more bits of the subset hash value in order to encode several bits of the watermark.

Finally, We have analysed the performances of our method by conducting a comparison of our solution and the state if the art, especially methods that have been proposed for integrity control of outsourced databases.

CHAPTER 4

Privacy-preserving GWAS for rare mutations

In chapters 2 and 3, we have seen that homomorphic encryption (HE) is one of strong mechanisms that are used in protecting outsourced genetic data. This is to the fact that, it allows processing on these data without decryption them. However, HE-based solutions still have an important overhead in terms of computation and communication complexities. The objective of this chapter is to overcome this issue by proposing a privacy-preserving GWAS solution that allows the secure computation of association tests and achieves the same performances and accuracy as its nonsecure version.

We have proposed a scenario where a Genome Research Unit (GRU) that has collected genetic data from cases and wants to compare them against genetic data from controls collected by a Genomic Research Center (GRC). This requires a data sharing between the GRU and the GRC and this operation is usually performed in open environments. Data are being exchanged through internet and often processed by a third-party such as cloud service providers (CSP). As we have seen in previous chapters, this induces several security problems, especially in terms of privacy and data confidentiality. In the method we propose in this chapter, GRC is positioned as as a proxy between GRU and the CSP. By doing so, it is possible to use classical cryptographic mechanisms so as to securely conduct association tests with no computation complexity increase, contrarily to actual state of the art solutions. We recall that most of these solutions are of very high complexity being based on homomorphic encryption, for instance. In particular, we show how sensitive data confidentiality can be ensured with secret key based cryptographic hashing with no need to modify statistical algorithms. In our solution, the CSP simply conducts statistical analyses on partially hashed data. Secondly, we introduce a novel privacy constraint: GRU's identity should remain unknown to the server as this knowledge can give it clues about GRU's data (e.g., diseases and genes of interest). We exhibit how Pretty Good Privacy (PGP) can be used to solve this problem, and we illustrate our protocol in the case of one rare variant association test, the Weighted-Sum Statistic (WSS) algorithm, carried out on real genetic data.

In the first time, we will explain how the proposed solution works for privacy-preserving WSS algorithm. In addition, we have analysed communication and computation complexity of solution

before conducting its security analysis.

4.1 Overview on related works and our contributions

Securing shared or externalized genetic association studies does not simply mean securing the storage and transmission of genetic data [212, 278]. Indeed, parties involved in such studies may not want that the other parties access their data, the objective and the conclusions of the study, these ones being highly valuable assets. At the same time, the trust one can have in a cloud service provider is quite relative. Thus, it is the data analysis algorithm itself and the way it is shared between parties that have to be secured. As presented in Chapter 1, Section 1.2.2.4, different methods have been proposed in order to perform privacy-preserving association studies, especially for common variants (these studies are usually referred to as Genome-Wide Association Studies, GWAS). We refer the reader to Section 1.2.2.4 and table 1.3 for more details about privacy-preserving GWAS methods. We recall that these methods are based on different cryptographic techniques such as Differential Privacy (DP), Homomorphic Encryption (HE), Secure Multiparty Computation (SMC) and Secure Hardware Computation (SHC).

Most HE cryptosystems that have been used are fully homomorphic (they allow the computation of both addition and multiplication), like BGV, YASHE and FV. Due to their complexity, some other works have been proposed to exploit the Paillier cryptosystem which is only additive. Other encryption algorithms that have been used are AES and Lightweight computational footprints (cryptosystems with low computation complexity). It is also important to notice that all these proposals do not consider mutualizing genotypes. At the least, parties share frequency tables, after having computed them locally on their respective data, that is to say without sharing these data into a unique server for instance. Moreover, all the methods developed so far considered single marker tests where each marker (SNP) is tested individually. These tests are not useful with rare variants as they will lack power. Only in [251] is the case of rare variants considered but the solution proposed is to still test for association at the single locus but use exact logistic regression to deal with parse data. None of the methods proposed solution to perform rare variant burden test at the gene level.

In this chapter, we present a new secure GWAS protocol adapted to various GWAS statistical analysis, especially iterative ones based on large sets of genotypes provided and shared by different parties in open and nonsecure environments. We were particularly interested by the analysis of sequence data and testing association with rare variants since sequencing data are more informative than genotyping data used to test for association with common variants and considered in all the previous studies. Rare variants that can even be private to a single individual more easily allow individual identification than common variants. To test for association with rare variants, they need to be considered in group within a gene and a score is computed to measure the rare variant burden in each individual and scores are then compared between cases and controls. The Weighted Sum Statistics (WSS) is an example of method commonly used to test for association between rare variants and disease. Like in common outsourced GWAS studies, this protocol considers three distinct entities: a Genomic Research Unit (GRU) with genomes of individuals presenting

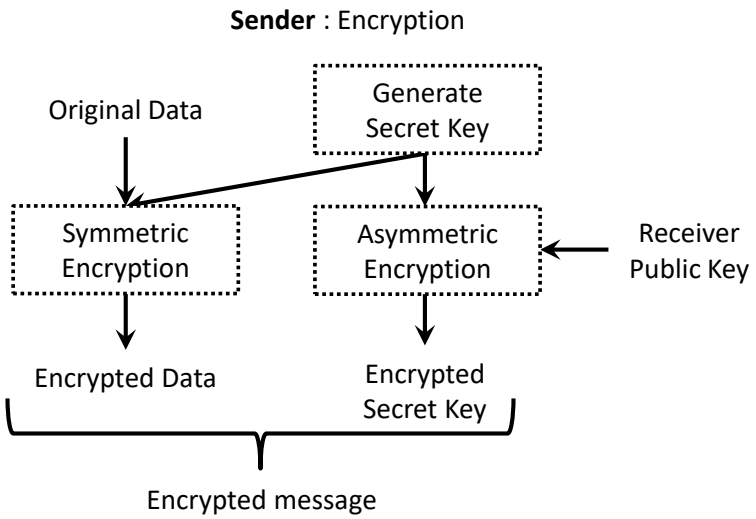

Figure 4.1: PGP encryption on the Sender side.

a phenotype (case) who wants to conduct association studies in collaboration with a Genomic Research Center (GRC) who possesses genomes of healthy people (used as control), using the large computation of a cloud service provider (Server).

In our framework, in addition to the common security constraints (all entities are considered as honest but curious (HBC); none of the parties want to disclose their confidential data), we introduce a new constraint: GRU does not want to be identified by the Server. This constraint takes into account the fact that most genomic research units are known for the diseases they are studying. Under the HBC model, this information can for example give clues to an attacker about the name of a gene and its expression for the individuals considered in a study.

The protocol we propose responds to these constraints and more. One originality stands on the fact that GRC serves as an intermediary, similarly to a proxy, in communications between all entities. By doing so, and as we will see, it becomes possible to come back to classical cryptographic tools in order to secure the WSS algorithm, or any algorithm working in a similar way. In particular, our solution takes advantage of the combination of Pretty Good Privacy (PGP) encryption with secure cryptographic hash Functions. Our main idea is that GRC and GRU ensure data confidentiality with the help of secure hash functions salted with a secret key. By using the same hashed data values, GRC and GRU allow the Server to conduct WSS counting operations on their data without accessing to their clear text values. More clearly, Server will run WSS on partially hashed data. On its side, PGP is used to secure communications while considering GRC as proxy. As we will see, GRC will never access GRU data while Server will never know GRU's identity nor his confidential data. Compared to actual solutions, our protocol preserves data and WSS result confidentiality with no WSS computation complexity increase. It can be extended to any statistical analysis equivalent to WSS, being iterative or not.

To go further, we extend our proposal under the malicious security model. It is important to notice that all papers listed in table 1.3 as well as the vast majority of privacy-preserving GWAS solutions, only consider the semi-honest security model where it is assumed that parties will not

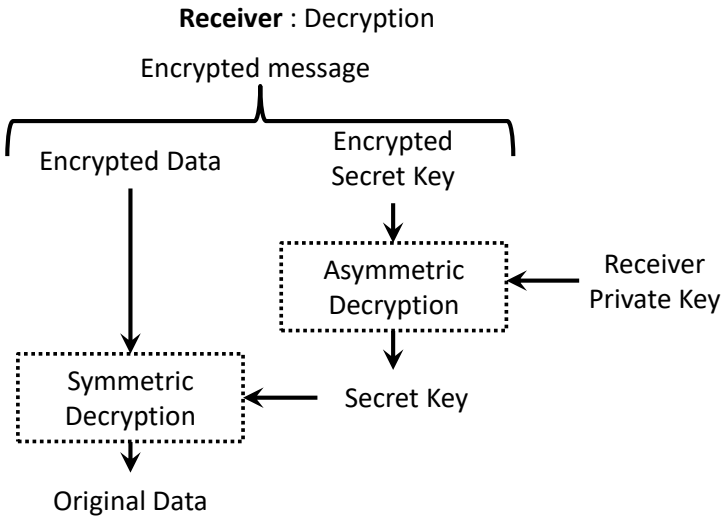

Figure 4.2: PGP decryption on the Receiver side.

alter data. This model is less constraint-full than the malicious model, and leads to computation and communication complexities of lower orders of magnitude. We suggest considering the case where Server is a malicious adversary, that is to say, it can deviate from the protocol and fails the correctness of the output or the input. To overcome this issue, we propose a practical counter-measure based on the zero-knowledge protocol, capable for instance to detect if a malicious server modifies the result of a GWAS study.

4.2 Preliminaries

4.2.1 Pretty Good Privacy Encryption

Pretty Good Privacy (PGP) is a well-known secure protocol adapted to the exchange of a large volume of data between two parties. It relies on the combination of a public key encryption (PKE) with a symmetric encryption cryptosystem. As given in Fig. 4.1, to send a message with PGP, the emitter first symmetrically encrypts it with a secret key. The same key will be used during the decryption process (see Fig. 4.2). Then, it asymmetrically encrypts this secret key by the recipient public key and sends both pieces of information (i.e., the symmetrically encrypted message and the asymmetrically encrypted secret key). On its side, the recipient first accesses the secret key by asymmetrically decrypting it using his private key. It just has to use this key to finally get access to the message. In this work, PGP is implemented with RSA [279] and AES [123] algorithms, two well-known PKE and symmetric encryption cryptosystems, respectively. RSA is parameterized by a pair of keys (K_p, K_s) where K_p is the public key and K_s the private key while the secret key of AES is noted by K_{AES}. For a given message m and a recipient A, the PGP encryption is such as

$$(m^e, K^e) = PGP(m, K_p^A, K_{AES}) \tag{4.1}$$

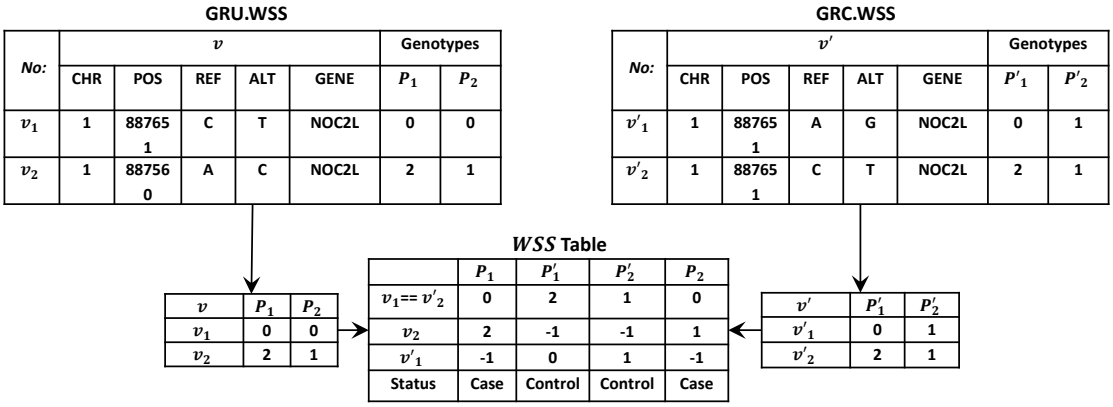

Figure 4.3: Aggregation of cases and controls tables, i.e., of $GRU.WSS$ and $GRC.WSS$ respectively, in order to produce the WSS table that servers will use as input of the WSS algorithm.

where m^e is the AES encryption version of m and K^e is the RSA encryption of K_{AES}. m is retrieved from m^e as follows:

$$m = PGP^{-1}(m^e, K^e, K_s^A) \tag{4.2}$$

4.2.2 Weighted-Sum Statistic algorithm (WSS)

WSS is one commonly used rare variant association test that was designed to identify the association between a phenotype and rare variants located in a region of the genome (e.g., gene) using sequence data on cases and controls [280]. WSS tests whether there exists an enrichment in rare variant in a gene of interest in cases compared to controls. The input data are two WSS tables. One contains case data, extracted from the database of the Genomic Research Unit (case table: $GRU.WSS$), and the second table contains control data provided by the Genomic Research Center ($GRC.WSS$). As shown in Fig. 4.3, both tables hold the information about genetic variants for one or more individuals. One line corresponds to one variant uniquely indexed or identified by: the chromosome (CHR) where it is located; its position in this chromosome (POS); the reference allele (REF); the alternate alleles (ALT); and, the name of the gene ($GENE$). Following these five columns is the list of genotypes for the sample of individuals (see P_i and P'_j in Fig. 4.3). The genotype of a patient at a given position is given by a positive integer indicating the number of alternate alleles the patient has. "0" indicates that both chromosomes of this patient contain the reference allele at this position, "1" indicates that the individual is heterozygous with one REF and one ALT and "2" indicates that the individual is homozygous with 2 ALT alleles. If data is missing then the value is "-1".

The WSS algorithm requires first to select genetic positions within the gene where there are variants of interest (based on their predicted effect on the gene protein product and on their frequencies) and then to construct a genetic score for each individual based on their genotypes at these different genetic positions and to contrast these genetic scores between cases and controls. To

better explain how the WSS algorithm works and its complexity, let us consider one gene that contains v genetic positions where there are variants of interest. The first step consists in merging $GRU.WSS$ and $GRC.WSS$ tables in a single WSS table. To do so, and as illustrated in Fig. 4.3, individual information on the same variants are grouped together. Genetic scores are then computed as a linear combination of the number of rare alleles carried by the individual at each of the v variants weighted by the minor allele frequency at this position in the control group. The idea is to give more weight to the least frequent variants since these variants are expected to be more often deleterious and thus more likely involved in disease. All individuals affected and unaffected are ranked according to this genetic score and the sum of ranks S_{obs} for affected individuals is calculated. To test the null hypothesis H_0 that the gene is not associated to the disease, a permutation procedure is then used where the case/control status are permuted between individuals N times and the sum of ranks S_{rep} is recomputed each time to obtain the distribution of S under H_0. A *p-value* which is the probability to reject H_0 given H_0 is true is estimated by determining how many time the S_{rep} value obtained on the permuted data exceeds S_{obs}. The null hypothesis is rejected if this *p-value* is less than a fixed threshold value α. Since many different tests are performed, it is necessary to account for multiple testing and fix a very small α value, typically in the range $[10^{-5}, 10^{-8}]$. The WSS algorithm works in four iterative steps. We herein describe them in details in order to give an idea about WSS complexity.

1. For each variant $i \in \{1, 2, \cdots, v\}$, we calculate a weight w_i that depends on the allele frequencies
$$w_i = \sqrt{n_i q_i (1 - q_i)} \tag{4.3}$$
where: n_i is the number of individuals genotyped for the i^{th} variant (cases and controls), $q_i = \frac{m_i+1}{2d_i+2}$ where d_i is the number of control individuals genotyped for the i^{th} variant, and m_i is the number of minor alleles observed at the i^{th} variant in the control individuals.

2. A genetic score is computed for each individual j:
$$s_j = \sum_{j=1}^{v} \frac{g_{ij}}{w_i} \tag{4.4}$$
where g_{ij} is the genotype of individual j for the variant i (it takes values 0, 1 or 2 depending on the number of minor alleles).

3. Individuals are ranked accordingly to their genetic scores (s_j) and the rank sum x for affected individuals (cases) is calculated
$$x = \sum_{j \in Cases} rank(s_j) \tag{4.5}$$

4. A standard permutation test [281] is used to compute an empirical *p-value*. The statuses (case/control) are permuted for all individuals and steps 1 to 3 are repeated k times to obtain k rank sums $x_1, x_2, \cdots, x_k$. These values are compared to the observed rank sum x and the number of permutations k_0 where it exceeds x are determined to obtain the *p-value*:
$$p - value = \frac{k_0 + 1}{k + 1} \tag{4.6}$$

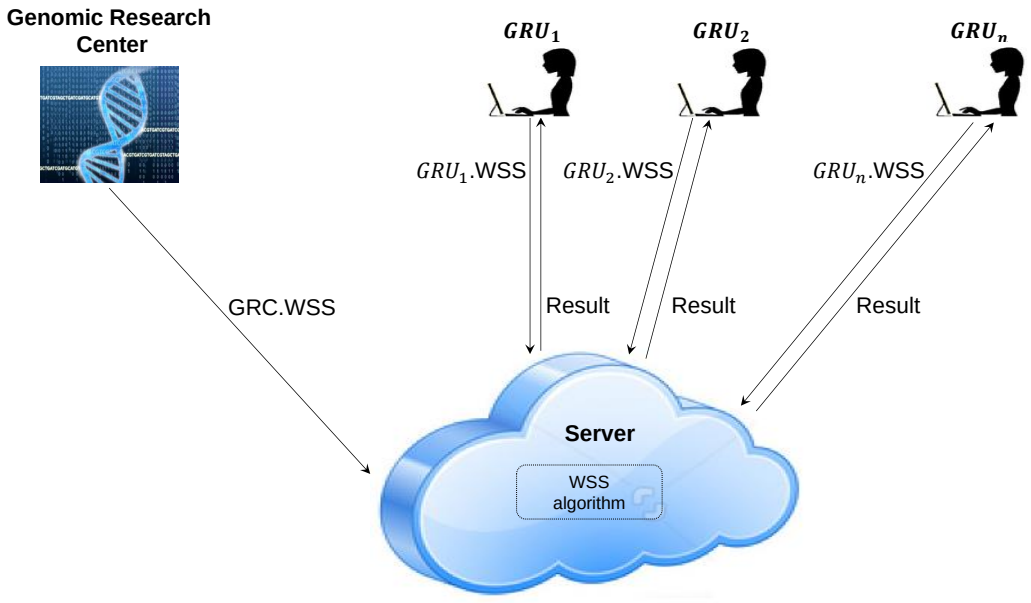

Figure 4.4: General framework of outsourced GWAS-WSS

where k_0 is the number of permutations that give a rank sum x_l at least as extreme as x, and k is total number of permutations (this is a number that will determine the maximum level of significance that can be reached).

4.3 Proposed privacy-preserving WSS algorithm

4.3.1 General GWAS framework and threat model

The scenario considered in order to conduct an outsourced GWAS study is described in Fig. 4.4 where both GRUs and GRC send their data to a server. Once Server has performed the computation and obtained the *p-value* results, it sends them to the GRUs. In such a framework, and as seen in section 4.1, different threats have to be considered. Beyond common security needs such as data confidentiality, integrity and availability [282], data privacy is of major concern.

The $GRU.WSS$, $GRC.WSS$ and WSS tables contain pieces of information that can be used to identify individuals [80]. Indeed, they provide the genotypes of several individuals for a set of variants, identified by their position (*POS*) on a specific chromosome (*CHR*) (see Section 4.2.1 and Fig. 4.3). Moreover, information is provided on the gene that contains the variants. As a consequence, *CHR*, *POS* as well as *GENE* are very sensitive pieces of information from a privacy point of view. They constitute a potential leak of information with important consequences for an individual and his/her relatives and penalties for institutions [76]. Nevertheless, it is important to notice that knowing genotypes with no information about the gene, the chromosome or the variants they belong to, it is not possible to infer information about individuals. The result of a WSS test along with the knowledge of the gene GRU is interested in, also leak important information [282].

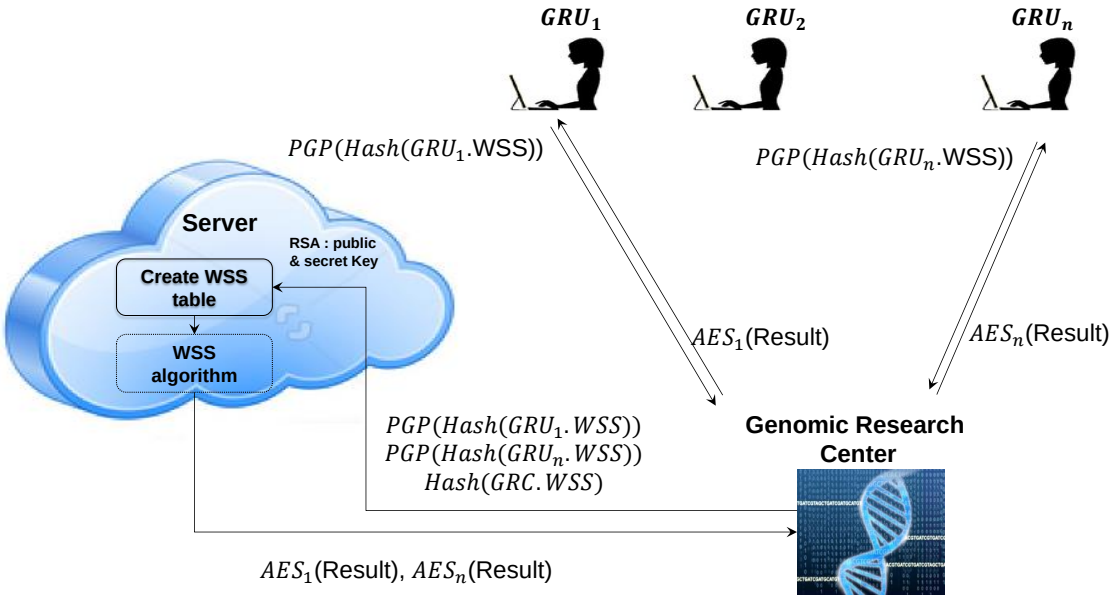

Figure 4.5: Our secure GWAS-WSS framework.

Unfortunately, in the classic framework depicted in Fig. 4.4, Server knows the identity of GRU, by definition. As a consequence, it has clues about the disease the GRU study focuses on, and so knows the *p-values* that measure the degree of association between all genes and this disease. This can both lead to patient re-identification (if data were taken from a database related to this disease) and to an intellectual property breach about the association of the gene X with the disease Y. As we will see in the next Section 4.3.2, we propose a novel architecture to overcome this problem. It is important to notice that, in a WSS study, even if the server has some knowledge about the study results (i.e., *p-values*) and about unlocalized WSS genotypes, it can not infer significant information without knowing details about the variant and the gene name.

Beyond the sensitivity of WSS data, in our framework, we further assume that first GRC and Server are honest but curious and that they do not collude. More clearly, both of them may try to infer information about confidential data but they will not exchange information they have to keep secret.

To sum up the above discussion, to outsource a WSS computation in such an open environment, the following security constraints have to be considered:

1. Confidential data of GRU (resp. GRC) that can help to identify individuals should not be disclosed to GRC (resp. GRU) and Server.
2. Server should have no idea about the gene GRU is working on, nor on the GRU identity.
3. GRC should not know the results of the WSS (*p-values* of a set of genes) due to the fact it knows the GRU identity and thus the disease the GRU might be interested in.

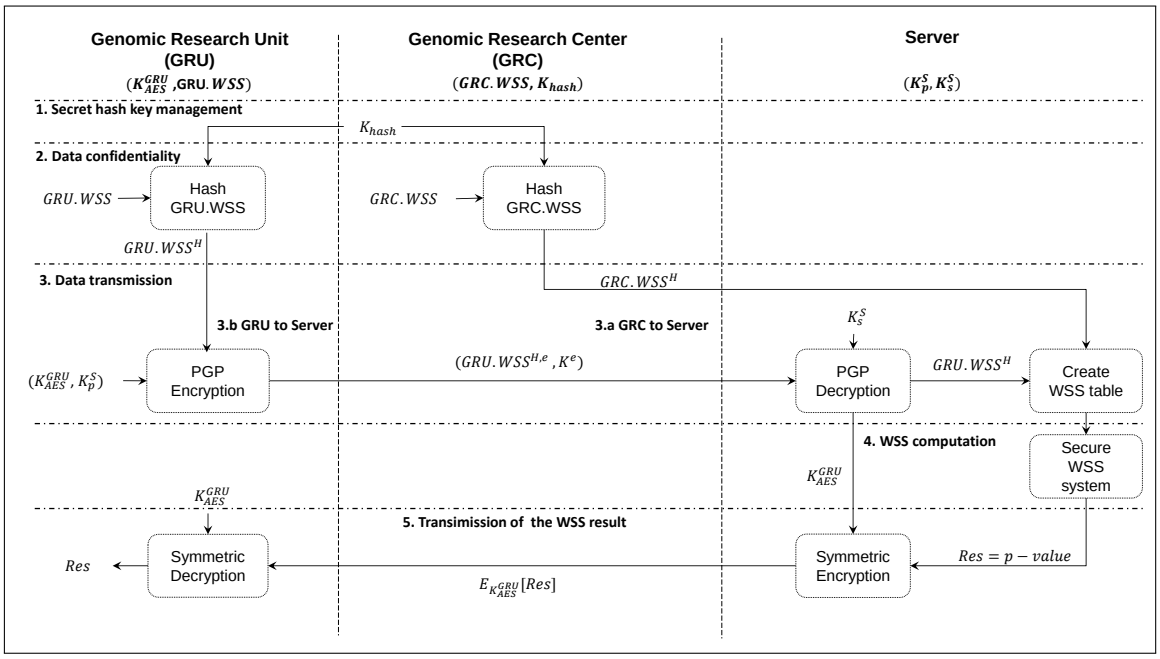

Figure 4.6: Different steps of our secured WSS protocol in the case of one gene.

In the next section, we propose a new framework that satisfies these constraints while securing the WSS algorithm. As we will discuss in Section 4.3.2, the following framework can be extended to any other statistical analysis processes close to WSS, these ones being also concerned by the above constraints and using the same type of inputs.

4.3.2 Proposed secured WSS algorithm

The implementation of our framework will guarantee that all point-to-point communications in-between parties are secured with common security mechanisms (e.g., user authentication, access control policy, firewalls, SSH protocol and so on). Furthermore, in order to escape a man-in-the-middle attack, we assume that the key setup works correctly and that all entities obtain the correct encryption key which can be enforced with appropriate use of Certificate Authorities and/or a Public Key Infrastructure.

As stated above, the framework we propose takes into account a new constraint: Server should not be able to identify GRU, as this knowledge can give clues about the possible disease of the genotyped individuals. To achieve this goal, and as depicted in Fig. 4.5, we suggest that GRC plays the role of a "proxy" between GRU and Server. More clearly, all communications from GRU to Server and from Server to GRU go through GRC. Server thus has no idea about the GRU. In this situation, we take advantage of PGP in order to ensure the confidentiality of GRC's data. To do so and as explained in Section 4.2, GRU firstly AES encrypts his data based on an AES secret key it generates and, then sends these data along with the AES secret key asymmetrically encrypted with the Server RSA public key. Only Server will be able to access the AES key and consequently decrypt the data. Server can conduct this task without knowing the identity of GRU. As GRC has no knowledge of the AES key nor Server's Private Key, it is unable to decrypt GRU data

while transmitting them to Server. The second important point to manage is to make it possible for Server to compute the WSS algorithm without being able to identify the variants of GRU and GRC. To ensure the confidentiality of GRC and GRU variants, the confidential attributes CHR, POS, REF, ALT and $GENE$ values in $GRC.WSS$ and $GRU.WSS$ tables are substituted by secret hashed values, computed with a cryptographic hash function based on a secret hash key K_{hash} GRU and GRC previously agreed on through the use of a secure channel of communication. This step allows the creation of secured WSS tables without compromising GRU and GRC data security. Notice that genotype data in $GRC.WSS$ and $GRU.WSS$ are not modified. As seen in Section 4.1, this does not endanger individual privacy as Server does not know the real variant's genetic location and alleles.

In the following, we give more details about this protocol when only one GRU collaborates with GRC to conduct a WSS study, but it can easily be extended to support several GRUs. If GRC provides several data sets, it is of course essential that GRU selects the one most suited to its analysis and especially that cases and controls are matched on ethnicity to limit population stratification bias. Thus, let us consider that one GRU wants to perform a WSS study with GRC for a specific gene so as to see if this latter is associated to a phenotype. Prior to any security consideration, we assume that GRU and GRC have followed common guidelines to produce their data, and that similar quality controls have been applied on the data. Let us also assume that Server has a RSA pair of key (K_p^S, K_s^S). The main steps of our protocol which are depicted in Fig. 4.6 works as follows:

1. **Secret hash key management**: GRC and GRU first have to agree on a unique secret hash key K_{hash} using a secure key exchange protocol like the SFTP protocol [283].

2. **Data confidentiality**: GRU and GRC substitute the confidential attribute values in their WSS tables (i.e., $GRU.WSS$ and $GRC.WSS$, respectively), by secure hash values using the secret hash key K_{hash}. More clearly, taking $GRU.WSS$ as example, GRU computes:

$$\begin{aligned} hash(CHR_i||POS_i||REF_i||ALT_i||K_{hash}) \\ ||hash(GENE||K_{hash}) = \\ v_i^H||hash(GENE||K_{hash}) = h_i \end{aligned} \tag{4.7}$$

   where the confidential attributes CHR_i, POS_i, REF_i, ALT_i and $GENE$ constitute what we name in the following the variant v_i. It can be noticed that in (4.7), we concatenate the secret hashes of the variant confidential attributes with the one of the gene (i.e., "$GENE$"). This is due to the fact WSS computes one *p-value* per gene and not per variant (see Section 4.2.1). Server has thus to be able to discriminate the variants located on the same gene. In the case GRU just wants to study one gene, then h_i can be refined in

$$\begin{aligned} hash(CHR_i||POS_i||REF_i||ALT_i||K_{hash}) &= v_i^H \\ &= h_i \end{aligned}$$

   The resulting hash tables are referred to as $GRU.WSS^H$ and $GRC.WSS^H$. An example of this process is given in Fig. 4.7. Finally, GRC sends its hashed table $GRC.WSS$ to Server

3. **Data transmission**-

   a) **GRC to Server**: GRC sends $GRC.WSS^H$ to Server. Due to the fact that the communication between GRC and Server is point-to-point, and by definition secured (see above), there is no need to use PGP.

   b) **GRU to Server**: GRU securely sends its secured table $GRU.WSS^H$ to Server using PGP. To do so, it generates the PGP symmetric key K_{AES}^{GRU}. Then it entirely PGP encrypts them, that is to say (see Section 4.2.1).

$$(GRU.WSS^{H,e}, K^e) = PGP(GRU.WSS^H, K_p^S, K_{AES}^{GRU})$$

   where K_p^S is the Server RSA public key. Next, GRU sends $(GRU.WSS^{H,e}, K^e)$ to Server through GRC so as to preserve its privacy.

4. **WSS computation**- When Server receives $(GRU.WSS^{H,e}, K^e)$, it first decrypts the AES key K_{AES}^{GRU} from K^e using its RSA secret key K_s^S. Then, it AES deciphers $GRU.WSS^{H,e}$ to get access to $GRU.WSS^H$. Server also gets the data from GRC. As shown in Fig. 4.7, Server creates the WSS hashed table (WSS^H) from $GRU.WSS^H$ and $GRC.WSS^H$ (see Section 4.2.2). Due to the fact that genotype data are not encrypted, Server can directly apply WSS on WSS^H. Indeed, the WSS algorithm is not modified. It will simply work with hashed values instead of real values, by comparing hashed values of genes to group variants and hashed values of variants to group genotypes.

5. **Transmission of WSS result**- Once Server obtains the WSS results, that is to say the Gene's WSS *p-value* (see Section 4.2.2), it AES encrypts it using the GRU AES key (K_{AES}^{GRU}) and sends it to GRU through GRC. Finally, GRU just has to decrypt this piece of data using the same AES Key to get access to the results of its WSS study. By doing so, its identity is never revealed to Server.

Notice that, in the case GRU wants to analyze several genes, it will receive as many *p-values* from Server. In order to generalize this approach to more than one GRU willing to pool their data for more powerful statistical studies, all GRUs will follow the same steps as above:

i) They hash their sensitive data (variants) by using GRC secret key K_{hash}. Since all of them have access to the public key of Server, they encrypt their WSS table with PGP parameterized with their respective AES key and the public key of Server.

ii) The encrypted data are sent to Server through GRC.

iii) As shown in Fig. 4.5, Server decrypts the PGP encrypted $WSS.GRU$ tables and merges them with the $WSS.GRC$ table.

iv) Finally, Server runs the WSS algorithm, encrypts the results using the AES Key of each GRUs before sending it through GRC. The results received by each GRU contains only the *p-values* associated to the genes that particular GRU provided.

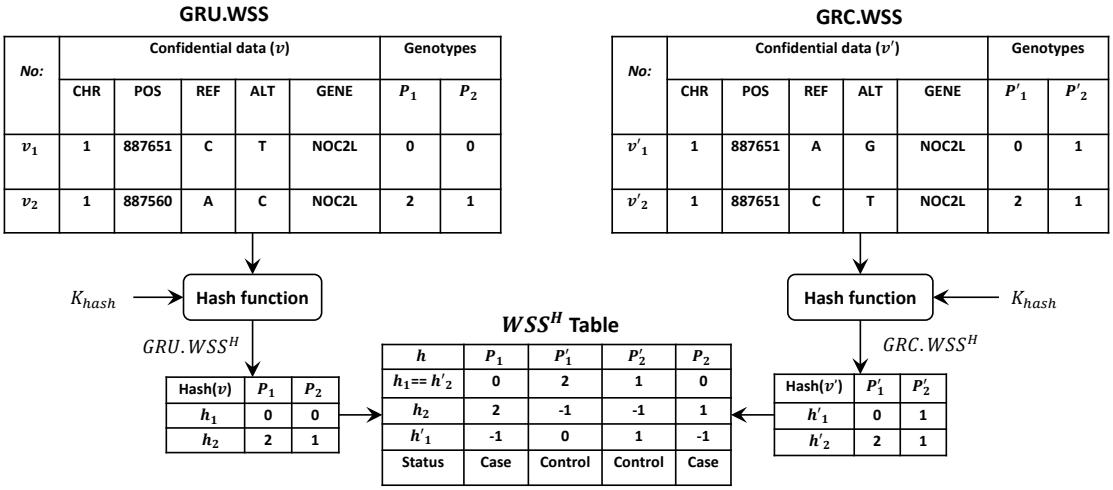

Figure 4.7: Creation of the secure WSS table from the hashed versions of $GRU.WSS$ and $GRC.WSS$. h_i and h'_i represent the hash values of v_i and v'_i, respectively.

4.4 Experimental results and discussion

The proposed secure GWAS framework was tested on real genetic data: exome data were compared between (1) 100 healthy individuals from the FrEx project [284] that served as GRC control data and (2) 59 individuals affected by a rare disease sequenced independently to the FrEx data (GRU cases). Cases and controls were sequenced on the same platform (CNRGH, Evry, France) at different times and using the Agilent SureSelect Human all exon V5 capture kit for the cases and the Agilent SureSelect Human all exon V5+UTR capture kit for the controls. Sequence data were processed using the exome analysis platform developed at CNG, which follows GATK best practices. Coverage/depth statistics were as follow: for each sample a minimum of 20X coverage for 80% of the targets was obtained and the average sequencing depth was of at least 70 to 80X. Polymorphism detection for each sample was performed using read mapping procedure onto the reference genome (hg19) followed by "SNP calling" algorithm implemented in GATK/samtools software. Stringent quality controls were performed after variant and genotype calling. Only genotypes with min GQ$\geq$ 20 and min DP$\geq$ 10 were kept and the other genotypes were set to missing. Variants failing any of the following thresholds in any of the two datasets were discarded from both GRC and GRU datasets: min callrate$\geq$ 0.9, HQ variants (as define in ExAC : 80% of genotype with DP $>$ 10 & GQ$>$ 20, at least one variant genotype with DP$>$ 10 & GQ$>$ 20), min QD$\geq$ 2, min inbreeding coef≥ -0.8, ABhet in the range $[0.25; 0.75]$, min MQRanSum≥ -12.5, max FS$\leq$ 60 for SNV or $\leq$ 200 for INDEL, max SOR$\leq$ 3 for SNV or $\leq$ 10 for INDEL, min MQ$\geq$ 40 for SNV or $\geq$ 10 for INDEL, min ReadPosRankSum≥ -8 for SNV or ≥ -20 for INDEL. Note that each party is expected to perform this same QC on its own dataset and send to the other party the list of variant sites excluded (only chromosome, position, reference and alternative alleles and no individual data).

In our example, a total of 11196 genes contained at least two qualifying variants and were tested for association. Qualifying variants kept in the analysis were those with an expected effect on the

Table 4.1: Computational costs of the WSS algorithm with and without parallelism.

Number of gene	Parallel WSS algorithm (56 cores)	Standard WSS algorithm	$p-value$	k_0	number of permutations (k)
1	20.22 s	14 min	5.504e-5	5	109000

encoded protein (i.e., variants that were annotated as transcript ablation, splice acceptor or donor, stop gained or lost, start lost, frameshift, inframe insertion or deletion and missense) and variants with a Minor Allele Frequency below 0.05. To compute the genetic score, missing genotypes were replaced by the most frequent genotype in the sample at the variant position. The WSS algorithm was run on each gene with a maximum of 10^9 permutations, and the overall runtime was 10 hours and 18 minutes on a server with 56 processors at 2.40 GHz and 512 GB RAM running on Ubuntu 16.04 LTS. Since in our implementation no encrypted data are used in the actual computation, runtime is the same as in the classical implementation of the algorithm. The only difference is an overhead of a few seconds to hash, encrypt and decrypt the input tables. Furthermore, the WSS *p-values* obtained for each gene are similar to the ones obtained from doing the same test on non-distributed data.

To determine if batch effects could be a concern linked to the fact that cases and controls were not sequenced together, we produced the corresponding QQ-plot as suggested in different works [285–287] and we computed an inflation factor [288]. This inflation factor was obtained by transforming the observed *p-values* into one degree-of-freedom χ^2-statistic and computing the median of these values divided by the expected median of the corresponding one degree-of-freedom χ^2 distribution.

Visual inspection of the QQ-plot (see Fig. 4.8) suggests that the stringent QC performed was efficient at correcting for batch effects and it even leads to conservative results with an inflation factor below 1 ($\lambda = 0.75$). This was however a favorable situation as cases and controls were sequenced on the same platform with capture kits that were only slightly different.

4.4.1 Computation and communication complexity

On the GRU and GRC sides, the computation complexity corresponds to the WSS table hashing and encryption processes. Notice that *SHA256* and AES computation complexities are low and increase linearly with the size of the WSS table. To give an idea, it takes about 0.53s to both hash and to AES encrypt the WSS table of 406 genes and 733 patients. Regarding Server, this one has to: 1) decrypt the $GRU.WSS$ table, 2) merge $GRU.WSS$ and $GRC.WSS$ into the complete WSS table and 3) perform the WSS algorithm before AES encrypting the WSS results. Here, the complexity of step 2) and 3) are the same as working with data in their clear form. The complexity overhead stands on the AES decryption of WSS tables; complexity which is quite close to the AES encryption process. We give in Table 4.2 experimental computational time of our solution where it can be seen that the time and the accuracy performances of secure WSS as well as the nonsecure WSS are the same.

One can also notice in this table 4.2 that our WSS implementation was parallelized in order to increase its speed. As seen in Section 4.2.2 after computing the rank sum x at the step 2, the status

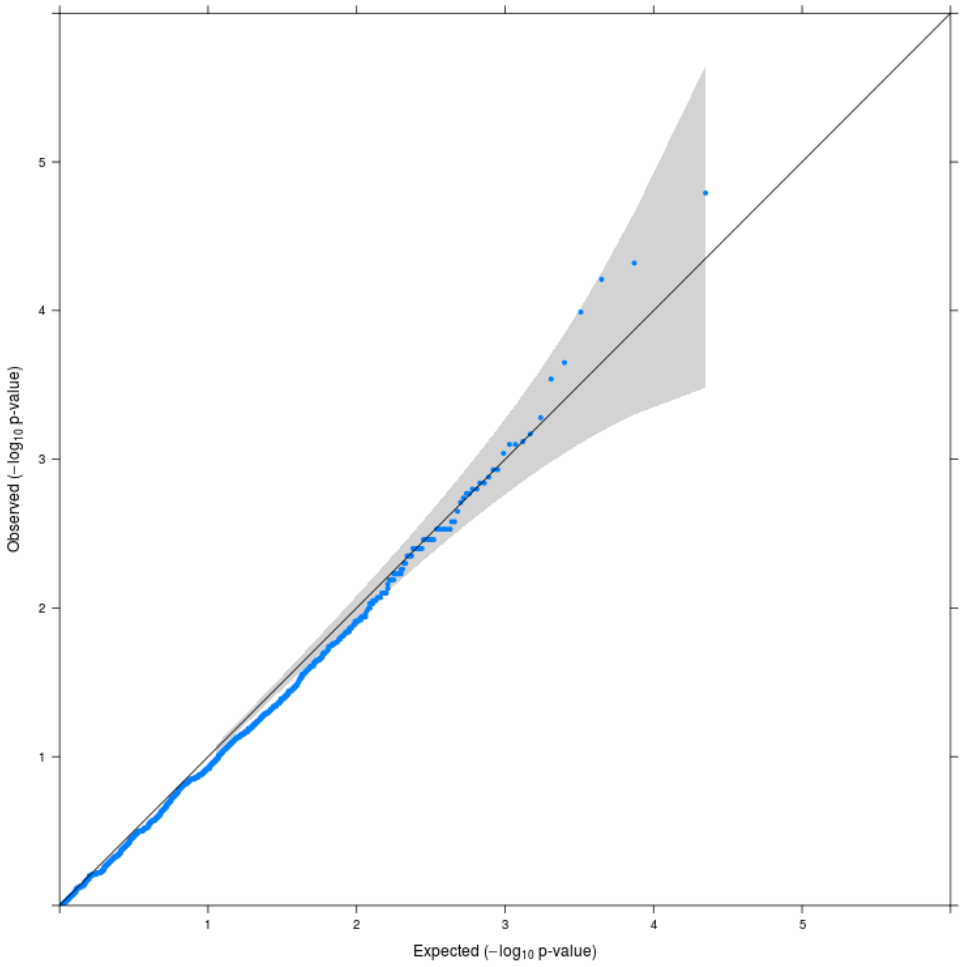

Figure 4.8: Quantile-Quantile plot of the WSS test *p-values* obtained when comparing exomes from 59 cases coming from one project against 100 controls coming from another project. Cases and controls were sequenced on the same sequencing platform but at different times and using different capture kits. The same variant calling pipeline was used and stringent QC were performed. Results are presented for each of the 11196 genes that contain at least two qualifying variant for the association test. The genomic inflation factor is $\lambda = 0.75$.

(case/control) is permuted k times so as to compute the *p-value*.

To take advantage of a server with multiple processing units (e.g., $PU_1, \cdots, PU_n$), this permutation test can be separated into k/n parts of n permutations, namely $\{x_{1,j}, \cdots, x_{n,j}\}_{j=1..k/n}$ where $x_{i,j}$ is the j^{th} rank-sum permutation computed at processing unit PU_i (see Section 4.2.2). As the processing units $PU_1, \cdots, PU_n$ can run in parallel, the *p-value* computation at step 4 (see Section 4.2.2) becomes as follows:

$$p - value = \frac{\sum_{j=1}^{k/n} \sum_{i=1}^{n} (x > x_{i,j}) + 1}{k + 1} \tag{4.8}$$

Table 4.2: Computational time of parallel Secure WSS algorithm vs nonsecure parallel version for 406 genes and 733 individuals

Number of Genes	Hashing and encryption Time	WSS algorithm Time (Clear form)	Time (Secure WSS)
406	0.53 s	4 days	4 days

where k is the number of permutations. Therefore, the use of parallel computation significantly increases the WSS algorithm speed, as experimentally shown in Table 4.1.

The communication complexity of our secure WSS algorithm for GRU or GRC is bounded by $O(n)$ bits where n is the size in bits of the WSS table. Compared to the nonsecured WSS algorithm, the communication overhead corresponds to the size of the hash key K_{hash} and to the RSA encryption of the AES key. This overhead does not depend on the size of the WSS table and is very small. Therefore, it is negligible compared to the rest of the WSS data to transmit.

4.4.2 Discussion and security analysis

The following analysis considers the semi-honest adversary model where it is assumed that parties involved in the protocol do not collude but try to infer information about sensitive data; that is to say GRU and GRC data. In our scheme, the confidentiality of WSS tables during their communication is ensured by the AES cryptosystem, the security of which has been demonstrated in [123]. GRC will never have any clues about the GRU data, these being PGP exchanged with Server. Once decrypted on the Server side, the confidentiality of the sensitive attributes of these tables (e.g., CHR, POS, $GENE$ and so on, see Section 4.3.2) stands on the secure hash function *SHA256*, the security level of which has been investigated in [275]. It is not possible for Server to retrieve the original sensitive attribute values from their hash values without the knowledge of the hash Key. This key is only known from GRC and GRU. Notice that, the fact GRC sends several times its data to Server for different studies is not a problem at the condition a new secret hash key is used. Doing so makes the computation of *SHA256* values semantically secure (i.e., the same variant has different hashes values for distinct studies). Notice that, as GRC has no knowledge about GRU AES key (K_{AES}^{GRU}), it can not access to the hashed GRU table nor to the results provided by Server.

Beyond data confidentiality, one must also consider statistical inference techniques that can be used for the re-identification of genetic data donors. These attacks have been extensively investigated [237]. They depend on the *a priori* knowledge one can have of the frequencies of genotypes for given variants or a gene. Homer *et al* [80] showed that inference techniques could be used to identify the presence/absence of an individual in a genomic dataset from aggregate statistics (e.g., allele frequencies). In [76], authors presented an attack for genetic data sharing beacons (publicly available genomic databases). This attack aims at seeing if an individual is in a beacon or not. To do so, they assume that the attacker has the genomic profile of an individual and a VCF file [48] listing all the variants for this individual. From the variants, and more specifically from the heterozygous alternate alleles of the victim, the attacker generates some queries he next addresses to

the beacons. Based on the responses, he conducts a statistical hypothesis test so as to decide if the victim is present in a particular beacon.

In our framework because GRU and GRC hash their confidential variants' values, Server is not able to conduct such an attack. In fact, Server has no idea about the variants and the genes being evaluated. This statement is valid at the condition GRU or GRC do not collude with Server. For instance, if Server and GRC collude, they have access to the AES and hash keys and can consequently breach GRU data confidentiality. Nevertheless, it is hard to believe that GRC or Server would collude, as their reputations are invaluable assets.

Although Genomic Research Units (GRUs) are known for the diseases they are working on, that is to say the genes that they more frequently focus on, Server cannot deduce any clues from GRU identity due to the fact Server only communicates with GRC; GRC which acts as a proxy.

To go further, one can notice that all papers listed in table 4.3, as well as the vast majority of genome privacy solutions, only consider the semi-honest security model. This one assumes that all entities involved follow the protocol and will not try to alter data or the result of a process. At the same time, under this model, solutions are significantly easier to instantiate with computation and communication of smaller complexities than under the malicious model. Under this latter model, there is no guaranty that the association test or patient information are not going to be altered. For instance, Server could modify the WSS algorithm or change the correct value of the *p-value*. To overcome this issue and to extend our framework under such a malicious model, we propose a zero-knowledge protocol. In this one, GRC sometimes plays the role of GRU and GRC at the same time. By doing so, GRU sends to Server both the $GRU.WSS$ and $GRC.WSS$ tables for which GRC has already the knowledge of the result (i.e., the *p-value*). If GRC finds that the *p-values* computed by Server did not match the pre-computed *p-values*, it can then deduce that Server is malicious.

It is important to notice that our framework is not limited to secure WSS association tests, it can easily be extended to any other GWAS statistic algorithms that rely on the same kind of data. CAST, SKAT [289] and SKAT-O [290] are association tests that can be implemented in our framework. Another useful method that could be implemented is Principle Component Analysis (PCA). This statistical method, run before the GWAS algorithm itself, can ensure that the merged dataset can be used to perform such an analysis. Indeed, a PCA where GRU and GRC data are separated indicates that any signal obtained through GWAS is unreliable and results from divergent quality of the data or population stratification.

The pieces of data they rely on and which are sensitive from a confidentiality/privacy point of view can also be replaced by secure hash values.

4.5 Comparison to the existing solutions

Comparing in terms of performance our framework with other proposals from the literature is a nontrivial task because each work in the genome privacy does not necessarily secure the same process. For this reason, we compare whenever is possible the secure versions to the nonsecure

versions of the same functionality. Inspired by [291], we choose different criteria aiming at capturing different aspects related to security, efficiency, and data utility. They correspond to:

4.5.1 Performance Criteria

- **Privacy Overhead.** It quantifies the overhead introduced by the security mechanisms used to secure an association test. All solutions given in table 4.3 have been analyzed in order to assess their efficiency in terms of communication, time and storage overhead in comparison with their nonsecured counterpart. We quantify these performances by means of three values: Communication - L.C.O: Low Communication Overhead vs. H.C.O: High Communication Overhead; Storage - L.S.O: Low Storage Overhead, H.S.O: High Storage Overhead; Time - L.T.O: Low Time Overhead, H.T.O: High Time Overhead.
- **Utility Loss.** This criterion evaluates the impact of privacy tools on the utility of the association test. This measurement also includes the overall flexibility of the proposed solution with the intended task. We quantify the utility loss on two levels: High or Low.
- **Security model.** It indicates which security model has been considered by the authors: semi-honest model or malicious model.

As shown in table 4.3, all methods based on differential privacy (DP) induce a utility loss compared to the same process over clear data. This is due to the fact these schemes add a noise to the data. Homomorphic encryption (HE) can help to solve this problem but at the price of significant computational and storage overheads. Most of the time, they are impractical for real life applications [21]. Secure multiparty computation (SMC) constitutes a nice alternative due to its lower computational overhead. However, garbled circuit-based need complex and optimized circuit design limiting its flexibility and usability, greatly. On its side, secret sharing involves huge communication overhead and is not suitable for client server architecture. Secure hardware-based approaches, like SGX based techniques, isolate sensitive data into a protected enclave for secure computation. However, they remain sensitive to side-channel attacks [231]. Notice that the full extent of SGX security has yet to be explored.

Compared to the previous solutions, our framework is based on PGP and *SHA256*, two cryptographic mechanisms of very low complexity, contrarily to HE. Furthermore, we do not intrinsically modify the association test algorithm. Sensitive data in terms of confidentiality are substituted by secret hash values. Thus, and as shown in Section 4.3, our framework preserves the accuracy of the association test. That is not the case of DP [233, 239, 240]. Server can also conduct the WSS algorithm without the need of additional communication as required in approaches based on SMC [20, 243–247] or to encrypt homomorphically the genotypes as proposed in [17–20, 248] which leads to high computation and storage complexity. Thus, our solution has no loss of accuracy and insignificant overheads (in memory, computation and communication) compared to the original WSS algorithm.

Table 4.3: Comparison of the most representative genomics privacy methodologies. Columns correspond performance criteria. Meaning of the acronyms: (Security) - Sh - semi-honest model - NC Noncollude model; (Overhead) L.S.O: Low Storage Overhead, H.S.O: High Storage Overhead, L.T.O: Low Time Overhead, H.T.O: High Time Overhead, L.T.O: Low Communication Overhead, H.T.O: High Comminication Overhead.

References	Security Model	Security Mechanisms	Overhead	Utility Loss
[233, 238, 239, 241, 242]	–	DP	L.S.O, L.T.O, L.C.O	High
[243]	SH, NC	Secret sharing	L.S.O, H.T.O, H.C.O	High
[244]	SH	Garbled circuit	L.S.O, H.T.O, H.C.O	Low
[245]	SH	Secret sharing, Lightweight computational footprints	L.S.O, H.T.O, H.C.O	High
[246]	SH, NC	Secret sharing	L.S.O, H.T.O, H.C.O	Low
[247]	–	Secret sharing	L.S.O, L.T.O, H.C.O	Low
[251]	SH	BGV	H.S.O, H.T.O, H.C.O	Low
[16]	SH, NC	FHE	H.S.O, H.T.O	High
[18]	SH	BGV, YASHE	H.S.O, H.T.O	High
[17]	SH	BGV	H.S.O, H.T.O	High
[19]	SH	BGV	H.S.O, H.T.O, L.C.O	Low
[20]	SH	Secret sharing, Blinding, FV	H.S.O, L.T.O	Low
[230]	Malicious	AES-GCM, SGX	L.S.O, L.T.O, L.C.O	Low
[252]	Malicious	AES-GCM, SGX	L.S.O, L.T.O, L.C.O	Low
[248]	SH	Paillier, SGX	H.S.O, L.T.O, H.C.O	Low
Our work	SH or malicious, NC	Hash, AES	L.S.O, L.T.O, L.C.O	Low

4.5.2 Statistical Power Criteria

Implementations of secured association tests proposed in the literature have considered single variant association tests that are mostly performed on genotyping data. In our work, we have considered rare variant association tests where, rather than testing each variant individually, we grouped them within a unit of analysis, here the gene. Rare variant association tests explore alternative genomic architectures for common diseases than the classical «common disease-common variant» model that was considered before. Indeed, different real examples and simulation studies have shown that rare variants might contribute more than common variants to common diseases [292]. To study the impact of these rare variants on disease susceptibility, it is necessary to sequence the genome of individuals and the sharing of sequence data is even more problematic than the sharing of genotyping data since sequence data contain information on all the genetic variants present in an individual genome including deleterious variants possibly involved in monogenic diseases that the individual could develop in the future and could transmit to offspring. It is therefore important to specifically address the problem of rare variant association tests as we have done here in a general framework that could also integrate common variant tests. This is the case in our proposed framework that could easily be extended to include other statistical tests and measures considered in previous works such as χ^2-statistic, Fisher's Exact Test, Logistic regression, MAF test, Cochran-Armitage Test for Trend, Goodness of Fit, Hardy-Weinberg Equilibrium. In the same way, we have only implemented one rare variant association test here but our framework is general enough to allow the easy implementation of other rare variant association tests including variance component tests that are widely used in rare variant association studies [293].

Contrary to the WSS test we have implemented here, some of the tests can be adjusted on covariates such as age or gender. Information on these covariates for each individual could be transmitted by the GRU to the GRC and to the GRC to the Server together with the WSS tables. Some particular covariates on which adjustment could also be required to avoid false positives due to population stratification are leading principal components (PCs) from the principal component analysis performed on genotypes data of both cases and controls. To obtain these leading PCs, a possibility will be to add ancestry informative SNPs and exchange information on individual genotypes at these SNPs to perform principal component analysis on the Server. This will however involve the sharing of genetic data. Another possibility could be to use spectral graphs in a manner similar to the approach suggested by Bodea et al. [294] or the singular value decomposition suggested by Artomov et al. [295]. This will however require some further developments that are beyond the scope of this chapter. Another concern when comparing sequence data of cases and controls that were not generated together is the possibility of systematic bias due to batch effects. The problem is even more drastic when different platforms are used to sequence cases and controls. Different studies have evaluated these biases and proposed some solutions to reduce them [285–287]. Strict quality control is key in this process and it is also important to visualize QQ-plot in order to diagnose any inflation of the statistics. We have illustrated this in the example provided and shown that with the strict QC parameters we used the QQ-plot was not inflated. In this example however, cases and controls were sequenced on the same platform and only the capture kits were slightly different. In less favorable conditions, it might be necessary to test different QC parameters to de-

termine the best combinations. This would require some extra-computations and a lighter version of the test where cases and controls statuses are not permuted should perhaps then be considered to fix the QC parameters. It might also be necessary to pre-select some different sets of parameters with different levels of QC and evaluate the level of inflation by computing a statistics similar to the genomic inflation factor [288].

4.6 Conclusion

Several genome-wide association studies are being conducted in order to identify associations between genetic variants and certain diseases. We have seen that these studies are being outsourced on the cloud as it allows more computation and storage capacity at a low cost. However, as we have seen at the beginning of this chapter, this comes with several security issues. The objective of this chapter is to allow the privacy-preserving computation of statistical algorithms used in GWAS such as WSS.

We have proposed a new privacy-preserving GWAS framework that allows performing in a secure way genome-wide association studies similar to the WSS algorithm. Our solution relies (1) on a Genomic Research Center which acts as proxy in order to preserve the privacy of Genomic Research Units, (2) on Pretty Good Privacy to secure communications and (3) on cryptographic hash functions so as to ensure the confidentiality of sensitive data in WSS input tables. The security analysis of our solution demonstrates that it is secure under the honest but curious adversarial model and robust to statistical inference attacks. We also have extended our framework under the malicious security model by means of zero-knowledge protocol. Experimental results conducted on real genetic data demonstrate that the proposed solution achieves the same performances and accuracy as the unsecured WSS algorithm. Consequently, it can be used in real world environments contrarily to other proposed solutions based on Homomorphic encryption. Furthermore, this solution can be extended to any other GWAS algorithms similar to the WSS algorithm.

CHAPTER 5

Robust database watermarking for GWAS data

In previous chapters, we have seen that the deployment of genome-wide association studies on the cloud requires a large amount of genetic data. As these data face several security threats during their storage and/or processing, we have proposed in Chapter 4, a solution that allows the privacy-preserving of genome-wide association study for rare mutations. In this method, data are extracted from VCF files based of several predefined parameters and are stored in WSS files. These later are sent to the server provider for WSS computations. However, as we have seen in chapter 1, these data may face several security threats. The objective of this chapter is to ensure the security of genetic data used in GWAS such as WSS data in terms of copyright protection or traitor tracing, i.e., identifying the person or entity who is the origin of an illegal information disclosure.

Until now, authors have mainly focused on cellular DNA for various purposes such as steganographic reasons [33], copyright protection [29] or for data storage [24]. In this work, we are the first to provide a watermarking method for genetic data (stored in VCF files) used in GWAS without inference on their results (p-values). Our method is derived from database watermarking due to closeness of VCF files with relational databases, and it is based on Quantization Index Modulation (QIM) for watermark embedding and majority vote for the detection/extraction of the watermark. More clearly, the watermark is secretly embedded within genetic data used in GWAS, without violating the identification of candidate variants or genes involved in pathology. We evaluate the theoretically performance in terms of insertion capacity, distortion and robustness against different attacks. Experimental results conducted on real genetic databases ensure the efficiency of the proposed scheme, and demonstrate that it can be used for identifying the cloud service providers or geneticists at the origin of information disclosure even if the genotype data have been modified.

5.1 Genetic data for Weighted sum statistic method

In order to conduct GWAS studies, individuals that are either affected (cases) and unaffected (controls) are genotyped so as to produce thousands or up to millions of genetic variants that are

Table 5.1: An example of weighted sum statistic (WSS) file. It stores genetic information that is used in WSS method

$chrom$	pos	id	ref	alt	NA00001	NA00002	NA00003
20	1234567	microsat1	GTC	G	1	0	1
20	1234567	microsat1	GTC	GTCT	0	2	0
20	17330	.	T	A	0	1	0
20	1110696	rs6040355	A	G	0	-1	2
20	1110696	rs6040355	A	T	2	2	2
20	1230237	.	T	.	0	0	-1
20	1234567	microsat1	GTC	G	1	0	1
20	1234567	microsat1	GTC	GTCT	0	2	0

then stored into VCF files. After that, an intermediary step is conducted so as to generate other files that are specific for each association study. We are interested in watermarking WSS files which contain extracted data from VCF files in order to conduct WSS method. As illustrated in Table 5.1, WSS file is composed of several columns including CHROM, POS, REF, ALT, GENE and an arbitrary number of individuals.

5.2 Overview of existing methods in genetic data watermarking

In this section, we present an overview of the state of the art in genetic data watermarking. Various methods have been proposed for genetic data watermarking. These methods can be classified accordingly many criteria. They can be classified based on their robustness against any modification illegal or not, the type of data to watermark, based on the imperceptibility of the watermark to insert or the technique used to insert the mark. The first level of classification we consider is the type of data to be watermarked. DNA watermarking schemes can be classified into two categories. First, there are methods that watermark DNA of living organisms in order to ensure copyright protection, etc. These methods must be robust in order to be able to resist against attacks or different biological modifications such as mutations. On the other hand, there are methods that watermark digital DNA with the aim of ensuring copyright protection, integrity of data or using DNA as a tool for pure storage of data only. In each of these categories, methods can be reversible or not; blind or not and most of them are based on one of three main techniques that are substitution, insertion or complementary between the bases.

5.2.1 Genetic data as medium for data storage

With rapid advances in genetic data processing, a large amount of data is available for various genetic studies, in particular genome-wide association studies. These data can be also used as storage medium due to their high information density and long-term storage. Therefore, digital information such as text message, audio or image can be hidden in DNA for long-term storage. Herein, we give some examples of methods that use DNA as storage medium. In the literature, several

methods were proposed so as to ensure the data storage within DNA [22–25, 296]. Clelland *et al* [22] proposed the first method for hiding secret messages into DNA. Inspired by the mocrodots used during the world war II, authors constructed artificial DNA strands in which secret messages are inserted. To do so, they construct a dictionary where each character is encoded using a triplet of DNA bases. Then, a simple substitution is used in order to encode English characters into DNA sequences. After that, these fake DNA sequences are mixed with human DNA sequences. In the same idea, Church *et al* [296] proposed a scheme using high fidelity DNA microships in order to encode 5.27 MB of data including a book, 11 JPEG images and a JavaScript program into DNA sequences of 54 898 159 bases, for long term storage. All encoded data were recovered with only 10 bit errors. Blawat *et al* [24] improved the schema of Church *et al* [296] by developing forward error correction codes so that they can recover all data without errors. In the following section we will describe methods that have been proposed for DNA watermarking or DNA steganography in order to ensures copyright protection, integrity, authenticity for DNA data as well as the protection of the message itself.

5.2.2 DNA watermarking and DNA steganography

In the literature, there are many methods proposed for genetic data watermarking or steganography. Herein, we give state of the art of these methods. As introduced before, genetic data watermarking methods that proposed for protecting genetic data or data hiding in genetic data without altering biological functionalities of the DNA. Theses method are divided in two categories: methods that watermark DNA of living organisms and methods that watermark DNA in numerical format.

5.2.2.1 DNA Watermarking restrictions

DNA watermarking or DNA steganography must not add mutations, remove or reduce the biological functionalities of carrier organism. In order to avoid these issues, DNA watermarking methods must ensure that watermarked DNA sequences are equivalent to those which are not watermarked. We discuss in this section some constraints that must be considered before developing a DNA watermarking scheme which is secure and resistant to all types of mutations. These constraints are as follows:

- *Primary structure preservation*: During message embedding, translation of amino acids into proteins for a given gene may not be altered. This means that nucleotide insertions and modifications must not alter the codons in such away that would change the original amino acid sequence. Watermarking schemes are restricted to embed messages by replacing synonymous codons i.e., codons which translate the same amino acid.
- *Truly nonfunctional regions*: As we will see in the next section, each DNA sequence has two regions: a coding region that encodes proteins and a non-coding region that does not encode proteins. For many years, the no-coding region which is called "junk DNA" was considered as non-functional region. Therefore, this region can be modified without constraint and data can be embedded in it without any restriction. However, recent studies have demonstrated

that even though this region does not encode proteins, up to 80% of this region may have other biochemical functions [39]. As a result, the message embedding in this region must performed in only 20% of this region which remains with no function.

- *Blindness without Appending*: For each DNA watermarking method, a blind detection and recovery which detects the watermark and/or recovers the original DNA sequence without using the original DNA sequence or a reference sequence, should be practicable while preserving the length of the DNA sequence.

- *Codon count preservation*: The other constrain to consider during DNA watermarking is gene optimization or the distribution of codons in organisms. In coding regions, gene optimization dictates the gene expression levels in living being's organisms, in particular, the speed at which genes' amino acids are translated into proteins [297]. Thus, it is desirable that the codon count in a given coding region be preserved when such a region is modified in order to embed messages.

- *No start/stop codons*: In DNA sequences, a non-coding region should not be mistaken as a coding region during protein synthesis. This means that during DNA watermarking, insertion or modification of nucleotides that introduce start codons should not be allowed.

- *No homopolymers*: Homopolymer is a region in a genome or DNA sequence where the same base is repeated multiple times. As many repeats can cause errors during DNA replication [298], DNA watermarking scheme must not include many and homopolymers in DNA sequences.

- *Dynamic range*: As explained in Chapter 1, nucleotide bases are described by one of four character symbols A, T, C, and G for each DNA sequence (G is replaced by U for RNA sequence). For most DNA watermarking methods, each nucleotide base is encoded with 2-bit representation. The 2-bit capacity for nucleotide bases is extremely low for high-capacity watermarking. A combination of nucleotide bases should be used to increase the dynamic range for more effective processing. For example, a series of four nucleotide bases can be coded with 8-bit values (256 levels).

In the sequel, we give a review on different methods that were proposed for DNA watermarking. Some of these methods consider previous constraints during the message embedding but it is very difficult for a DNA watermarking scheme to satisfy all of these constraints.

5.2.2.2 Watermarking of DNA data of living organisms

Several schemes were proposed in order to permit data hiding into DNA of living organisms. The first method in this category was proposed by Arita *et al.* [299]. This scheme permit the insertion of a digital signature in ncDNA regions of the genome of bacteria called Bacillus subtilis. To do so, they modify the redundant nucleotides of the wobble codons in the gene *stsZ*. Thus, if the message bit to insert is 0 the codon is not modified but if the bit to insert is 1, wobble nucleotide of the codon is modified to any nucleotide with the respect of the redundant property. The major

drawback of this method is that the method is non-blind, it requires the original DNA sequence for the receive to extract the hidden message. In addition, if any mutation is occurred, there is non proposed way to correct them and extract the message. To overcome this issue, Heider *et al.* [175] proposed DNA-Crypt which consists two watermarking methods. These permit the insertion of a message in ncDNA and pcDNA respectively. The method that inserts the message into ncDNA is a simple substitution. To do so, a binary message is mapped to bases to produce a fake DNA sequence before replaces some bases in ncDNA regions by the bases of the message. The proposed method that insert message in pcDNA region is based on genetic code. In this method, the message is inserted based on the modification of wobble nucleotide like Arita *et al.*'s method. To correct mutations or other biological phenomena that can complicate the extraction of the message, authors proposed a fuzzy controller that permit the correction of errors. It uses Hamming code for mutations differ in only one bit, and WDH code for mutations that differ by multiples bit. Authors tested their method *in vivo* using *Saccharomyces cerevisiae* [300]. Instead of using error correction codes, Yachie *et al.* [301] proposed the use of repetition coding as during the message embedding. More clearly, when a message is inserted, authors suggested that mutation errors may be corrected by embedding redundant copies of the message throughout an organism's genome. To test their method, a binary message was inserted in ncDNA of the genome and the inserted message was well recovered after some simulated mutations.

All above methods are not blind, the receiver must have the original sequence in order to extract the message. To overcome this problem, Liss *et al* [302] proposed the first blind method that insert messages in pcDNA especially into open reading frames (ORF) of synthetic genes considering gene optimization. To do so, they designed ORFs of the watermarked genes using codon usage table of host genes that were already optimized for protein expression. Then, all redundant codons are ranked according to their natural occurrence. For instance, for a given amino acid, codons with odd ranks in this table represent a binary 0 and codons with even ranks represent a binary 1. The message bits are inserted into four or six synonymous codons that retain a high degree of codon assignment flexibility. Even though this method is blind and retains gene optimization, it does not consider the mutation resistance.

Haughton *et al.* [34] proposed two DNA watermarking methods a cpDNA based method and a ncDNA based method which they called "BioCode ncDNA" for non coding regions and "BioCode pcDNA" for coding regions. These methods were designed under many constraints in order to permit the the insertion of messages in living organisms and reinforce the security of methods. For "BioCode ncDNA", they proposed how to preserve *no start codon*. This means that the modification of ncDNA bases during the message embedding could not introduce a start codon as explained in previous section (See Section 5.2.2.1). This method is an extension of non-coding version of DNA-crypt [175]. For "BioCode cpDNA", they ensure that the embedded message does not compromise the translations of amino acids into proteins and must preserve the codon count as each species has its own codon rate. In order to do that, a lookup table was designed, and this table maps a set of available codons to message bits according to a dynamic graduated mapping so as to respect two previous constraints. First, the lookup table is initialized according to the codon count. The codon of the DNA sequence to watermark is substituted with a codon that corresponds to input message bits of the lookup table. The count of this codon is then decreased by one. If the

count of any codon is zero, the lookup table is updated using dynamic graduated mapping. This process is repeated continuously until the message bits or the host codon sequences end. In order to correct mutations, authors proposed to marker codes or watermark codes for message bitframe resynchronization. BioCode methods permits to preserve codon count, protein translation and no start codons but do not efficiently resist to intentional mutations and are therefore not sufficiently secure.

More recently, Wang *et al.* [33] proposed a DNA data hiding in living organisms. In this method, a secret message is encoded in a DNA sequence by a certain coding rule. Then, the result is hidden in DNA of a living organism by recombining DNA technique. Therefore, the proposed scheme has two hidden layers, if one of the two layers is cracked, the other one can also ensure the security of secret message.

5.2.2.3 Watermarking of DNA data in numerical format

In this section we discuss, DNA watermarking methods that watermark numerical data. These methods were proposed for ensuring copyright protection, integrity of authenticity of data, as well as the confidentiality of hidden messages.

Shiu *et al* [26] proposed three reversible data hiding methods in DNA data called the insertion, the complementary pair and the substitution methods. In the insertion method, secret message bits are inserted randomly in separated positions within a DNA sequence to watermark. In the complementary pair method, authors proposed to choose the longest complementary pairs in a DNA sequence. Therefore, secret message bits can be hidden before in these complementary pairs. Finally, the substitution method which consists on substituting some part of chosen DNA nucleotides in the sequence with others based on the message bits. These schemes could hold high embedding capacity. However, both the complementary pair and the insertion methods expand original DNA sequences. In addition, in each of these three methods, a reference DNA sequence (original sequence is transmitted so as to permit the receiver to recover the hidden message), and they did not focus on the expansion or DNA modification rate.

To reduce the modification rate and resolve the expansion problem of these methods, Huang *et al* [303] proposed a new reversible data hiding scheme based on histogram technique. In this method, the DNA sequence to watermark is first converted into a binary string. Then, several bits are combined in order to produce a sequence $\{d_1, d_2, \cdots, d_n\}$ of decimal integers and an histogram is generated based on these decimal integers. After generating the histogram, the most frequent integer h, the least frequent integer l_1 and the second least frequent integer l_2 are identified and a location map is initialized. For embedding message bits, if the decimal integer d_i is equal to l_1, set d_i to l_2 and set the value of location map to 1. If the decimal integer d_i is equal to l_2, the decimal integer d_i remains unchanged and the value of the location map is set to 0. Besides, if the decimal integer d_i is not equal to l_1 or l_2, the decimal integer d_i remains unchanged and we do not need to set the location map. In order to recover the original DNA sequence, the location map must be concealed into the DNA sequence with secret message. If d_i is equal to h and the message bit is equal to 0, d_i does not change. Otherwise, if d_i is equal to h and the message bit is 1, set

d_i to be l_1. We can then obtain new decimal integers. These decimal integers are converted into binary string which is also converted into DNA sequence in order to obtain a watermarked DNA sequence. Note that the message extraction is conducted in the same way. This method resolve many problems of Shiu *et al*'s method but it still have several limitations such as sending the original sequence for message extraction. This is also the case of methods proposed in [304–306]. These methods are not blind and this reduces their security.

Several methods have been proposed in order to overcome the problem of blindness in DNA watermarking [27–33]. For instance, in [27], authors presented a blind reversible watermarking method that prevent biological mutations. It is based on multilevel histogram shifting. In this method, a DNA sequence is encoded into integer values using the numeric order. multiple bits are embedded in each integer value by exploiting multilevel histogram shifting of noncircular type (NHS) and circular type (CHS). During message embedding a verification of each codon is conducted so as to prevent the generation of false start/stop codons. To do so, they check whether a start/stop codon is included in an integer value or between adjacent integer values. Rahman *et al* [29] proposed another reversible and blind DNA watermarking method that can embed a secret identification message in order to ensure the authenticity and copyright protection of a DNA sequence. In this method, a DNA sequence to watermark is divided into multiple segments and these segments are used to construct a matrix of nucleotide bases. After that, a message to embed is also divided into segments and each message segment is inserted in each line of the matrix. Note that, positions in which message bits are embedded are chosen randomly based on a pseudo-random generator which allows the generation of matrix indices.

All solutions presented in this section allow genetic data watermarking for various purposes but have also several limitations. They were proposed for cellular DNA, and they can not be used for genetic data that are outsourced for genome-wide associations studies (e.g., genetic variants stored in VCF files). To overcome these issues, we are the first to propose a robust watermarking for GWAS data. It allows ensuring traceability and traitor tracing for genetic data externalized for GWAS studies. Our watermarking proposal is bling in the sense that to extract the watermark we do not need the original VCF file, and the watermarked file has the same size that the original one.

5.3 Proposed database watermarking scheme for GWAS data

In this section, we first present a common chain of database watermarking [106], the way we have exploited the modulation of Kuribayashi *et al.* [307] in order to watermark genomic data and, by next, the watermarking solution we propose. Before entering into the details and in order to simplify the comprehension of our scheme, we illustrate in table 5.2, the acronyms that we have used in this chapter.

5.3.1 Database watermarking

As explained in Chapter 1, a database is an organized collection of data that are generally stored and accessed from a computer system. As formally defined in chapters 1 and 3, we keep *DB*

Table 5.2: Acronyms used in the watermarking method we propose

Acronyms according to WSS file	
Δ	Distortion factor
N_c	Number of columns in the WSS file
D_Δ	Percentage of modulation for a given Δ
N_g	Number of groups
S_g	Number of rows for each group
N_r	Number of rows in the WSS file (i.e $S_g \times N_g$)
db_{size}	Size of WSS file (i.e $N_r \times N_c$)
P_i	Probability value of *i* in original group
P_i^w	Probability value *i* in watermarked group

as a database *DB* composed by N_R relations $\{R_i\}_{i=1,\cdots,N_R}$. To give an overview on database watermarking, let us consider un database *DB* that contains one single relation constituted of *N* tuples $\{t_u\}_{u=1,\cdots,N}$, each of *M* attributes $\{A_1, A_2, \cdots, A_M\}$. The attribute A_n takes its values within an attribute domain and $t_u.A_n$ refers to the value of the n^{th} attribute of the u^{th} tuple of the relation. The value $t_u.PK$ which is an attribute value or a set of attribute values, represents the unique identifier of each tuple in the database, and is called primary key.

In the literature, most schemes that have been proposed for database watermarking follow the process illustrated in Figure 5.1. This process is based on two fundamental procedures: watermark embedding and watermark detection/extraction. The watermark embedding procedure includes a pretreatment, the purpose of which is to make the watermark insertion/extraction independent of the database structure or the way database's data is stored. To do so, database tuples are grouped into N_g non-overlapping groups $\{G_i\}_{i=1,\cdots,N_g}$. This grouping is usually conducted by computing the index number $n_u \in [0, N_g - 1]$ of each group for the tuple t_u [308] such that

$$n_u = H(K_w|H(K_w||t_u.PK)) \mod N_g \tag{5.1}$$

where H, K_w and $||$ represent the cryptographic hash function, the secret watermarking key and the concatenation operator, respectively. We use a cryptographic hash function, such as the Secure Hash Algorithm (SHA), in order to ensure the secure grouping and the equal distribution of tuples into different groups. After database partitioning, one bit of the watermark is inserted into each group of tuples by modifying or modulating attribute values accordingly the rules of the retained watermarking modulation such as the order of database tuples [196]. Therefore, within a database of N_g groups, a watermark $W = \{w_i\}_{i=1,\cdots,N_g}$ of N_g bits can be embedded.

The Watermark detection works in a similar way. First, the protected database is partitioned into N_g groups based on the secret watermarking key K_w. Then, one watermark bit is detected and/or extracted from each group based on used modulation and the use of the scheme. In the sequel, we explain the solution we have proposed. It follows the above procedures and is based on Quantization Index Modulation (QIM) and majority vote.

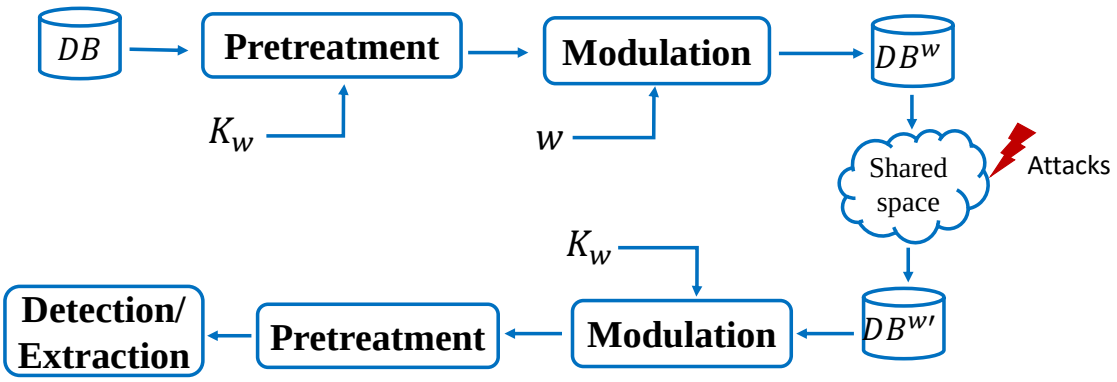

Figure 5.1: A common database watermarking chain.

5.3.2 Quantization Index Modulation (QIM) watermarking

Quantization Index Modulation (QIM) is a watermarking technique that is based on the quantifying some elements (samples, group of samples or transform coefficients) of a host data (e.g., image [309], video [310], etc.) according to a set of quantizers based on codebooks in order to embed the watermark. More clearly, to each message m_i issued from a finite set of U possible messages $M = \{m_i\}_{i=0,\cdots,U}$, the QIM associates a codebook $\{C_{m_i}\}_{i=0,\cdots,U}$ such that

$$C_{m_i} \cap C_{m_j} = \emptyset \quad \text{if} \quad i \neq j \tag{5.2}$$

For embedding the message m_i into one element X of a given host data, this one is substituted by X_w which is the nearest element of X in the codebook C_{m_i}. This process is conducted using the insertion function Q_{m_i} such that

$$X_w = Q_{m_i}(X, C_{m_i}) \tag{5.3}$$

This function determines the the nearest element X_w of X from the codebook C_{m_i}. In this case, the watermarking distortion corresponds to the distance between X and X_w. To give a simple example which illustrates this process, let us consider the case of an image with X that represents an image pixel. This latter may take its values from a one-dimensional space $[0, 255]$. This scalar space is divided into non-overlapping cells or intervals of equal size. Each cell is then related to only one codebook $\{C_{m_i}\}_{i=0,\cdots,U}$ so as to satisfy (5.2). Consequently, m_i has several representations in $[0, 255]$ and Q_{m_i} corresponds to a scalar quantizer. In the embedding process, if X belongs to a cell that encodes the desired symbol m_i, its watermarked version X_w corresponds to the centroid of this cell. Otherwise, X is replaced by the centroid of the nearest cell encoding m_i. In the extraction process, the knowledge of the cell to which X_w belongs is enough to identify the embedded message. This process is illustrated in figure 5.2 in the case of a binary message, i.e., $m_i \in \{0, 1\}$ and two codebooks C_0 and C_1 for which the cells are defined according to a uniform scalar quantization of quantization step Δ. In this example, X will be quantized to the nearest square or circle in order to encodem_i.uring message extraction, the watermark reader has to determine the cell to which the received version X'_w of X_w belongs. We explain in the next section, how a modified version of QIM is adapted for watermarking genetic data.

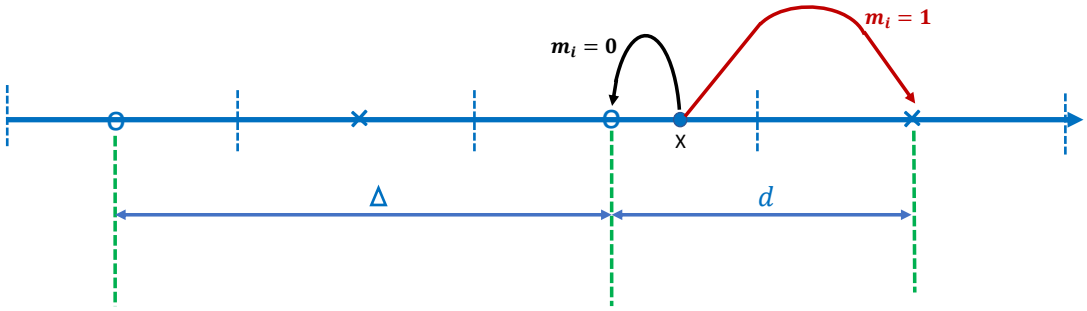

Figure 5.2: Example of QIM in the case where X is a scalar value for the embedding of a sequence of binary values . Codebooks are based on an uniform quantization of quantization step Δ. Cells centered on crosses represent C_0 $(m_i = 0)$ while cells centered on circles represent C_1 $(m_i = 1)$. $d = \Delta/2$ establishes the measure of robustness to image perturbations.

5.3.3 Modified QIM for genetic data watermarking

Kuribayashi *et al.* [307] proposed a robust and secure data hiding scheme for PDF text documents, by embedding a watermark into the spaces among characters in each line. The collection of the space lengths in each line is denoted as a host vector and the watermark is embedded into its frequency component based on Dither Modulation-Quantization Index Modulation (DM-QIM). In using QIM, they propose to round the host frequency component to the nearest odd/even quantized value according to the value of watermark bit w using step size Δ. Let $w \in \{0,1\}$ be a watermark bit, Δ be a quantization step size that controls the level of distortion and d an element of the selected host signal. In QIM method, according to the value of the watermark bit to be embedded in frequency component, this operation consists in shifting $\pm\Delta$ the DCT (Discrete cosine transform) coefficient of the collection of the space length in the t^{th} line.

In this work, we apply this QIM method in order to embed one watermark bit w_i into each group of tuples, i.e. $\{G_i\}_{i=1,\cdots,N_g}$. More clearly, let $w_i \in \{0,1\}$ be a watermark bit, Δ be a quantization step size that control the level of distortion and d be the difference between the cardinality of zero values in sub-group G_i^A and sub-group G_i^B for each individual (P_i: $i = 1, \cdots, |\text{patients}|$), where

$$d = |C_0^A|_{P_i} - |C_0^B|_{P_i} \tag{5.4}$$

According to the value of w_i, d is rounded to the nearest even/odd quantized value using step size Δ. As illustrated in Figure 5.3, the global quantized values are in the multiple of Δ either as positive or negative, and embedding modulation is performed as follows:

$$d^* = (\lfloor \frac{d}{\Delta} \rfloor + (\lfloor \frac{d}{\Delta} \rfloor_2 != w)) \times \Delta \tag{5.5}$$

5.3.4 Watermark embedding in WSS data

As explained in the previous chapter, in this work, we consider a framework which is composed by threes entities: a Genomic Research Unity (GRU), a Genomic Research Center (GRC) and a

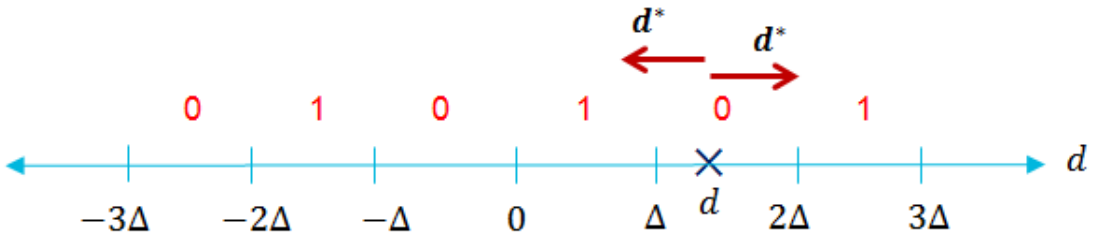

Figure 5.3: An example of QIM modulation.

Cloud Services Provider (CSP). GRU and GRC decide to outsource their genetic data on the cloud for storage and/or processing purposes. Before being outsourced, these data are watermarked so as to ensure their copyright protection and traitor tracing. To do so, we describe in this section a robust database watermarking scheme that allows message embedding for WSS data. Our solution is implemented through six following steps:

- **Step 1.** The first step consists on reading genotypes data from the WSS file which consists of many genes, into a database that composed by one table *DB*.
- **Step 2.** The table *DB* is secretly reorganized into the database DB^r. To do so, data owner assigns a primary key $v_u.PK$ for each variant $v_u \in \{u = 1, \cdots, |variants|\}$, where $v_u.PK = CHROM \| POS \| GENE$. Then, this primary key is used for partitioning the database (WSS file variants) into N_g groups using a secret watermarking key K_w. The group index number for each variant n_{v_u} is computed based on secure hash algorithm (SHA256) using (5.6) and N_g groups $\{G_i\}_{i=1,2,\cdots,N_g}$, are constituted.

$$n_{v_u} = SHA256(K_w(SHA256(v_u.PK|K_w)) \mod N_g \tag{5.6}$$

  Once all groups are obtained, one bit of the watermark is embedded into each group.
- **Step 3.** The user or data owner (in our case GRU or GRC) generates a binary watermark $W = \{w_1, w_2, \cdots, w_{N_g}\}$, where N_g is the number of groups in the database DB^r. W is uniformly distributed where the probability p to have 0 is equal to probability to have 1..
- **Step 4.** Each group G_i of the database is divided into two tuple sub-groups G_i^A and G_i^B, based on the secret watermarking key K_w. To do so, the sub-group index number n_{gv_u} for each variant v_u in G_i, is computed using secure hash algorithm (SHA256) such that

$$n_{gv_u} = SHA256(K_w||(SHA265(v_u.PK||K_w)) \mod 2 \tag{5.7}$$

  If the value $n_{gv_u} = 1$, then the variant v_u belongs to G_i^A, otherwise ($n_{gv_u} = 0$), then it belongs to G_i^B.
- **Step 5.** QIM modulation is used for embedding one watermark bit in these sub-groups so as to produce the watermarked sub-groups G_i^{AW} and G_i^{BW}. The watermark embedding process is illustrated in Algorithm 3.
- **Step 6.** After sub-group watermarking, the watermarked database DB^w is constituted.

Algorithm 3 Watermark embedding modulation in one group

INPUT: Subgroups G_i^A and G_i^B, A watermark bit w_i, a quantization step size Δ
procedure GROUPWATERMARKING(G_i^A,G_i^B,w_i,Δ) $d = \|C_0^A\| - \|C_0^B\|$
$d \leftarrow \lfloor \frac{d}{\Delta} \rfloor$

    if d % 2 == w **then**
        d^* = $\tilde{d} \times \Delta$
    else
        d^* = $d \times \Delta + \Delta$
        modulationValue = abs(d^* - d)
Case 1
        if d* $\geq$ d and $\|C_0^B\| \geq$ modulationValue **then**
            $\|C_0^{BW}\|$ = $\|C_0^B\|$ - modulationValue
Case 2
        else if d* $<$ d and $\|C_0^A\| \geq$ modulationValue **then**
            $\|C_0^{AW}\|$ = $\|C_0^A\|$ - modulationValue
Case 3
        else
            not embeddable group
        end if
    end if
    return G_i^{AW},G_i^{BW}
end procedure

Notice that during watermarking, one watermark bit is embedded in each database column. Thus, during extraction stage a majority vote is performed in order to decide which watermark bit will be extracted.

Watermark reading works in a similar way. The watermarked database is first reorganized into N_g groups and each group is partitioned into two sub-groups. From each group, one message bit is detected and extracted in each column according to the equation (5.8). After that, a majority vote is conducted in order to decide which watermark bit is extracted. While tuple primary keys are not modified, the knowledge of the watermarking key ensures synchronization between watermark embedding and watermark detection/ extraction.

$$w = \lfloor \frac{d^* + \frac{\Delta}{2}}{\Delta} \rfloor \% 2 \tag{5.8}$$

where

$$d^* = |C_0^{AW}|_{P_i} - |C_0^{BW}|_{P_i}$$

We discuss theoretical performances of our solution in next section before presenting experimental results.

5.4 Theoretical performance

In this section, we start by presenting the constraints of some parameters in the proposed model and then present the theoretical performance of our scheme in terms of distortion introduced to data

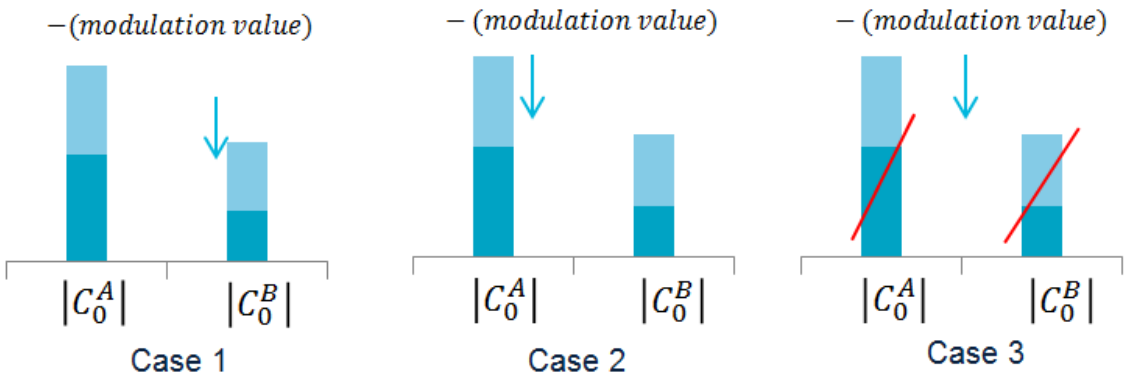

Figure 5.4: Embedding modulation cases

during watermarking embedding, embedding capacity and robustness against different database watermarking attacks.

5.4.1 Parameter constraints

In our solution, in order to work properly and intuitively, some constraints such as distortion factor (Δ), number of groups in the database (N_g), number of tuples in the database (N_r) and the probability to have 0 in one group (P_0) must to be defined and respected. These constraints are such that

$$\begin{aligned} &\frac{S_g}{2} > \Delta \\ &\frac{N_r}{N_g} > 2 \times \Delta \\ &N_r > 2 \times \Delta \times N_g \\ &P_0 \times N_r > 2 \times \Delta \times N_g \end{aligned} \tag{5.9}$$

this constrain is important, because the number of zeros in a sub-group should be greater than the distortion factor Δ. As we will see later, this constraint will help us in analysing the performance of our watermarking method.

5.4.2 Distortion performance

Let us consider a database DB which contains N_c columns, N_r rows and db_{size} attribute values. During the watermarking process, this database is divided into N_g groups and each group is partitioned into two sub-groups. If S_g is the number of rows in one group. Then, the distortion value D_Δ for the database DB corresponds to the number of modified attribute values in the database for a given Δ, and can be computed as follows:

$$D_\Delta = N_g \times \frac{\Delta}{2} \times N_c \tag{5.10}$$

$$D_\Delta = \frac{N_r}{S_g} \times \frac{\Delta}{2} \times N_c \tag{5.11}$$

$$D_{\Delta} = db_{size} \times \frac{\Delta}{2 \times S_g} \tag{5.12}$$

As example, if we take Δ = 2 and S_g = 100, then we can say that the distortion is $\frac{1}{50}$ of the db_{size}. This is due to symmetric distribution for the difference value of zero frequency between sub-group G^A and sub-group G^B.

5.4.3 Robustness performance

In this section, we analyse the robustness of our watermarking scheme under three well-known database attacks that are deletion attack and insertion attack. We evaluate the robustness of our solution by means of the bit error rate (BER), which corresponds to the ratio of the number of incorrectly extracted watermark bits to the number of the original watermark bits. BER is such that

$$BER = \frac{\sum_{i=1}^{N_g} w_i \oplus w_i^{'}}{N_g} \tag{5.13}$$

where w_i and $w_i^{'}$ are the embedded watermark bit and the extracted watermark bit, respectively. Using BER, its lower value means that we have a higher watermarking robustness.

5.4.3.1 Deletion attack

In this section, let us consider an attack that consist at a randomly deletion of attribute values or tuples in the database. We distinguish two cases for this attack and are described below.

- **Column deletion:** In this case, an attacker tries to delete N_{c_1} columns in the database. No matter how many columns are deleted, one column is enough to detect the watermark if all columns are watermarked.
- **Tuple deletion:** Let us consider the attacker randomly eliminates N_d tuples in the database. In this case watermark may not be detected depending on the percentage of deleted data and the group in which deleted elements belongs. We will come back to this case in section 5.5, where we demonstrated the robustness of our solution against this attack using BER.

5.4.3.2 Insertion attack

In this kind of attacks, an attacker may tries to insert a certain number of columns or tuples in the database. Two cases are distinguished as described below.

- **Column insertion:** In this attack, an attacker tries to insert a certain number if columns in the database. By doing so, it requires to an attacker to duplicate at least one time the number of columns (or individuals) so as to change the watermark bit. Assume that the original group verifies the probability to have 1 values is greater than the probability to have 0 values ($P_1 > P_0$). Then, the watermarked group verifies $P_0^w > P_1^w$. Hence, we can define

$X = P_1 - P_0$ and $X^w = P_0^w - P_1^w$. There are three cases for which the data can be added by attacker.

- **Case 1:** If $P_1 < P_0$, there is no problem as the attacker will be always detected.
- **Case 2:** If $P_1 = P_0$, as in the previous case, the attack will always be detected.
- **Case 2:** If $P_1 > P_0$, the attacker requires to add *M* elements in the database such that

$$M = \frac{N_c \times X^w}{X} \tag{5.14}$$

- **Tuple insertion:** This attack corresponds to the suppression a certain number of tuples in the database. If *N* is the number of tuples that the attacker want to insert in the database. Let *k* be the number of success out of the total number of trials and *p* the probability to succeed, while *q* is the probability of failure. Thus, we have

$$p = \frac{1}{2 \times N_g}$$

$$q = 1 - p$$

The probability of *k* successes out of *N* trials when the probability of one success is *p* is computed according to the equation (5.15)

$$P(N, k, p) = \begin{pmatrix} N \\ k \end{pmatrix} p^k q^{N-k} \tag{5.15}$$

In the previous equation (5.15), the binomial coefficient express the number of combinations of *N* takes *k*. It is calculated according to equation (5.16).

$$\begin{pmatrix} N \\ k \end{pmatrix} = \frac{N!}{(N-k)!k!} \tag{5.16}$$

We give in next section, obtained results after simulating some attacks.

5.5 Experimental results and discussion

The purpose of this section is to evaluate our watermarking method in terms of distortion, robustness and watermarking capacity in the framework of one real genetic database.

5.5.1 Test database

To experiment our watermarking method, we have used a genetic relational database constituted of one table of 80 tuples issued from a real genetic database that contains pieces of information related to genetic variants of 733 individuals. As we have in Chapter 4, such genetic variants are used by researchers or/and geneticists in genome-wide association studies (GWAS) [6] in order to determine if there is a relationship between these genetic variants and certain diseases. In our test database, one tuple corresponds to one variant and is composed by 738 attributes including 5

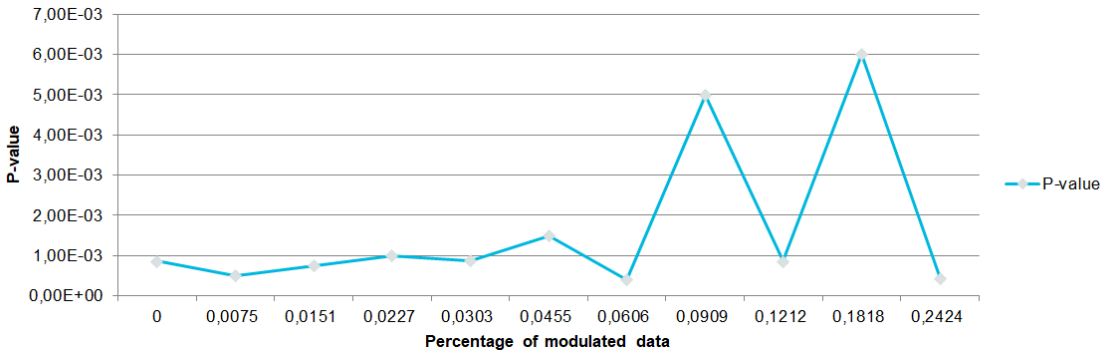

Figure 5.5: Distortion percentage of modulated data

first ones that give information about the genetic variant. They correspond to: the identity of the chromosome ($chrom$) to which belongs the variant, the position (pos) of the variant at the chromosome, the reference allele (ref), the alternative allele (alt) and the name of the gene ($GENE$) in which belongs the variant. The rest of attributes correspond to individual genotypes in the database. As explained in Chapter 1, for each individual and each variant, the genotype corresponds to an integer value that takes the value 0 if the alternative allele is equal to the reference allele, 1 to the second alternative allele and $k \in \{1, \cdots, g\}$ in case of g possible alternative alleles. In the sequel, a set of attributes that composed by the chromosome ($chrom$), the position (pos) and the gene ($GENE$) is considered as the primary key. We chose these attributes because their combination uniquely identifies each database tuple of variant. Our watermarking scheme was implemented in Python and we conduct all experiments using a machine equipped with 8 GB RAM running on Ubuntu 18.04 LTS.

5.5.2 Distortion results

As introduced at the beginning of this chapter, the objective of our watermarking method is to ensure copyright protection and traitor tracing for genetic data used in GWAS. In order to test the impact of watermarking method for GWAS results, we have conducted secure WSS method presented in Chapter 4. It is one of GWAS algorithms that are used in order to conduct association studies for rare variants. To test our watermarking method on the database presented in Section 5.5.1, this database is divided into N_g groups, considering several cases. These cases correspond to $N_g \in \{1, \cdots, 20\}$. We have also chosen different values of distortion step Δ such that $\Delta \in \{2, 4, 6, \cdots, 34\}$ the each group is also divided into two sub-groups. After watermarking, we conducted WSS method on the watermarked database in order to measure the distortion introduced by the watermark into watermarked data. As it can be seen in in Figure 5.5 that contains the values of $p - value$, results have the same significant value if they remain on the same order. In addition, the variation of the $p - value$ depends on the number of elements that are watermarked in the database, and the value 0 corresponds to the original $p - value$. The table 5.3 presents all $p - value$ results for above chosen N_g and Δ, and most of them are still significant compared to the original $p - value$.

Table 5.3: P-value results.

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	4.9×10^{-4}	3.5×10^{-4}	4.2×10^{-4}	4.4×10^{-3}	5.9×10^{-4}	2.9×10^{-3}	3.0×10^{-4}	5.9×10^{-4}	5.3×10^{-4}	5.4×10^{-4}	3.4×10^{-3}	7.9×10^{-3}	5.9×10^{-4}	3.9×10^{-3}	5.9×10^{-3}	3.9×10^{-3}	3.7×10^{-4}
2	5.9×10^{-4}	4.9×10^{-4}	6.6×10^{-4}	1.4×10^{-3}	4.2×10^{-4}	5.9×10^{-3}	5.9×10^{-4}	3.6×10^{-4}	9.9×10^{-4}	2.9×10^{-3}	3.8×10^{-4}	5.9×10^{-3}	2.9×10^{-3}	3.4×10^{-3}	2.3×10^{-3}	5.4×10^{-3}	9.9×10^{-4}
3	4.2×10^{-4}	1.9×10^{-3}	1.1×10^{-3}	4.9×10^{-4}	6.9×10^{-4}	1.1×10^{-3}	7.4×10^{-4}	1.1×10^{-3}	4.2×10^{-4}	5.8×10^{-4}	4.9×10^{-4}						
4	6.6×10^{-4}	6.6×10^{-4}	3.9×10^{-3}	2.9×10^{-3}	5.2×10^{-4}	5.4×10^{-4}	5.9×10^{-4}	7.7×10^{-4}	2.4×10^{-4}								
5	5.4×10^{-4}	7.4×10^{-4}	2.3×10^{-4}	8.5×10^{-4}	6.6×10^{-4}	2.9×10^{-3}	4.2×10^{-4}	8.7×10^{-4}									
6	5.9×10^{-4}	5.4×10^{-4}	9.9×10^{-4}	2.9×10^{-3}	2.9×10^{-3}	2.3×10^{-3}											
7	5.9×10^{-4}	6.6×10^{-4}	5.4×10^{-4}	1.3×10^{-3}	6.6×10^{-4}												
8	5.4×10^{-4}	3.7×10^{-4}	4.6×10^{-4}	6.6×10^{-4}													
9	1.1×10^{-3}	4.4×10^{-3}	6.6×10^{-4}														
10	3.3×10^{-4}	1.9×10^{-3}	2.3×10^{-4}														
11	2.3×10^{-3}	3.5×10^{-4}															
12	7.7×10^{-4}	2.3×10^{-3}															
13	9.9×10^{-4}	4.2×10^{-4}															
14	2.6×10^{-4}	5.9×10^{-4}															
15	1.4×10^{-3}	8.9×10^{-3}															
16	7.4×10^{-4}																
17	4.2×10^{-4}																
18	1.9×10^{-3}																
19	2.9×10^{-3}																
20	4.2×10^{-4}																

Table 5.4: BER results against column deletion 10%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0.48	0.12	0.63	0.37	0.24
3	0	0	0	0	0	0	0	0	0.47	0.27	0.25	0.43					
4	0	0	0	0	0	0.12	0.24	0.48	0.43								
5	0	0	0	0	0.09	0.15	0.47	0.39									
6	0	0	0	0.09	0.23	0.31											
7	0	0	0.04	0.09	0.53												
8	0	0	0.09	0.24													
9	0	0	0.16														
10	0	0															
11	0	0															
12	0	0.02															
13	0	0.12															
14	0	0.17															
15	0.04																
16	0																
17	0																
18	0.06																
19	0.06																
20	0.05																

5.5.3 Capacity results

In a database watermarking scheme, watermarking capacity is evaluated by the ratio of database elements that can be used for watermark embedding to the total number of elements in the database. Higher watermarking capacity means that more watermark information that we can embedded in the database. The watermarking capacity of our solution depends on the number of embeddable groups that we have in the database. This capacity can reach 100 % depending on genotypes that we have in the database. This means that in some cases, each group in the database can embed a watermark bit. However, if the capacity is the maximum, the robustness is reduced.

5.5.4 Robustness results

To test the robustness of our solution against different attacks, we have simulated several attacks including addition or deletion of columns in the watermarked database. We have considered an attacker that can try to insert, delete 10%, 20% and 30% of the data in the database. Obtained results are presented in tables 5.7,5.8, 5.9, 5.4, 5.5 and 5.6. In these results, watermark can be correctly detected from the database when BER approaches zero. Moreover, our scheme do not impact the p-value results of WSS after watermarking. The Figure 5.6 shows the variation of BER in function of the rate of database elements that have been changed, during addition and deletion attacks.

Since majority voting [35] is used during message extraction, these attacks will have no impact on the watermark.

Table 5.5: BER results against column deletion 20%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0.48	0.12	0.44	0.37	0.24
3	0	0	0	0	0	0	0	0	0.47	0.27	0.25	0.43					
4	0	0	0	0	0	0.12	0.24	0.48	0.43								
5	0	0	0	0	0.13	0.15	0.47	0.39									
6	0	0	0	0.09	0.23	0.31											
7	0	0	0.04	0.09													
8	0	0	0.09														
9	0	0															
10	0	0															
11	0	0															
12	0	0.05															
13	0	0.12															
14	0	0															
15	0.04																
16	0																
17	0																
18	0.05																
19	0.06																
20	0.05																

Table 5.6: BER results against column deletion 30%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0.48	0.12	0.56	0.37	0.24
3	0	0	0	0	0	0	0	0.15	0.47	0.27	0.25	0.43					
4	0	0	0	0	0	0.12	0.24	0.48	0.43								
5	0	0	0	0	0.13	0.15	0.47										
6	0	0	0	0.09	0.23	0.31											
7	0	0	0.04	0.09	0.53												
8	0	0	0.09	0.24													
9	0	0	0.1														
10	0	0															
11	0	0															
12	0	0.04															
13	0	0.12															
14	0	0.17															
15	0.04																
16	0																
17	0																
18	0.02																
19	0.06																
20	0.05																

5.6 Conclusion

In this chapter, we have addressed the copyright protection and traitor tracing for genetic data during their storage or processing on the cloud. As we have previously seen, these data may face several security problems on the cloud such as illegal distribution of data or problem of copyright.

We have presented a robust database watermarking method that allows watermarking of genetic data used in GWAS. It is the first method of this kind, and it can be used for statistical algorithms such as WSS method. It can be used in protecting traitor tracing and copyright protection, and it is based on Quantization Index Modulation (QIM) and majority vote. We have studied theoretical performance and experimentally verified the performance of our solution in terms of robustness

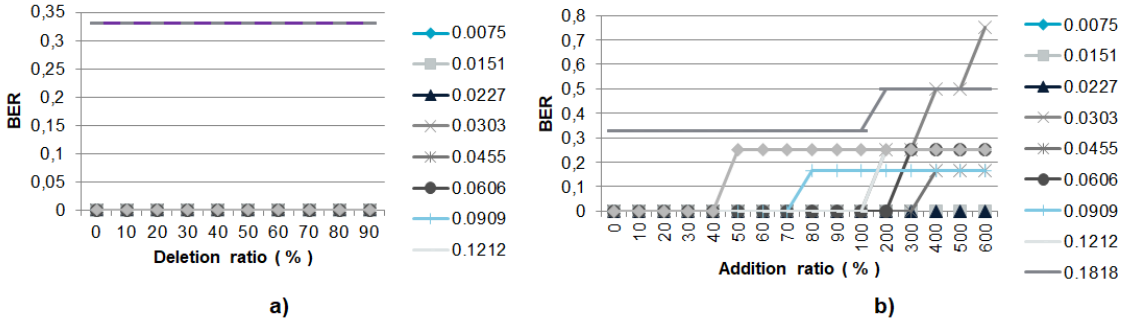

Figure 5.6: Robustness results against deletion and addition

Table 5.7: BER results against column addition 10%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0.48	0.12	0.69	0.37	0.24
3	0	0	0	0	0	0	0	0.19	0.47	0.27	0.25	0.43					
4	0	0	0	0	0	0.12	0.36	0.48	0.43								
5	0	0	0	0	0.14	0.15	0.47										
6	0	0	0.04	0.09	0.23	0.31											
7	0	0	0.09	0.24	0.53												
8	0	0	0.19	0.24													
9	0	0	0														
10	0	0															
11	0	0															
12	0	0.10															
13	0	0.12															
14	0	0.17															
15	0.04																
16	0																
17	0																
18	0.06																
19	0.06																
20	0.05																

Table 5.8: BER results against column addition 20%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0.40	0.48	0.12	0.69	0.37	0.24
3	0	0	0	0	0	0	0	0.51	0.47	0.27	0.45	0.43					
4	0	0	0	0	0	0.12	0.36	0.48	0.43								
5	0	0	0	0.05	0.14	0.24	0.47	0.39									
6	0	0	0	0.09	0.24												
7	0	0	0.04	0.24	0.53												
8	0	0	0.19	0.24													
9	0	0	0.22														
10	0	0															
11	0	0															
12	0	0.12															
13	0	0.14															
14	0	0.17															
15	0.04																
16	0																
17	0																
18	0.06																
19	0.06																
20	0.05																

Table 5.9: BER results against column addition 30%

$\frac{\Delta}{N_g}$	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0.40	0.48	0.55	0.69	0.37	0.24
3	0	0	0	0	0	0	0.11	0.51	0.47	0.27	0.45	0.43					
4	0	0	0	0	0	0.12	0.36	0.48	0.64								
5	0	0	0	0.05	0.34	0.24	0.64	0.39									
6	0	0	0	0.09	0.24												
7	0	0	0.04														
8	0	0	0.19														
9	0	0	0.004														
10	0	0															
11	0	0															
12	0	0.12															
13	0	0.14															
14	0																
15	0.04																
16	0																
17	0																
18	0.06																
19	0.06																
20	0.07																

against two attacks that are deletion attack and addition attack, capacity watermarking and distortion. In this method, a watermark is embedded in genetic data without altering results of association tests that can be conducted on these data. This comfort its future use in real life applications, especially in cloud environments.

Conclusion and perspectives

Nowadays, genetic sequencing has become more important as it allows generating large amount of genetic data. As a consequence, these data are getting widely collected, stored, processed and shared by health professionals, researchers or companies for various genetic applications such as disease tendency tests. To do so, cloud computing is being used as it allows the flexibly computation of large amount of data at a low cost. However, this comes with several issues in terms of data security. Indeed, a human genome has a sensitive nature and represents the unique biological identity of each individual. In addition, it can reveal the genetic origins of patients and possible corresponding diseases. What makes this information so sensible also makes it so valuable for research and medical purposes. By sequencing many genomes and cross-comparing the results, we can be able to understand new biological mechanisms, which leads to new diagnostic tools and treatments. For instance, genome-wide association studies conducted to some genes can help to study and prevent risks of illness or if there is no cure for the illness, anticipated genetic testing allow some life decisions (e.g., mastectomy in case of breast cancer). Thus, genetic data are submitted to satiric legislative and ethical rules and must be protected.

The protection of genetic data during their processing and storing on the cloud is the focus of this thesis work. Genetic data security can be expressed in terms of various security objectives such *i) privacy*, *ii) confidentiality*, *iii) integrity* and *iv) traceability*. Therefore, different security tools (physical or logical) must be defined in order to respond to these security objectives. However, choosing or developing a security mechanism for genetic data must respect several constraints depending on type of data to secure. For example, we have seen that genetic variants are stored on the cloud in order to be used for conducting different statistical algorithms that are used in genome-wide association studies. In this case, security mechanisms which do not compromise these studies must be preferred.

In this context, we have proposed a privacy-preserving method that based on fully homomorphic encryption [15]. We recall that homomorphic encryption allows a data owner to store their encrypted data on the cloud and they can ask the cloud to conduct processing on these data without need to decrypt them. After processing, obtained results are sent to the data owner in encrypted form. The method we have proposed allows the secure computation of collapsing method using a logistic regression model, and it uses fully homomorphic encryption, secure multiparty computation and multiplicative data masking so as to allow two entities that are a genomic research unity

and a genomic research center, to perform collapsing method on their outsourced encrypted data without decrypting them. In our solution, there is no estimations in statistical tests and thus, it achieves exactly the same results as association test conducted on clear data. This method protects the confidentiality of data but we have seen that these data may also face several security issues in terms of data integrity.

Thus, a second contribution of this work corresponds to the watermarking of homomorphically encrypted data [14], in order to ensure their integrity for the cloud service provider point of view. This method takes advantage of semantic security property of homomorphic encryption schemes so as to embed a watermark in encrypted databases without altering clear data. In addition, this method is dynamic, in the sense that it makes possible the protection of databases while maintaining them updated by their owners (e.g. tuple additions, tuple suppressions and encrypted attribute value modifications). The watermark embedding is conducting by modifying the center element of a subset of attribute values. This watermark which is a binary message can be used for database integrity verification. In fact, any differences between the extracted and the embedded watermarks will indicate the database integrity loss.

One of the constraints of the above schemes is that they are based on homomorphic encryption and this one is still having an important overhead in terms of storage, computation and communication complexities. In order to overcome these issues, we have proposed a privacy preserving method that the protection of gnomic data during their processioning and storage on the cloud, and without increasing computation and communication time compared to non-secure version [311]. Our solution uses Pretty Good Privacy for securing communications and cryptographic hash functions for securing the confidentiality of sensitive genetic data such as weighted-sum statistic input tables. Our method achieves the same performances and accuracy as its nonsecure version. As a consequence, contrarily to actual state of the art, our solution can be used in real world environments. This solution has given rise to a genetic platform that is being put in place in order to allow the scientific community to securely perform genome-wide association studies.

Above method ensures the confidentiality and privacy of outsourced data, but in some cases, these data are illegally disclosed. This is why we have proposed robust watermarking solution that ensures the copyright protection as well as traitor tracing. It allows identifying the person or entity who is the origin of an illegal information disclosure. This method combines Quantization Index Modulation (QIM) for watermark embedding in the database, and majority vote during the extraction of the watermark. In addition, we have embedded the watermark in genetic data used in GWAS without compromising association tests that can be conducted on these data.

Even though all these methods provide some good contributions to genetic data security, there are still several open issues or problems that can be considered in the future.

- The method we have presented in [15] has many limitations due to the use of fully homomorphic encryption. Several operations such as multiplicative data masking solution we have proposed requires a higher computation time. This is due to the use of binary representation of encrypted values. Possible areas of improvement for these operations and for the overall solution can the use of parallelization computation that can allows the com-

putations of many masking exchanges at the same time. Another idea can be the use of a partially homomorphic encryption for multiplicative data masking as we have a limited number of multiplication operations. In addition, in the implementation of our solution, we were limited to the implementation the bootstrapping pre-implemented in HElib. This point can be improved by using recent advances methods [263] that have significantly improved bootstrapping operations in terms of computation time.

- The dynamic watermarking method [14] must be improved in order to offer a better localization precision while reducing computation time. In fact, our database watermarking method localize illegal modification at subset level, and an improvement can be a localization precision at tuple level. Our method is used by cloud service providers for protecting integrity of encrypted databases. However, this method can be extended so as to allow watermarking on the user side where data would be protected in encrypted and clear forms at the same time. In addition, the embedded watermark is is accessible in encrypted domain. Thus, a generalization of our method in order to allow the watermark reading or extraction in clear form without impacting their use, would be a good improvement. In order to achieve dynamic watermarking, our solution is based on a journal table which is used for storing historical details about all added or suppressed tuples. However, this journal table comes with some issues such as storage complexity. Thus, a same dynamic watermarking without a journal table should be an important improvement.

- To continue our work in improving the computation complexity of previous methods, we are working on a solution that allows secure computation of GWAS based on a combination of different security mechanisms that are symmetric encryption, homomorphic encryption, watermarking and Intel Software Guard Extensions (SGX).

- Contrary to the state of the art based on homomorphic encryption, the proposed method for privacy-preserving genome-wide association studies for rare variants [311] achieves better performances. As a consequence, it can be used in real world environments. Moreover, this method can be extended to methods that are similar to WSS. However, some security can be considered in order to improve the security of our framework, due to the fact that we assume that all entities cannot collude with the Server. Thus, future works should focus on adapting our solution by considering that all parties in the framework can collude.

- The robust database watermarking method that we have developed for GWAS data can be improved by conducting its combination with [311]. This can help at protecting GWAS data on each side. Another point that can be studied is a theoretical analysis of the method in terms of distortion. As the method is developed for GWAS data, more tests for more association tests can be conducted so as to validate its accuracy for each GWAS method.

- We have proposed methods for protecting confidentiality of outsourced genetic data and their integrity in encrypted form. However, integrity of genetic data on user side should be also considered. In other words, future work should consists in developing a fragile watermarking method that can allow integrity control of genetic data in clear form. This method can for instance be an adaptation of existing database watermarking methods on genetic

data with the constraint that the inserted watermark does not interfere the interpretation of results for association tests conducted on watermarked data.

Bibliography

[1] Pixabay.com. *The structure of human DNA*, 2013.

[2] Opanclipart.org. *The genetic code*, 2010.

[3] Annalisa Lonetti, Maria Chiara Fontana, Giovanni Martinelli, and Ilaria Iacobucci. Single nucleotide polymorphisms as genomic markers for high-throughput pharmacogenomic studies. In *Microarray Technology*, pages 143–159. Springer, 2016.

[4] Veronica Aedo Lopez, Athina Stravodimou, Sheila Unger, Lucien Perey, and Khalil Zaman. Mutations de brca1/2: d'angelina jolie à la thérapie. *Rev Med Suisse*, pages 973–4, 2016.

[5] Geoffrey Ginsburg. Medical genomics: Gather and use genetic data in health care. *Nature News*, 508(7497):451, 2014.

[6] Maggie Haitian Wang, Heather J Cordell, and Kristel Van Steen. Statistical methods for genome-wide association studies. In *Seminars in cancer biology*, volume 55, pages 53–60. Elsevier, 2019.

[7] Kerem Ayoz, Erman Ayday, and A Ercument Cicek. Genome reconstruction attacks against genomic data-sharing beacons. *arXiv preprint arXiv:2001.08852*, 2020.

[8] Zaobo He, Jiguo Yu, Ji Li, Qilong Han, Guangchun Luo, and Yingshu Li. Inference attacks and controls on genotypes and phenotypes for individual genomic data. *IEEE/ACM transactions on computational biology and bioinformatics*, 2018.

[9] A Gutmann, J Wagner, Y Ali, AL Allen, JD Arras, BF Atkinson, N Farahany, A Garza, C Grady, S Hauser, et al. Privacy and progress in whole genome sequencing. *Presidential Committee for the Study of Bioethical*, (2012), 2012.

[10] Commission nationale de l'informatique et des libertés (CNIL). *Les données génétiques*. Collection CNIL, September 2017.

[11] Dalel Bouslimi, Gouenou Coatrieux, Michel Cozic, and Christian Roux. Data hiding in encrypted images based on predefined watermark embedding before encryption process. *Signal Processing: Image Communication*, 47:263–270, 2016.

[12] Safwat Hamad, Ahmed Elhadad, and Amal Khalifa. Dna watermarking using codon postfix technique. *IEEE/ACM transactions on computational biology and bioinformatics*, 15(5):1605–1610, 2017.

[13] Bingshan Li and Suzanne M Leal. Methods for detecting associations with rare variants for common diseases: application to analysis of sequence data. *The American Journal of Hum. Genetics*, 83(3):311–321, 2008.

[14] David Niyitegeka, Gouenou Coatrieux, Reda Bellafqira, Emmanuelle Genin, and Javier Franco-Contreras. Dynamic watermarking-based integrity protection of homomorphically encrypted databases–application to outsourced genetic data. In *International Workshop on Digital Watermarking*, pages 151–166. Springer, 2018.

[15] David Niyitegeka, Reda Bellafqira, Emmanuelle Genin, and Gouenou Coatrieux. Secure collapsing method based on fully homomorphic encryption. *Studies in health technology and informatics*, 270:412–416, 2020.

[16] Kristin Lauter, Adriana López-Alt, and Michael Naehrig. Private computation on encrypted genomic data. In *International Conference on Cryptology and Information Security in Latin America*, pages 3–27. Springer, 2014.

[17] Yuchen Zhang, Wenrui Dai, Xiaoqian Jiang, Hongkai Xiong, and Shuang Wang. Foresee: Fully outsourced secure genome study based on homomorphic encryption. In *BMC medical informatics and decision making*, volume 15, page S5. BioMed Central, 2015.

[18] Miran Kim and Kristin Lauter. Private genome analysis through homomorphic encryption. In *BMC medical informatics and decision making*, page S3. BioMed Central, 2015.

[19] Wen-Jie Lu, Yoshiji Yamada, and Jun Sakuma. Privacy-preserving genome-wide association studies on cloud environment using fully homomorphic encryption. In *BMC medical informatics and decision making*, page S1. BioMed Central, 2015.

[20] Charlotte Bonte, Eleftheria Makri, Amin Ardeshirdavani, Jaak Simm, Yves Moreau, and Frederik Vercauteren. Privacy-preserving genome-wide association study is practical. *IACR Cryptology ePrint Archive*, 2018:955, 2018.

[21] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124. ACM, 2011.

[22] Catherine Taylor Clelland, Viviana Risca, and Carter Bancroft. Hiding messages in dna microdots. *Nature*, 399(6736):533, 1999.

[23] Robert N Grass, Reinhard Heckel, Michela Puddu, Daniela Paunescu, and Wendelin J Stark. Robust chemical preservation of digital information on dna in silica with error-correcting codes. *Angewandte Chemie International Edition*, 54(8):2552–2555, 2015.

[24] Meinolf Blawat, Klaus Gaedke, Ingo Huetter, Xiao-Ming Chen, Brian Turczyk, Samuel Inverso, Benjamin W Pruitt, and George M Church. Forward error correction for dna data storage. *Procedia Computer Science*, 80:1011–1022, 2016.

[25] SM Hossein Tabatabaei Yazdi, Yongbo Yuan, Jian Ma, Huimin Zhao, and Olgica Milenkovic. A rewritable, random-access dna-based storage system. *Scientific reports*, 5:14138, 2015.

[26] HJ Shiu, Ka-Lok Ng, Jywe-Fei Fang, Richard CT Lee, and Chien-Hung Huang. Data hiding methods based upon dna sequences. *Information Sciences*, 180(11):2196–2208, 2010.

[27] Suk-Hwan Lee. Reversible data hiding for dna sequence using multilevel histogram shifting. *Security and Communication Networks*, 2018, 2018.

[28] Guoyan Liu, Hongjun Liu, and Abdurahman Kadir. Hiding message into dna sequence through dna coding and chaotic maps. *Medical & biological engineering & computing*, 52(9):741–747, 2014.

[29] Mohammad Saidur Rahman, Ibrahim Khalil, and Xun Yi. A lossless dna data hiding approach for data authenticity in mobile cloud based healthcare systems. *International Journal of Information Management*, 45:276–288, 2019.

[30] Juntao Fu, Weiming Zhang, Nenghai Yu, Guoli Ma, and Qi Tang. Fast tamper location of batch dna sequences based on reversible data hiding. In *2014 7th International Conference on Biomedical Engineering and Informatics*, pages 868–872. IEEE, 2014.

[31] Ghada Hamed, Mohammed Marey, Safaa El-Sayed Amin, and Mohamed Fahmy Tolba. Hybrid randomized and biological preserved dna-based crypt-steganography using generic n-bits binary coding rule. In *International Conference on Advanced Intelligent Systems and Informatics*, pages 618–627. Springer, 2016.

[32] Tianding Chen. A novel biology-based reversible data hiding fusion scheme. In *International Workshop on Frontiers in Algorithmics*, pages 84–95. Springer, 2007.

[33] Yanfeng Wang, Qinqin Han, Guangzhao Cui, and Junwei Sun. Hiding messages based on dna sequence and recombinant dna technique. *IEEE Transactions on Nanotechnology*, 18:299–307, 2019.

[34] David Haughton and Félix Balado. Biocode: Two biologically compatible algorithms for embedding data in non-coding and coding regions of dna. *BMC bioinformatics*, 14(1):121, 2013.

[35] Suah Kim, Xiaochao Qu, Vasily Sachnev, and Hyoung Joong Kim. Skewed histogram shifting for reversible data hiding using a pair of extreme predictions. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(11):3236–3246, 2018.

[36] Muhammad Naveed, Erman Ayday, Ellen W Clayton, Jacques Fellay, Carl A Gunter, Jean-Pierre Hubaux, Bradley A Malin, and XiaoFeng Wang. Privacy in the genomic era. *ACM Computing Surveys (CSUR)*, 48(1):1–44, 2015.

[37] Steven L Salzberg. Open questions: How many genes do we have? *BMC biology*, 16(1):94, 2018.

[38] Jian-Jun Shu. A new integrated symmetrical table for genetic codes. *Biosystems*, 151:21–26, 2017.

[39] ENCODE Project Consortium et al. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57, 2012.

[40] 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature*, 526(7571):68, 2015.

[41] Council of Europe. Committee of Ministers et al. *The Protection of Medical Data: Recommendation No. R (97) 5 Adopted by the Committee of Ministers of the Council of Europe on 13 February 1997 and Explanatory Memorandum*. Council of Europe Publ., 1997.

[42] HD Abbing. International declaration on human genetic data. *European journal of health law*, 11(1):93–107, 2004.

[43] Protection Regulation. Regulation (eu) 2016/679 of the european parliament and of the council. *REGULATION (EU)*, 679:2016, 2016.

[44] Joshua T Burdick, Wei-Min Chen, Gonçalo R Abecasis, and Vivian G Cheung. In silico method for inferring genotypes in pedigrees. *Nature genetics*, 38(9):1002–1004, 2006.

[45] Nicole Chavez and Sonya Hamasaki. He spent 14 years in prison for murder. now, he's the first person in california to be exonerated with the help of genetic genealogy. *CNN*, 2020.

[46] Patrícia Santana Correia, Pedro Vitiello, Maria Helena Cabral de Almeida Cardoso, and Dafne Dain Gandelman Horovitz. Conceptions on genetics in a group of college students. *Journal of community genetics*, 4(1):115–123, 2013.

[47] Nasim Mavaddat, Kyriaki Michailidou, Joe Dennis, Michael Lush, Laura Fachal, Andrew Lee, Jonathan P Tyrer, Ting-Huei Chen, Qin Wang, Manjeet K Bolla, et al. Polygenic risk scores for prediction of breast cancer and breast cancer subtypes. *The American Journal of Human Genetics*, 104(1):21–34, 2019.

[48] Petr Danecek, Adam Auton, Goncalo Abecasis, Cornelis A Albers, Eric Banks, Mark A DePristo, Robert E Handsaker, Gerton Lunter, Gabor T Marth, Stephen T Sherry, et al. The variant call format and vcftools. *Bioinformatics*, 27(15):2156–2158, 2011.

[49] Jay Shendure and Hanlee Ji. Next-generation dna sequencing. *Nature biotechnology*, 26(10):1135, 2008.

[50] Tiffany Amariuta, Yang Luo, Rachel Knevel, Yukinori Okada, and Soumya Raychaudhuri. Advances in genetics toward identifying pathogenic cell states of rheumatoid arthritis. *Immunological reviews*, 2019.

[51] J Vineela Krupanidhi Srirama, K Tejaswi, and MN Thanmayi. Huntington's chorea, a neurological disorder of all ages–bioinformatics approach for its precise diagnosis. *International journal of health sciences*, 13(6):26, 2019.

[52] Thomas N Williams and Swee Lay Thein. Sickle cell anemia and its phenotypes. *Annual review of genomics and human genetics*, 19:113–147, 2018.

[53] Katharina Weiss, Amelie Lotz-Havla, Katharina Dokoupil, and Esther M Maier. Management of three preterm infants with phenylketonuria. *Nutrition*, 71:110619, 2020.

[54] Holm Schneider, Florian Faschingbauer, Sonia Schuepbach-Mallepell, Iris Körber, Sigrun Wohlfart, Angela Dick, Mandy Wahlbuhl, Christine Kowalczyk-Quintas, Michele Vigolo, Neil Kirby, et al. Prenatal correction of x-linked hypohidrotic ectodermal dysplasia. *New England Journal of Medicine*, 378(17):1604–1610, 2018.

[55] Francis S Collins, Michael Morgan, and Aristides Patrinos. The human genome project: lessons from large-scale biology. *Science*, 300(5617):286–290, 2003.

[56] Nayanah Siva. 1000 genomes project, 2008.

[57] Julian G Barwell, Rory BG O'Sullivan, Laura K Mansbridge, Joanna M Lowry, and Huw R Dorkins. Challenges in implementing genomic medicine: the 100,000 genomes project. *J Transl Genet Genom*, 2(10.20517), 2018.

[58] Francis S Collins and Harold Varmus. A new initiative on precision medicine. *New England journal of medicine*, 372(9):793–795, 2015.

[59] Naomi E Allen, Cathie Sudlow, Tim Peakman, Rory Collins, et al. Uk biobank data: come and get it, 2014.

[60] Andreas Menke, Katharina Domschke, Darina Czamara, Torsten Klengel, Johannes Hennings, Susanne Lucae, Bernhard T Baune, Volker Arolt, Bertram Müller-Myhsok, Florian Holsboer, et al. Genome-wide association study of antidepressant treatment-emergent suicidal ideation. *Neuropsychopharmacology*, 37(3):797–807, 2012.

[61] Marc Fiume, Miroslav Cupak, Stephen Keenan, Jordi Rambla, Sabela de la Torre, Stephanie OM Dyke, Anthony J Brookes, Knox Carey, David Lloyd, Peter Goodhand, et al. Federated discovery and sharing of genomic data using beacons. *Nature biotechnology*, 37(3):220–224, 2019.

[62] Brett S Abrahams, Dan E Arking, Daniel B Campbell, Heather C Mefford, Eric M Morrow, Lauren A Weiss, Idan Menashe, Tim Wadkins, Sharmila Banerjee-Basu, and Alan Packer. Sfari gene 2.0: a community-driven knowledgebase for the autism spectrum disorders (asds). *Molecular autism*, 4(1):36, 2013.

[63] Michael D Edge and Graham Coop. Attacks on genetic privacy via uploads to genealogical databases. *eLife*, 9, 2020.

[64] Pascal Su. Direct-to-consumer genetic testing: a comprehensive view. *The Yale journal of biology and medicine*, 86(3):359, 2013.

[65] MA Allyse, DH Robinson, MJ Ferber, and RR Sharp. Direct-to-consumer testing 2.0: Emerging models of direct-to-consumer genetic testing. In *Mayo Clinic proceedings*, volume 93, page 113, 2018.

[66] QS Li, C Tian, GR Seabrook, WC Drevets, and VA Narayan. Analysis of 23andme antidepressant efficacy survey data: implication of circadian rhythm and neuroplasticity in bupropion response. *Translational psychiatry*, 6(9):e889–e889, 2016.

[67] Alec J Jeffreys, Victoria Wilson, and Swee Lay Thein. Individual-specific 'fingerprints' of human dna. *Nature*, 316(6023):76–79, 1985.

[68] John M Butler. Genetics and genomics of core short tandem repeat loci used in human identity testing. *Journal of forensic sciences*, 51(2):253–265, 2006.

[69] Elizabeth E Joh. Reclaiming abandoned dna: the fourth amendment and genetic privacy. *Nw. UL Rev.*, 100:857, 2006.

[70] David H Kaye and Michael E Smith. Dna identification databases: legality, legitimacy, and the case for population-wide coverage. *Wis. L. Rev.*, page 413, 2003.

[71] L Dusserre, H Ducrot, and FA Allaert. L'information médicale. *L'ordinateur et la loi. Deuxième edition Editions Médicales Internationales*, 1999.

[72] Ganthan Narayana Samy, Rabiah Ahmad, and Zuraini Ismail. Security threats categories in healthcare information systems. *Health informatics journal*, 16(3):201–209, 2010.

[73] Latanya Sweeney, Akua Abu, and Julia Winn. Identifying participants in the personal genome project by name (a re-identification experiment). *arXiv preprint arXiv:1304.7605*, 2013.

[74] Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. Identifying personal genomes by surname inference. *Science*, 339(6117):321–324, 2013.

[75] Turi E King and Mark A Jobling. What's in a name? y chromosomes, surnames and the genetic genealogy revolution. *Trends in genetics*, 25(8):351–360, 2009.

[76] Suyash S Shringarpure and Carlos D Bustamante. Privacy risks from genomic data-sharing beacons. *The American Journal of Human Genetics*, 97(5):631–646, 2015.

[77] Shuang Wang, Xiaoqian Jiang, Haixu Tang, Xiaofeng Wang, Diyue Bu, Knox Carey, Stephanie OM Dyke, Dov Fox, Chao Jiang, Kristin Lauter, et al. A community effort to protect genomic data sharing, collaboration and outsourcing. *NPJ genomic medicine*, 2(1):1–6, 2017.

[78] Andrew J Pakstis, William C Speed, Rixun Fang, Fiona CL Hyland, Manohar R Furtado, Judith R Kidd, and Kenneth K Kidd. Snps for a universal individual identification panel. *Human genetics*, 127(3):315–324, 2010.

[79] Eric E Schadt, Sangsoon Woo, and Ke Hao. Bayesian method to predict individual snp genotypes from gene expression data. *Nature genetics*, 44(5):603–608, 2012.

[80] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, 4(8):e1000167, 2008.

[81] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pages 17–32, 2014.

[82] Jean Louis Raisaro, Florian Tramer, Zhanglong Ji, Diyue Bu, Yongan Zhao, Knox Carey, David Lloyd, Heidi Sofia, Dixie Baker, Paul Flicek, et al. Addressing beacon re-identification attacks: quantification and mitigation of privacy risks. *Journal of the American Medical Informatics Association*, 24(4):799–805, 2017.

[83] Arif Harmanci and Mark Gerstein. Quantification of private information leakage from phenotype-genotype data: linking attacks. *Nature methods*, 13(3):251–256, 2016.

[84] Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 534–544. ACM, 2009.

[85] Accountability Act. Health insurance portability and accountability act of 1996. *Public law*, 104:191, 1996.

[86] Louise Slaughter. Genetic information non-discrimination act. *Harv. J. on Legis.*, 50:41, 2013.

[87] R Guy Thomas. Genetics and insurance in the united kingdom 1995–2010: the rise and fall of "scientific" discrimination. *New Genetics and Society*, 31(2):203–222, 2012.

[88] Rene Saint-Germain et al. Information security management best practice based on iso/iec 17799. *Information management journal*, 39(4):60–66, 2005.

[89] ISO ISO. 27799-health informatics-information security management in health using iso.

[90] IHE IT Infrastructure Technical Committee et al. Ihe it infrastructure (iti) technical framework: integration profiles. integrating the healthcare enterprise, 2016.

[91] Dalel Bouslimi, Gouenou Coatrieux, Michel Cozic, and Ch Roux. Combination of watermarking and joint watermarking-decryption for reliability control and traceability of medical images. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4495–4498. IEEE, 2014.

[92] Suzy A Buckovich, Helga E Rippen, and Michael J Rozen. Driving toward guiding principles: a goal for privacy, confidentiality, and security of health information. *Journal of the American Medical Informatics Association*, 6(2):122–133, 1999.

[93] Yann Joly, Ida Ngueng Feze, Lingqiao Song, and Bartha M Knoppers. Comparative approaches to genetic discrimination: chasing shadows? *Trends in Genetics*, 33(5):299–302, 2017.

[94] Ellen Wright Clayton, Barbara J Evans, James W Hazel, and Mark A Rothstein. The law of genetic privacy: applications, implications, and limitations. *Journal of Law and the Biosciences*, 6(1):1–36, 2019.

[95] Mark A Rothstein. *Genetic secrets: protecting privacy and confidentiality in the genetic era*. Yale University Press, 1997.

[96] Samuel J Aronson and Heidi L Rehm. Building the foundation for genomics in precision medicine. *Nature*, 526(7573):336–342, 2015.

[97] Jongsik Chun, Aharon Oren, Antonio Ventosa, Henrik Christensen, David Ruiz Arahal, Milton S da Costa, Alejandro P Rooney, Hana Yi, Xue-Wei Xu, Sofie De Meyer, et al. Proposed minimal standards for the use of genome data for the taxonomy of prokaryotes. *International journal of systematic and evolutionary microbiology*, 68(1):461–466, 2018.

[98] EBIOS ANSSI. Ebios-expression des besoins et identification des objectifs de sécurité, 2016.

[99] Christopher J Alberts, Sandra G Behrens, Richard D Pethia, and William R Wilson. Operationally critical threat, asset, and vulnerability evaluation (octave) framework, version 1.0. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1999.

[100] CLUSIF. Mehari 2010 : Guide de l'analyse et du traitement des risques, 2010.

[101] Niranchana Radhakrishnan and Marimuthu Karuppiah. An efficient and secure remote user mutual authentication scheme using smart cards for telecare medical information systems. *Informatics in Medicine Unlocked*, 16:100092, 2019.

[102] Wei Pan, Gouenou Coatrieux, Dalel Bouslimi, and Nicolas Prigent. Secure public cloud platform for medical images sharing. In *MIE*, pages 251–255, 2015.

[103] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (CSUR)*, 51(4):79, 2018.

[104] David W Archer, Dan Bogdanov, Yehuda Lindell, Liina Kamm, Kurt Nielsen, Jakob Illeborg Pagter, Nigel P Smart, and Rebecca N Wright. From keys to databases—real-world applications of secure multi-party computation. *The Computer Journal*, 61(12):1749–1771, 2018.

[105] Saif Al-Kuwari, James H Davenport, and Russell J Bradford. Cryptographic hash functions: recent design trends and security notions. *IACR Cryptology ePrint Archive*, 2011:565, 2011.

[106] Javier Franco-Contreras and Gouenou Coatrieux. Robust watermarking of relational databases with ontology-guided distortion control. *IEEE transactions on information forensics and security*, 10(9):1939–1952, 2015.

[107] EMC INC. Rsa securid, 2018.

[108] Afnan Binduf, Hanan Othman Alamoudi, Hanan Balahmar, Shatha Alshamrani, Haifa Al-Omar, and Naya Nagy. Active directory and related aspects of security. In *2018 21st Saudi Computer Society National Computer Conference (NCC)*, pages 4474–4479. IEEE, 2018.

[109] Thomas Matthew McCann and Kedar Kashinath Karmarkar. Methods, systems, and computer readable media for remote authentication dial in user service (radius) message loop detection and mitigation, March 20 2018. US Patent 9,923,984.

[110] Wei Pan. *Protection des images médicales: tatouage réversible pour le contrôle d'accès et d'usage*. PhD thesis, 2012.

[111] Butler W Lampson. Protection. *ACM SIGOPS Operating Systems Review*, 8(1):18–24, 1974.

[112] Ravi S Sandhu, Edward J Coyne, Hal L Feinstein, and Charles E Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[113] Anas Abou El Kalam, R El Baida, Philippe Balbiani, Salem Benferhat, Frédéric Cuppens, Yves Deswarte, Alexandre Miege, Claire Saurel, and Gilles Trouessin. Organization based access control. In *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, pages 120–131. IEEE, 2003.

[114] Vincent C Hu, D Richard Kuhn, David F Ferraiolo, and Jeffrey Voas. Attribute-based access control. *Computer*, 48(2):85–88, 2015.

[115] Yuanyu Zhang, Shoji Kasahara, Yulong Shen, Xiaohong Jiang, and Jianxiong Wan. Smart contract-based access control for the internet of things. *IEEE Internet of Things Journal*, 6(2):1594–1605, 2018.

[116] Artem Voronkov, Leonardo A Martucci, and Stefan Lindskog. Measuring the usability of firewall rule sets. *IEEE Access*, 8:27106–27121, 2020.

[117] Robert Zalenski. Firewall technologies. *IEEE potentials*, 21(1):24–29, 2002.

[118] William Stallings, Lawrie Brown, Michael D Bauer, and Arup Kumar Bhattacharjee. *Computer security: principles and practice*. Pearson Education Upper Saddle River, NJ, USA, 2012.

[119] Jianguo Ren, Xiaofan Yang, Qingyi Zhu, Lu-Xing Yang, and Chunming Zhang. A novel computer virus model and its dynamics. *Nonlinear Analysis: Real World Applications*, 13(1):376–384, 2012.

[120] Poonam Jindal and Brahmjit Singh. Rc4 encryption-a literature survey. *Procedia Computer Science*, 46:697–705, 2015.

[121] Philipp Jovanovic. Analysis and design of symmetric cryptographic algorithms. 2015.

[122] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009.

[123] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.

[124] D Mukhopadhyay. Cryptography: Advanced encryption standard (aes). *Encyclopedia of Computer Science and Technology*, 279, 2017.

[125] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[126] Reda Bellafqira, Gouenou Coatrieux, Dalel Bouslimi, Gwénolé Quellec, and Michel Cozic. Proxy re-encryption based on homomorphic encryption. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, pages 154–161. ACM, 2017.

[127] Ayantika Chatterjee and Khin Mi Mi Aung. *Fully Homomorphic Encryption in Real World Applications*. Springer, 2019.

[128] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pages 173–201. 2019.

[129] Andrew C Yao. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (sfcs 1982)*, pages 160–164. IEEE, 1982.

[130] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, July 1985.

[131] Josh Benaloh. Dense probabilistic encryption. In *Proceedings of the workshop on selected areas of cryptography*, pages 120–128, 1994.

[132] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM conference on Computer and communications security*, pages 59–66, 1998.

[133] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *International conference on the theory and applications of cryptographic techniques*, pages 308–318. Springer, 1998.

[134] Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for nc/sup 1. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 554–566. IEEE, 1999.

[135] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 223–238, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[136] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography Conference*, pages 325–341. Springer, 2005.

[137] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Multi-bit cryptosystems based on lattice problems. In *International Workshop on Public Key Cryptography*, pages 315–329. Springer, 2007.

[138] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In *Theory of Cryptography Conference*, pages 575–594. Springer, 2007.

[139] Ivan Damgård and Mads Jurik. A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system. In *International workshop on public key cryptography*, pages 119–136. Springer, 2001.

[140] Ivan Damgård, Mads Jurik, and Jesper Buus Nielsen. A generalization of paillier's public-key system with applications to electronic voting. *International Journal of Information Security*, 9(6):371–385, Dec 2010.

[141] Miaomiao Zhang and Steven Romero. Design and implementation of an e-voting system based on paillier encryption. In *Future of Information and Communication Conference*, pages 815–831. Springer, 2020.

[142] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 364–373. IEEE, 1997.

[143] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[144] Ronald L Rivest, Len Adleman, Michael L Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[145] Burt Kaliski. *Quadratic Residuosity Problem*, pages 493–493. Springer US, Boston, MA, 2005.

[146] Henry Corrigan-Gibbs and Dmitry Kogan. The discrete-logarithm problem with preprocessing. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 415–447. Springer, 2018.

[147] Tibor Jager. The generic composite residuosity problem. In *Black-Box Models of Computation in Cryptology*, pages 49–56. Springer, 2012.

[148] Michael Fellows and Neal Koblitz. Combinatorial cryptosystems galore! *Contemporary Mathematics*, 168:51–51, 1994.

[149] Kristian Gjøsteen. Subgroup membership problems and public key cryptosystems. 2004.

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

[151] Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with d-operand multiplications. In *Annual Cryptology Conference*, pages 138–154. Springer, 2010.

[152] Frederik Armknecht and Ahmad-Reza Sadeghi. A new approach for algebraically homomorphic encryption. *IACR Cryptology ePrint Archive*, 2008:422, 2008.

[153] Craig Gentry and Dan Boneh. *A fully homomorphic encryption scheme*, volume 20. Stanford university Stanford, 2009.

[154] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13, 2014.

[155] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.

[156] Craig Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *Annual Cryptology Conference*, pages 116–137. Springer, 2010.

[157] Peter Scholl and Nigel P Smart. Improved key generation for gentry's fully homomorphic encryption scheme. In *IMA International Conference on Cryptography and Coding*, pages 10–22. Springer, 2011.

[158] Naoki Ogura, Go Yamamoto, Tetsutaro Kobayashi, and Shigenori Uchiyama. An improvement of key generation algorithm for gentry's homomorphic encryption scheme. In *International Workshop on Security*, pages 70–83. Springer, 2010.

[159] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 24–43. Springer, 2010.

[160] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Fully homomorphic encryption over the integers with shorter public keys. In *Annual Cryptology Conference*, pages 487–504. Springer, 2011.

[161] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Scale-invariant fully homomorphic encryption over the integers. In *International Workshop on Public Key Cryptography*, pages 311–328. Springer, 2014.

[162] Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrede Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 315–335. Springer, 2013.

[163] Koji Nuida and Kaoru Kurosawa. (batch) fully homomorphic encryption over integers for non-binary message spaces. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 537–555. Springer, 2015.

[164] Steven D Galbraith, Shishay W Gebregiyorgis, and Sean Murphy. Algorithms for the approximate common divisor problem. *LMS Journal of Computation and Mathematics*, 19(A):58–72, 2016.

[165] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Annual cryptology conference*, pages 505–524. Springer, 2011.

[166] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[167] Shai Halevi and Victor Shoup. Algorithms in helib. In *Annual Cryptology Conference*, pages 554–571. Springer, 2014.

[168] Michael O Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptology ePrint Archive*, 2005:187, 2005.

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

[170] Sebati Ghosh and Palash Sarkar. Variants of wegman-carter message authentication code supporting variable tag lengths. 2020.

[171] FRANCIS MORGAN Boland, Joseph JK O'Ruanaidh, and C Dautzenberg. Watermarking digital images for copyright protection. 1995.

[172] Hector Santoyo-Garcia, Eduardo Fragoso-Navarro, Rogelio Reyes-Reyes, Clara Cruz-Ramos, and Mariko Nakano-Miyatake. Visible watermarking technique based on human visual system for single sensor digital cameras. *Security and Communication Networks*, 2017, 2017.

[173] Fred Mintzer, Jeffrey Lotspiech, Norishige Morimoto, and T Almaden. Safeguarding digital library contents and users. *D-lib magazine*, 3(7/8), 1997.

[174] Gouenou Coatrieux, Emmanuel Chazard, Régis Beuscart, and Christian Roux. Lossless watermarking of categorical attributes for verifying medical data base integrity. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 8195–8198. IEEE, 2011.

[175] Dominik Heider and Angelika Barnekow. Dna-based watermarks using the dna-crypt algorithm. *BMC bioinformatics*, 8(1):176, 2007.

[176] Rajendra Acharya, UC Niranjan, S Sitharama Iyengar, N Kannathal, and Lim Choo Min. Simultaneous storage of patient information with medical images in the frequency domain. *Computer methods and programs in biomedicine*, 76(1):13–19, 2004.

[177] Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.

[178] Adnan M Alattar. Smart images using digimarc's watermarking technology. In *Security and Watermarking of Multimedia Contents II*, volume 3971, pages 264–273. International Society for Optics and Photonics, 2000.

[179] Rakesh Agrawal and Jerry Kiernan. Watermarking relational databases. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 155–166. Elsevier, 2002.

[180] Neal R Wagner. Fingerprinting. In *1983 IEEE Symposium on Security and Privacy*, pages 18–18. IEEE, 1983.

[181] Yingjiu Li, Vipin Swarup, and Sushil Jajodia. Fingerprinting relational databases: Schemes and specialties. *IEEE Transactions on Dependable and Secure Computing*, 2(1):34–45, 2005.

[182] Wade Trappe, Min Wu, Z Jane Wang, and KJ Ray Liu. Anti-collusion fingerprinting for multimedia. *IEEE Transactions on Signal Processing*, 51(4):1069–1087, 2003.

[183] Ana Charpentier, Caroline Fontaine, Teddy Furon, and Ingemar Cox. An asymmetric fingerprinting scheme based on tardos codes. In *International Workshop on Information Hiding*, pages 43–58. Springer, 2011.

[184] Saman Iftikhar, Sharifullah Khan, Zahid Anwar, and Muhammad Kamran. Genin-foguard—a robust and distortion-free watermarking technique for genetic data. *PloS one*, 10(2), 2015.

[185] Ibrahim Kamel and Kareem Kamel. Toward protecting the integrity of relational databases. In *2011 World Congress on Internet Security (WorldCIS-2011)*, pages 258–261. IEEE, 2011.

[186] Jie Guo. Fragile watermarking scheme for tamper detection of relational database. In *2011 International Conference on Computer and Management (CAMAN)*, pages 1–4. IEEE, 2011.

[187] Farnaz Arab, Mazdak Zamani, Sofya Poger, Carol Manigault, and Songmei Yu. A framework to evaluate the performance of video watermarking techniques. In *2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*, pages 114–117. IEEE, 2019.

[188] Javier Franco Contreras. *Watermarking services for medical database content security*. PhD thesis, 2014.

[189] J. Franco-Contreras, G. Coatrieux, F. Cuppens, N. Cuppens-Boulahia, and C. Roux. Robust lossless watermarking of relational databases based on circular histogram modulation. *IEEE Transactions on Information Forensics and Security*, 9(3):397–410, March 2014.

[190] Donghui Hu, Dan Zhao, and Shuli Zheng. A new robust approach for reversible database watermarking with distortion control. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1024–1037, 2018.

[191] Muhammad Kamran and Muddassar Farooq. An optimized information-preserving relational database watermarking scheme for ownership protection of medical data. *arXiv preprint arXiv:1801.09741*, 2018.

[192] Zhi-hao Zhang, Xiao-Ming Jin, Jian-Min Wang, and De-Yi Li. Watermarking relational database using image. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*, volume 3, pages 1739–1744. IEEE, 2004.

[193] Fabian M Suchanek and David Gross-Amblard. Adding fake facts to ontologies. In *Proceedings of the 21st International Conference on World Wide Web*, pages 421–424. ACM, 2012.

[194] Damien Hanyurwimfura, Yuling Liu, and Zhijie Liu. Text format based relational database watermarking for non-numeric data. In *2010 International Conference On Computer Design and Applications*, volume 4, pages V4–312. IEEE, 2010.

[195] Gaurav Gupta and Josef Pieprzyk. Database relation watermarking resilient against secondary watermarking attacks. In *International Conference on Information Systems Security*, pages 222–236. Springer, 2009.

[196] Yingjiu Li, Huiping Guo, and Sushil Jajodia. Tamper detection and localization for categorical data using fragile watermarks. In *Proceedings of the 4th ACM workshop on Digital rights management*, pages 73–82. ACM, 2004.

[197] Huiping Guo, Yingjiu Li, Anyi Liu, and Sushil Jajodia. A fragile watermarking scheme for detecting malicious modifications of database relations. *Information Sciences*, 176(10):1350–1378, 2006.

[198] Vahab Prasannakumari. A robust tamperproof watermarking for data integrity in relational databases. *Research Journal of Information Technology*, 1(3):115–121, 2009.

[199] Sukriti Bhattacharya and Agostino Cortesi. A distortion free watermark framework for relational databases. In *ICSOFT (2)*, pages 229–234, 2009.

[200] Jung-Nan Chang and Hsien-Chu Wu. Reversible fragile database watermarking technology using difference expansion based on svr prediction. In *2012 International Symposium on Computer, Consumer and Control*, pages 690–693. IEEE, 2012.

[201] Waheeb Yaqub, Ibrahim Kamel, and Zeyar Aung. Toward watermarking compressed data in columnar database architectures. *Security and Privacy*, page e84, 2019.

[202] Aihab Khan and Syed Afaq Husain. A fragile zero watermarking scheme to detect and characterize malicious modifications in database relations. *The Scientific World Journal*, 2013, 2013.

[203] Vidhi Khanduja and Shampa Chakraverty. A generic watermarking model for object relational databases. *Multimedia Tools and Applications*, pages 1–25, 2019.

[204] Meng-Hsiun Tsai, Hsiao-Yun Tseng, and Chen-Ying Lai. A database watermarking technique for temper detection. In *9th Joint International Conference on Information Sciences (JCIS-06)*. Atlantis Press, 2006.

[205] Farah Naz, Abid Khan, Mansoor Ahmed, Majid Iqbal Khan, Sadia Din, Awais Ahmad, and Gwanggil Jeon. Watermarking as a service (waas) with anonymity. *Multimedia Tools and Applications*, pages 1–25, 2019.

[206] Shabir A Parah, Farhana Ahad, Javaid A Sheikh, and Ghulam Mohiuddin Bhat. Hiding clinical information in medical images: a new high capacity and reversible data hiding technique. *Journal of biomedical informatics*, 66:214–230, 2017.

[207] Shijun Xiang and Jiayong He. Database authentication watermarking scheme in encrypted domain. *IET Information Security*, 12(1):42–51, 2018.

[208] David Niyitegeka, Gouenou Coatrieux, Reda Bellafqira, Emmanuelle Genin, and Javier Franco-Contreras. Dynamic watermarking-based integrity protection of homomorphically encrypted databases – application to outsourced genetic data. In Chang D. Yoo, Yun-Qing Shi, Hyoung Joong Kim, Alessandro Piva, and Gwangsu Kim, editors, *Digital Forensics and Watermarking*, pages 151–166, Cham, 2019. Springer International Publishing.

[209] Kee Sung Kim, Minkyu Kim, Dongsoo Lee, Je Hong Park, and Woo-Hwan Kim. Security of stateful order-preserving encryption. In *International Conference on Information Security and Cryptology*, pages 39–56. Springer, 2018.

[210] Liangliang Xiao and I-Ling Yen. Security analysis for order preserving encryption schemes. In *2012 46th annual conference on information sciences and systems (CISS)*, pages 1–6. IEEE, 2012.

[211] Konrad Karczewski and L Francioli. The genome aggregation database (gnomad). *MacArthur Lab*, 2017.

[212] Murat Kantarcioglu, Wei Jiang, Ying Liu, and Bradley Malin. A cryptographic approach to securely share and query genomic sequences. *IEEE Transactions on information technology in biomedicine*, 12(5):606–617, 2008.

[213] Mustafa Canim, Murat Kantarcioglu, and Bradley Malin. Secure management of biomedical data with cryptographic hardware. *IEEE Transactions on Information Technology in Biomedicine*, 16(1):166–175, 2011.

[214] Reza Ghasemi, Md Momin Al Aziz, Noman Mohammed, Massoud Hadian Dehkordi, and Xiaoqian Jiang. Private and efficient query processing on outsourced genomic databases. *IEEE journal of biomedical and health informatics*, 21(5):1466–1472, 2016.

[215] Mohamed Nassar, Qutaibah Malluhi, Mikhail Atallah, and Abdullatif Shikfa. Securing aggregate queries for dna databases. *IEEE Transactions on Cloud Computing*, 2017.

[216] Mohammad Zahidul Hasan, Md Safiur Rahman Mahdi, Md Nazmus Sadat, and Noman Mohammed. Secure count query on encrypted genomic data. *Journal of biomedical informatics*, 81:41–52, 2018.

[217] Mikhail J Atallah and Jiangtao Li. Secure outsourcing of sequence comparisons. *International Journal of Information Security*, 4(4):277–287, 2005.

[218] Somesh Jha, Louis Kruger, and Vitaly Shmatikov. Towards practical privacy for genomic computation. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 216–230. IEEE, 2008.

[219] Gilad Asharov, Shai Halevi, Yehuda Lindell, and Tal Rabin. Privacy-preserving search of similar patients in genomic data. *Proceedings on Privacy Enhancing Technologies*, 2018(4):104–124, 2018.

[220] João Sá Sousa, Cédric Lefebvre, Zhicong Huang, Jean Louis Raisaro, Carlos Aguilar-Melchor, Marc-Olivier Killijian, and Jean-Pierre Hubaux. Efficient and secure outsourcing of genomic data storage. *BMC medical genomics*, 10(2):46, 2017.

[221] Xiaofei Wang and Yuqing Zhang. E-sc: collusion-resistant secure outsourcing of sequence comparison algorithm. *IEEE Access*, 6:3358–3375, 2017.

[222] Bing Wang, Wei Song, Wenjing Lou, and Y Thomas Hou. Privacy-preserving pattern matching over encrypted genetic data in cloud computing. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[223] Pierre Baldi, Roberta Baronio, Emiliano De Cristofaro, Paolo Gasti, and Gene Tsudik. Countering gattaca: Efficient and secure testing of fully-sequenced human genomes (full version). *arXiv preprint arXiv:1110.2478*, 2011.

[224] J Raphael Gibbs and Andrew Singleton. Application of genome-wide single nucleotide polymorphism typing: simple association and beyond. *PLoS genetics*, 2(10), 2006.

[225] Fons Bruekers, Stefan Katzenbeisser, Klaus Kursawe, and Pim Tuyls. Privacy-preserving matching of dna profiles. *IACR Cryptology ePrint Archive*, 2008:203, 2008.

[226] Marina Blanton and Fattaneh Bayatbabolghani. Improving the security and efficiency of private genomic computation using server aid. *IEEE Security & Privacy*, 15(5):20–28, 2017.

[227] Karthik A Jagadeesh, David J Wu, Johannes A Birgmeier, Dan Boneh, and Gill Bejerano. Deriving genomic diagnoses without revealing patient genomes. *Science*, 357(6352):692–695, 2017.

[228] Juan Ramón Troncoso-Pastoriza, Stefan Katzenbeisser, and Mehmet Celik. Privacy preserving error resilient dna searching through oblivious automata. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 519–528, 2007.

[229] Paul J McLaren, Jean Louis Raisaro, Manel Aouri, Margalida Rotger, Erman Ayday, István Bartha, Maria B Delgado, Yannick Vallet, Huldrych F Günthard, Matthias Cavassini, et al. Privacy-preserving genomic testing in the clinic: a model using hiv treatment. *Genetics in medicine*, 18(8):814–822, 2016.

[230] Feng Chen, Chenghong Wang, Wenrui Dai, Xiaoqian Jiang, Noman Mohammed, Md Momin Al Aziz, Md Nazmus Sadat, Cenk Sahinalp, Kristin Lauter, and Shuang Wang. Presage: privacy-preserving genetic testing via software guard extension. *BMC medical genomics*, 10(2):48, 2017.

[231] Michael Schwarz, Samuel Weiser, and Daniel Gruss. Practical enclave malware with intel sgx. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 177–196. Springer, 2019.

[232] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[233] Florian Tramèr, Zhicong Huang, Jean-Pierre Hubaux, and Erman Ayday. Differential privacy with bounded priors: reconciling utility and privacy in genome-wide association studies. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1286–1297. ACM, 2015.

[234] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[235] Bradley Malin and Latanya Sweeney. How (not) to protect genomic data privacy in a distributed network: using trail re-identification to evaluate and design anonymity protection systems. *Journal of biomedical informatics*, 37(3):179–192, 2004.

[236] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *null*, page 24. IEEE, 2006.

[237] Yaniv Erlich and Arvind Narayanan. Routes for breaching and protecting genetic privacy. *Nature Reviews Genetics*, 15(6):409, 2014.

[238] Aaron Johnson and Vitaly Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1079–1087, New York, NY, USA, 2013. ACM.

[239] Caroline Uhlerop, Aleksandra Slavković, and Stephen E Fienberg. Privacy-preserving data sharing for genome-wide association studies. *The Journal of privacy and confidentiality*, 5(1):137, 2013.

[240] Sean Simmons, Cenk Sahinalp, and Bonnie Berger. Enabling privacy-preserving gwass in heterogeneous human populations. *Cell systems*, 3(1):54–61, 2016.

[241] Daniel Kifer and Ryan Rogers. A new class of private chi-square tests. *arXiv preprint arXiv:1610.07662*, 2016.

[242] Yuichi Sei and Akihiko Ohsuga. Privacy-preserving chi-squared testing for genome snp databases. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3884–3889. IEEE, 2017.

[243] Liina Kamm, Dan Bogdanov, Sven Laur, and Jaak Vilo. A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics*, 29(7):886–893, 2013.

[244] Scott D Constable, Yuzhe Tang, Shuang Wang, Xiaoqian Jiang, and Steve Chapin. Privacy-preserving gwas analysis on federated genomic datasets. In *BMC medical informatics and decision making*, volume 15, page S2. BioMed Central, 2015.

[245] Yihua Zhang, Marina Blanton, and Ghada Almashaqbeh. Secure distributed genome analysis for gwas and sequence comparison computation. In *BMC medical informatics and decision making*, volume 15, page S4. BioMed Central, 2015.

[246] Hyunghoon Cho, David J Wu, and Bonnie Berger. Secure genome-wide association analysis using multiparty computation. *Nature biotechnology*, 36(6):547, 2018.

[247] Jonathan M Bloom. Secure multi-party linear regression at plaintext speed. *arXiv preprint arXiv:1901.09531*, 2019.

[248] Md Nazmus Sadat, Al Aziz, Md Momin, Noman Mohammed, Feng Chen, Xiaoqian Jiang, and Shuang Wang. Safety: Secure gwas in federated environment through a hybrid solution. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 16(1):93–102, 2019.

[249] Ferdinand Brasser, Urs Müller, Alexandra Dmitrienko, Kari Kostiainen, Srdjan Capkun, and Ahmad-Reza Sadeghi. Software grand exposure:{SGX} cache attacks are practical. In *11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17)*, 2017.

[250] Marcus Hähnel, Weidong Cui, and Marcus Peinado. High-resolution side channels for untrusted operating systems. In *2017 {USENIX} Annual Technical Conference ({USENIX} {ATC} 17)*, pages 299–312, 2017.

[251] Shuang Wang, Yuchen Zhang, Wenrui Dai, Kristin Lauter, Miran Kim, Yuzhe Tang, Hongkai Xiong, and Xiaoqian Jiang. Healer: homomorphic computation of exact logistic regression for secure rare disease variants analysis in gwas. *Bioinformatics*, 32(2):211–218, 2015.

[252] Feng Chen, Shuang Wang, Xiaoqian Jiang, Sijie Ding, Yao Lu, Jihoon Kim, S Cenk Sahinalp, Chisato Shimizu, Jane C Burns, Victoria J Wright, et al. Princess: Privacy-protecting rare disease international network collaboration via encryption through software guard extensions. *Bioinformatics*, 33(6):871–878, 2016.

[253] Rui Wang, XiaoFeng Wang, Zhou Li, Haixu Tang, Michael K Reiter, and Zheng Dong. Privacy-preserving genomic computation through program specialization. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 338–347, 2009.

[254] Ernest Yeboah Boateng and Daniel A Abaye. A review of the logistic regression model with emphasis on medical research. *Journal of Data Analysis and Information Processing*, 7(4):190–207, 2019.

[255] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3):13:1–13:36, July 2014.

[256] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. Secure pattern matching using somewhat homomorphic encryption. In *Proceedings of the 2013 ACM workshop on Cloud computing security workshop*, pages 65–76, 2013.

[257] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

[258] Joppe W Bos, Kristin Lauter, and Michael Naehrig. Private predictive analysis on encrypted medical data. *Journal of biomedical informatics*, 50:234–243, 2014.

[259] Sergiu Carpov, Nicolas Gama, Mariya Georgieva, and Juan Ramon Troncoso-Pastoriza. Privacy-preserving semi-parallel logistic regression training with fully homomorphic encryption. *BMC Medical Genomics*, 13(7):1–10, 2020.

[260] Shai Halevi and Victor Shoup. Bootstrapping for helib. In *Annual International conference on the theory and applications of cryptographic techniques*, pages 641–670. Springer, 2015.

[261] Chen Xu, Jingwei Chen, Wenyuan Wu, and Yong Feng. Homomorphically encrypted arithmetic operations over the integer ring. In *International Conference on Information Security Practice and Experience*, pages 167–181. Springer, 2016.

[262] Grant Taylor Frame. Heide: An ide for the homomorphic encryption library helib. 2015.

[263] Daniele Micciancio and Yuriy Polyakov. Bootstrapping in fhew-like cryptosystems. *IACR Cryptol. ePrint Arch.*, 2020:86, 2020.

[264] Tiziano Bianchi and Alessandro Piva. Ttp-free asymmetric fingerprinting based on client side embedding. *IEEE Transactions on Information Forensics and Security*, 9(10):1557–1568, 2014.

[265] Y. Peng, Y. Hsieh, C. Hsueh, and J. Wu. Cloud-based buyer-seller watermarking protocols. In *2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–9, Aug 2017.

[266] Sarita P Ambadekar, Jayshree Jain, and Jayshree Khanapuri. Digital image watermarking through encryption and dwt for copyright protection. In *Recent Trends in Signal and Image Processing*, pages 187–195. Springer, 2019.

[267] Ali Al-Haj, Noor Hussein, and Gheith Abandah. Combining cryptography and digital watermarking for secured transmission of medical images. In *2016 2nd International Conference on Information Management (ICIM)*, pages 40–46. IEEE, 2016.

[268] Ming Li, Di Xiao, Ye Zhu, Yushu Zhang, and Lin Sun. Commutative fragile zero-watermarking and encryption for image integrity protection. *Multimedia Tools and Applications*, pages 1–16, 2019.

[269] R Mothi and M Karthikeyan. Protection of bio medical iris image using watermarking and cryptography with wpt. *Measurement*, 136:67–73, 2019.

[270] Xiaochun Cao, Ling Du, Xingxing Wei, Dan Meng, and Xiaojie Guo. High capacity reversible data hiding in encrypted images by patch-level sparse representation. *IEEE transactions on cybernetics*, 46(5):1132–1143, 2015.

[271] Xinpeng Zhang. Reversible data hiding in encrypted image. *IEEE signal processing letters*, 18(4):255–258, 2011.

[272] Mohamed Elhoseny, Gustavo Ramírez-González, Osama M Abu-Elnasr, Shihab A Shawkat, N Arunkumar, and Ahmed Farouk. Secure medical data transmission model for iot-based healthcare systems. *IEEE Access*, 6:20596–20608, 2018.

[273] Brian Chen and Gregory W Wornell. Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory*, 47(4):1423–1443, 2001.

[274] Smriti Gupta, Sandeep Kumar Yadav, Alok Pratap Singh, and Krishna C Maurya. A comparative study of secure hash algorithms. In *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 2*, pages 125–133. Springer, 2016.

[275] Shahram Bakhtiari, Reihaneh Safavi-Naini, Josef Pieprzyk, et al. Cryptographic hash functions: A survey. Technical report, Citeseer, 1995.

[276] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES-The Advanced Encryption Standard*. Springer Science & Business Media, 2002.

[277] M. Lacharité, B. Minaud, and K. G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 297–314, May 2018.

[278] Reza Ghasemi, Md Momin Al Aziz, Noman Mohammed, Massoud Hadian Dehkordi, and Xiaoqian Jiang. Private and efficient query processing on outsourced genomic databases. *IEEE journal of biomedical and health informatics*, 21(5):1466–1472, 2017.

[279] Paulo Martins, Leonel Sousa, and Artur Mariano. A survey on fully homomorphic encryption: An engineering perspective. *ACM Comput. Surv.*, 50(6):83:1–83:33, December 2017.

[280] Bo Eskerod Madsen and Sharon R. Browning. A groupwise association test for rare mutations using a weighted sum statistic. *PLOS Genetics*, 5(2):1–11, 02 2009.

[281] Herbert A David. The beginnings of randomization tests. *The American Statistician*, 62(1):70–72, 2008.

[282] Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers: Information leaks in genome wide association study. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 534–544, New York, NY, USA, 2009. ACM.

[283] Krishna Murali, Jamwal Pradeep, Chaitanya K. S. R, and Kumar B. Vinod. Secure file multi transfer protocol design. *Journal of Software Engineering and Applications*, 4(5):311–315, 05 2011.

[284] Emmanuelle Genin, Richard Redon, Jean-François Deleuze, Dominique Campion, Jean-Charles Lambert, Jean-François Dartigues, and the FREX Consortium. The french exome (frex) project: a population-based panel of exomes to help filter out common local variants. *International Genetic Epidemiology Society*, 2017.

[285] Anubha Mahajan and Neil Robertson. Rare variant quality control. In *Assessing rare variation in complex traits*, pages 33–43. Springer, 2015.

[286] Jennifer A Tom, Jens Reeder, William F Forrest, Robert R Graham, Julie Hunkapiller, Timothy W Behrens, and Tushar R Bhangale. Identifying and mitigating batch effects in whole genome sequencing data. *BMC bioinformatics*, 18(1):351, 2017.

[287] Kalliope Panoutsopoulou and Klaudia Walter. Quality control of common and rare variants. In *Genetic Epidemiology*, pages 25–36. Springer, 2018.

[288] Bernie Devlin and Kathryn Roeder. Genomic control for association studies. *Biometrics*, 55(4):997–1004, 1999.

[289] Michael C Wu, Seunggeun Lee, Tianxi Cai, Yun Li, Michael Boehnke, and Xihong Lin. Rare-variant association testing for sequencing data with the sequence kernel association test. *The American Journal of Human Genetics*, 89(1):82–93, 2011.

[290] Seunggeun Lee, Mary J. Emond, Michael J. Bamshad, Kathleen C. Barnes, Mark J. Rieder, Deborah A. Nickerson, David C. Christiani, Mark M. Wurfel, and Xihong Lin. Optimal Unified Approach for Rare-Variant Association Testing with Application to Small-Sample Case-Control Whole-Exome Sequencing Studies. *American Journal of Human Genetics*, 91(2):224–237, August 2012.

[291] Alexandros Mittos, Bradley Malin, and Emiliano De Cristofaro. Systematizing genome privacy research: a privacy-enhancing technologies perspective. *Proceedings on Privacy Enhancing Technologies*, 2019(1):87–107, 2019.

[292] Aude Saint Pierre and Emmanuelle Génin. How important are rare variants in common disease? *Briefings in Functional Genomics*, 13(5):353–361, September 2014.

[293] Benjamin M. Neale, Manuel A. Rivas, Benjamin F. Voight, David Altshuler, Bernie Devlin, Marju Orho-Melander, Sekar Kathiresan, Shaun M. Purcell, Kathryn Roeder, and Mark J. Daly. Testing for an unusual distribution of rare variants. *PLoS genetics*, 7(3):e1001322, March 2011.

[294] Corneliu A Bodea, Benjamin M Neale, Stephan Ripke, Murray Barclay, Laurent Peyrin-Biroulet, Mathias Chamaillard, Jean-Frederick Colombel, Mario Cottone, Anthony Croft, Renata D'Incà, et al. A method to exploit the structure of genetic ancestry space to enhance case-control studies. *The American Journal of Human Genetics*, 98(5):857–868, 2016.

[295] Mykyta Artomov, Alexander A Loboda, Maxim N Artyomov, and Mark Daly. A platform for case-control matching enables association studies without genotype sharing. *bioRxiv*, page 470450, 2018.

[296] George M Church, Yuan Gao, and Sriram Kosuri. Next-generation digital information storage in dna. *Science*, 337(6102):1628–1628, 2012.

[297] Shibsankar Das, Uttam Roymondal, and Satyabrata Sahoo. Analyzing gene expression from relative codon usage bias in yeast genome: a statistical significance and biological relevance. *Gene*, 443(1-2):121–131, 2009.

[298] Enrique Viguera, Danielle Canceill, and S Dusko Ehrlich. Replication slippage involves dna polymerase pausing and dissociation. *The EMBO journal*, 20(10):2587–2595, 2001.

[299] Masanori Arita and Yoshiaki Ohashi. Secret signatures inside genomic dna. *Biotechnology progress*, 20(5):1605–1607, 2004.

[300] Dominik Heider and Angelika Barnekow. Dna watermarks: A proof of concept. *BMC molecular biology*, 9(1):40, 2008.

[301] Nozomu Yachie, Kazuhide Sekiyama, Junichi Sugahara, Yoshiaki Ohashi, and Masaru Tomita. Alignment-based approach for durable data storage into living organisms. *Biotechnology progress*, 23(2):501–505, 2007.

[302] Michael Liss, Daniela Daubert, Kathrin Brunner, Kristina Kliche, Ulrich Hammes, Andreas Leiherer, and Ralf Wagner. Embedding permanent watermarks in synthetic genes. *PloS one*, 7(8):e42465, 2012.

[303] Ying-Hsuan Huang, Chin-Chen Chang, and Chun-Yu Wu. A dna-based data hiding technique with low modification rates. *Multimedia Tools and applications*, 70(3):1439–1451, 2014.

[304] Mohammad Reza Abbasy, Pourya Nikfard, Ali Ordi, and Mohammad Reza Najaf Torkaman. Dna base data hiding algorithm. *International Journal of New Computer Architectures and their Applications (IJNCAA)*, 2(1):183–192, 2012.

[305] Ahmed Atito, Amal Khalifa, and SZ Rida. Dna-based data encryption and hiding using playfair and insertion techniques. *Journal of Communications and Computer Engineering*, 2(3):44–49, 2012.

[306] Ban Ahmed Mitras and AK Abo. Proposed steganography approach using dna properties. *international journal of information technology and business management*, 14(1):96–102, 2013.

[307] Minoru Kuribayashi, Takuya Fukushima, and Nobuo Funabiki. Robust and secure data hiding for pdf text document. *IEICE TRANSACTIONS on Information and Systems*, 102(1):41–47, 2019.

[308] Javier Franco-Contreras, Gouenou Coatrieux, Frederic Cuppens, Nora Cuppens-Boulahia, and Christian Roux. Robust lossless watermarking of relational databases based on circular histogram modulation. *IEEE transactions on information forensics and security*, 9(3):397–410, 2013.

[309] Vitaly Mitekin and Victor Fedoseev. A new qim-based watermarking algorithm robust against multi-image histogram attack. *Procedia engineering*, 201:453–462, 2017.

[310] Jyotsna Singh and Abhinav Dubey. Mpeg-2 video watermarking using quantization index modulation. In *2010 IEEE 4th International Conference on Internet Multimedia Services Architecture and Application*, pages 1–6. IEEE, 2010.

[311] Reda Bellafqira, Thomas E Ludwig, David Niyitegeka, Emmanuelle Génin, and Gouenou Coatrieux. Privacy-preserving genome-wide association study for rare mutations-a secure framework for externalized statistical analysis. *IEEE Access*, 8:112515–112529, 2020.

Titre : Composition de mécanismes cryptographiques et de tatouage pour la protection de données génétiques externalisées

Mots clés : Sécurité, données génétiques, études d'association pangénomiques, tatouage, chiffrement homomorphe

Résumé : De nos jours, le "cloud computing" permet de mutualiser et de traiter de grandes quantités de données génétiques à un coût minime et sans avoir à maintenir une infrastructure propre. Ces données sont notamment utilisées dans des études d'association pangénomiques ("Genome Wide Association Studies" ou GWAS) afin d'identifier des variants génétiques associées à certaines maladies. Cependant, leur externalisation induit de nombreux problèmes en matière de sécurité. Notamment, le génome humain représente l'unique identité biologique d'un individu et est donc par nature une information très sensible. L'objectif de ces travaux de thèse est de protéger des données génétiques lors de leur partage, stockage et traitement sur le cloud. Nous avons développé différents outils de sécurité fondés sur le tatouage, des mécanismes cryptographiques et leur combinaison. Dans un premier temps, en utilisant le chiffrement homomorphe, nous avons proposé une version originale sécurisée de l'approche GWAS fondée sur la technique dite "collapsing method" ; une technique qui s'appuie sur la régression logistique. Pour faire face aux problemes de complexité de calcul et de mémoire liés à l'exploitation du chiffrement homomorphe, nous avons proposé un protocole qui profite de différents outils cryptographiques (PGP, fonctions de hachage) pour partager entre plusieurs unités de recherche, des GWAS sur des variants rares de manière sécurisée, cela sans augmenter la complexité de calcul. En parallèle, nous avons développé une méthode de crypto-tatouage qui exploite la sécurité sémantique des schémas de chiffrement homomorphe, pour permettre à un fournisseur de services cloud de protéger/vérifier l'intégrité de bases de données génétiques externalisées par différents utilisateurs. Ce schéma de crypto-tatouage est dynamique dans le sens où le tatouage est réactualisé au fil des mises à jour des données par leurs propriétaires sans cependant retatouer l'ensemble des jeux de données. Dans le même temps, nous avons proposé la première solution de tatouage robuste qui permet de protéger la propriété intellectuelle et le traçage de traitres pour des données génétiques utilisées dans des GWAS.

Title: Composition of cryptographic mechanisms and watermarking for the protection of externalized genetic data

Keywords: Security, genetic data, genome-wide association studies, watermarking, homomorphic encryption

Abstract: Nowadays, cloud computing allows researchers and health professionals to flexibly store and process large amounts of genetic data remotely, without a need to purchase and to maintain their own infrastructures. These data are especially used in genome-wide association studies (GWAS) in order to conduct the identification of genetic variants that are associated with some diseases. However, genetic data outsourcing or sharing in the cloud environments induces many security issues. In addition, a human genome is very sensitive by nature and represents the unique biological identity of its owner. The objective of this thesis work is to protect genetic data during their sharing, storage and processing. We have developed new security tools that are based on watermarking and cryptographic mechanisms, as well as on the combination of them. First, we have proposed a privacy-preserving method that allows to compute the secure collapsing method based on the logistic regression model using homomorphic encryption (HE). To overcome the computational and storage overhead of HE-based solutions, we have developed a framework that allows secure performing of GWAS for rare variants without increasing complexity compared to its nonsecure version. It is based on several security mechanisms including encryption and hash functions. In parallel of these works, we have exploited the semantic security of some HE schemes so as to develop a dynamic watermarking method that allows integrity control for encrypted data. At last, we have developed a robust watermarking tool for GWAS data for traitor tracing purposes and copyright protection.