A Hybrid Cloud Deployment Architecture for Privacy-Preserving Collaborative Genome-Wide Association Studies

Fatima-zahra Boujdad[1,3], David Niyitegeka[2], Reda Bellafqira[2], Gouenou Coatrieux[2], Emmanuelle Genin[4], and Mario Südholt[1,3(✉)]

[1] IMT Atlantique, Nantes, France
{fatima-zahra.boujdad,david.niyitegeka,reda.bellafqira, gouenou.coatrieux,mario.sudholt}@imt-atlantique.fr
[2] IMT Atlantique, Brest, France
[3] Inria & LS2N, Nantes, France
[4] INSERM, Paris, France

Abstract. The increasing availability of sequenced human genomes is enabling health professionals and genomics researchers to well understand the implication of genetic variants in the development of common diseases, notably by means of genome-wide association studies (GWAS) which are very promising for personalized medicine and diagnostic testing. However, the ever present need to handle genetic data from different sources to conduct large studies entails multiple privacy and security issues. Actually, classical methods of anonymization are inapplicable for genetic data that are now known to be identifying per se. In this paper, we propose a novel framework for privacy-preserving collaborative GWAS performed in the cloud. Indeed, our proposal is the first framework which combines a hybrid cloud deployment with a set of four security mechanisms that are digital watermarking, homomorphic encryption, meta-data de-identification and the Intel Software Guard Extensions technology in order to ensure confidentiality of genetic data as well as their integrity. Furthermore, our approach describes meta-data management which has rarely been considered in state-of-the-art propositions despite their importance to genetic analyses; in addition, the new deployment model we suggest fits with existing infrastructures which makes its integration straightforward. Experimental results of a prototypical implementation on typical data sizes demonstrate that our solution protocol is feasible and that the framework is practical for real-world scenarios.

Keywords: Secure GWAS · Privacy · Data watermarking · Homomorphic encryption · Integrity · Intel SGX · Anonymization

1 Introduction

Nowadays, cloud computing services are gaining more and more interest in the field of biomedical analyses and more specifically genetic ones due to its appeal-

Supported by the PrivGen project: https://project.inria.fr/privgen/.

P. Gladyshev et al. (Eds.): ICDF2C 2021, LNICST 441, pp. 342–359, 2022.
https://doi.org/10.1007/978-3-031-06365-7_21

ing characteristics of rapid scalability and low cost. Indeed, cloud computing solutions allow users or data owners to use massive data storage and large computation capabilities while avoiding in-house infrastructure overhead of purchase and maintenance. For instance, cloud computing has greatly contributed to the rapid growth of sequencing data archives (e.g., Sequence Read Archive (SRA) [16]), and genetic data sharing by enabling collaborations on large amounts of data in various genetic projects like the International Cancer Genome Consortium (ICGC) [28].

Despite these benefits, outsourcing genomic private data to environments such as a public cloud platforms exposes it to many security threats ranging from unintentional disclosure (*e.g.,* due to human administrative errors) to on-purpose theft by means of cyber attacks. In fact, human genomes are a very sensitive asset as it allows the unique identification of individuals and contains valuable information, such as their medical condition. Worse, classical anonymization techniques such as identifier deletion or k-anonymity cannot be used for genetic data as information, *e.g.,* about physical appearances, health status can be directly inferred from it. Therefore, cloud services for genetic data storage and/or processing require strong security mechanisms to ensure confidentiality, privacy, and integrity [7].

Several solutions based on homomorphic encryption [5,17,20], secure multiparty computation [4,11] and differential privacy [26,27] have been proposed for securing genetic data during their storage and/or processing on the cloud. Unfortunately, these methods often induce important memory and computation time cost, huge network loads or simply compromise the usability of the data, *e.g.,* because of noise addition, thus making them impractical for analysis of large-scale genetic data in real life scenarios.

Besides, we are recently witnessing the emergence of new hybrid approaches combined of cryptographic software and hardware in order to support privacy-preserving and better-performing security and data protection frameworks. For instance, Canim *et al.* [8] came up with a framework to secure queries like count and join over genetic data using secure co-processors; Sadat *et al.* [23] combine Paillier' cryptosystem with SGX in an architecture featuring several distributed data contributors and GWAS computations that are conducted at a central server. Chen *et al.* [9] consider data querying in addition to data outsourcing problems for which they exploit the enclave sealing feature. Finally, for a distributed genetic analysis, Chen *et al.* also propose, see [10], secure family-based analyses for Transmission Disequilibrium tests using SGX, AES encryption and compression techniques. However, all these frameworks for secure GWAS still do not specify in their deployments how the meta-data is being accessed in order to select individuals whose genetic data will be used for the analysis. They also often tackle the problem from the confidentiality point of view not considering other vital security features such as integrity. In fact, in all these hardware-based protection methods, confidentiality is guaranteed by secure cryptography hardware and software encryption (*e.g.,* homomorphic encryption) while integrity is ensured by traditional digital signature methods or message authentication codes. However, these integrity

control mechanisms add more information to the data [19] which may increase the storage overhead if large-scale data is considered. In order to overcome this inconvenience, solutions based on watermarking have been proposed [15,21] that do not add extra data to the target database. Furthermore, the architectures considered in aforementioned frameworks often rely on in-house infrastructures and public cloud providers seen as a central server for combined-data analyses, thus not supporting more distributed analyses.

In this paper, we propose using a more comprehensive set of security mechanisms for GWAS together with a deployment model for secure GWAS based on hybrid cloud computing. This model supports community and public cloud providers and combines four security mechanisms in order to protect outsourced genetic data during storage and processing. More precisely, we propose three contributions in this paper:

- We present a novel hybrid cloud-based deployment model for secure collaborative GWAS.
- We introduce a framework to secure genetic analyses that provides four security mechanisms: asymmetric/symmetric encryption, watermarking, Intel's SGX and de-identification.
- A proof of concept implementation for a collaborative χ^2 statistical analysis in the context of a GWAS. The implementation is parallelized for time-consuming tasks and has been executed on the proposed deployment model.

The rest of this paper is organized as follows. Section 2 presents the background preliminaries about homomorphic encryption, de-identification, Intel SGX and encrypted data watermarking. The proposed privacy-preserving GWAS framework is detailed in Sect. 3. Experimental results and discussion are given in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Background Information

In this section, we present background information about genetic association tests and security mechanisms. Concretely, we introduce the GWAS case/control statistical method, SGX technology and metadata anonymization through fragmentation and homomorphic encryption.

2.1 Genetic Association Tests

To find genes related to common diseases, a strategy consists in testing for an association by comparing marker genotype distributions among individuals affected by the disease (case group) and individuals who are not (control group). Markers are usually single nucleotide polymorphisms (SNPs) with two alleles that can be any of the four nucleotides adenine (A), thymine (T), guanine (G) or cytosine (C). Hence, for a SNP with A and C alleles, the three possible genotypes are AA, AC and CC and test data can be summarized in a 2×3 table, cf. Table 1, with the numbers of cases having genotypes AA, AC and

CC respectively in the first row and the numbers of controls with these same genotypes in the second row. To test for an association, a chi-square test can be used that tests the null hypothesis that the genotype distribution is the same in the two groups. This test is a two degrees of freedom test and may lack power. In many genetic studies, a co-dominant model is assumed where the effects of alleles are cumulative and the Cochran-Armitage test for trend [1] is used. This is, in particular, the case in studies where hundred thousands to millions of SNPs spanning the entire genome are tested iteratively.

Table 1. Contingency table with totals.

	AA	AC	CC	Total
Case	o_{11}	o_{12}	o_{13}	T_{l_1}
Control	o_{21}	o_{22}	o_{23}	T_{l_2}
Total	T_{c_1}	T_{c_2}	T_{c_3}	N

Pearson's chi-square test is defined as follows:

$$\tilde{\chi}^2 = \sum_{i=1}^{2} \sum_{j=1}^{3} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{1}$$

where O_{ij} corresponds to the witnessed count from cell (i, j); E_{ij} is the expected value, defined as: $E_{ij} = \frac{T_{l_i} \times T_{c_j}}{N}$. Here, T_{l_i} and T_{c_j} are the marginal totals of lines and columns respectively, as shown in Table 1, N is the number of individuals in the study.

We use this test, which is commonly used in GWAS inferences [29], for illustration purposes; however, any contingency table based statistical test can be handled using our proposal. Moreover, it is noteworthy to point out that, as we consider the scenario of several data owners that collaboratively analyze all of their data, an additional step is added before deducing the statistical test value. In this additional step the global contingency table is computed: o_{ij}^k being the value from cell (i, j) in a contingency table of the k^{th} data owner among n entities, the global contingency table is calculated by simply summing values from corresponding cells.

2.2 Intel's Software Guard Extensions (SGX)

The recent hardware-based technology for secure remote computations provided by Intel processors which is known under the name SGX (for Software Guard Extensions) appeared for the first time in 2015. The SGX is an architecture extension providing a new set of instructions that allows the creation and management of an isolated environment, called an enclave, at the CPU level. The isolation is

implemented in such a way that even the host operating system, the hypervisor and any other privileged software is forbidden from accessing the enclave.

The SGX technology is designed to preserve integrity as well as confidentiality of the computations inside an enclave. Actually, two main features are at the heart of the SGX security model: i) software attestation, that is, the means by which users verify that they are communicating with legitimate hardware that contains their expected program (and data eventually) before they can send their private inputs to the enclave program; ii) data sealing which consists in storing enclave secrets securely at the hard disk level of the host system so that only the enclave that created the sealed data is able to recover it when required. Here, we only harness the attestation process for the provisioning of secret keys to the enclave.

2.3 Data Fragmentation and De-identification

Genetic data is identifying per se, but the process of re-identification is facilitated if the associated meta-data is not well protected. A well-known method to render meta-data, *e.g.,* clinical and demographic attributes, unexploitable to an attacker is de-identification. One de-identification technique consists in dropping any direct or non direct identifiers from the meta-data when they are exposed to a non trusted environment. Such identifiers can be a name or an association of several attributes as the zip code, gender and day of birth, so-called quasi-identifiers [25]. Constraint-based fragmentation techniques have been used as a mechanism for the de-identification of such data. Fragmentation consists primarily in splitting a database to separate identifying features [3,12] and/or in encrypting attributes when necessary: this requires a deployment model involving several servers for data storage and processing so that the splitting constraints can be satisfied.

In the example we consider, only two servers are available for meta-data storage, one of the two being trusted; therefore, the de-identification mechanism we apply on the meta-data consists in keeping singleton identifying attributes, *e.g.,* social security number, along with attributes dropped from each identifying association at the trusted server and outsource the rest. Actually, it is enough to separate only one attribute from each sensitive association for this last to become non-sensitive. To illustrate this, consider the following example: meta-data is abstracted to attributes, those with security-sensitive properties are between parentheses, whether they are singleton or associations:

attributes: `Id`, (`SSN`), (`Zip code`, `Birth date`, `Gender`), `Case/Control`
separate `SSN`, `Birth date`
store `SSN`, `Birth date` on the trusted server
outsource `Id`, `Zip code`, `Gender`, `Case/Control`

The choice of which attributes to separate from association constraints can be driven by the expected information content that represents how much an attribute can help re-identify subjects [13]. Doing so protects the identities of

participating subjects in a study but still allows researchers to customize the selection of case/control groups by keeping the phenotype attribute accessible in addition to other non-identifying clinical information. Indeed, we consider keeping the phenotype status attribute accessible to researchers while anonymizing the data on the untrusted side.

2.4 Homomorphic Encryption

Homomorphic encryption (HE) is a security mechanism that enables performing operations on encrypted data without the need for decryption. Decrypted results correspond to the same output that would be achieved when these operations are performed on clear data [2]. In practice, these operations are limited to linear ones, such as additions and multiplications. Formally, let *Enc* and *Dec* be the encryption and decryption functions of an HE cryptosystem respectively, and K_p and K_s be its public key and secret key respectively. If m_1 and m_2 are two clear messages, the homomorphism property states:

$$Dec(Enc(m_1, K_p) \star Enc(m_2, K_p), K_s) = m_1 \circ m_2 \tag{2}$$

where $\star$ and $\circ$ are two operators in the encrypted and the clear domains, respectively.

In this work, we opted for Paillier's HE cryptosystem because of the additive homomorphic property and its simplicity of use [22]. Notice that it is also semantically secure in that the same plain-text has different cipher-texts at each execution. Paillier cryptosystem is additively homomorphic; its utility is straightforward for secure GWAS analyses where computing global tables given contributed contingency tables is done through addition of corresponding values for each SNP.

2.5 Watermarking of Homomorphically Encrypted Genetic Data

In our framework, the integrity control of homomorphically encrypted data is ensured using a database watermarking method recently proposed by Niyitegeka *et al.* [21].

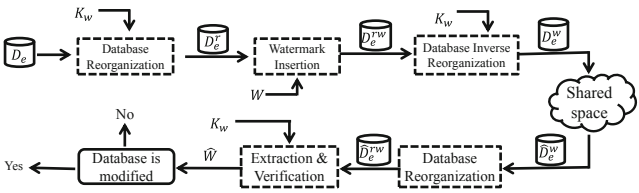

Fig. 1. Steps of watermark embedding and watermark extraction for used watermarking scheme.

Watermark Embedding in Encrypted Databases. As shown in Fig. 1, the watermark embedding process takes as input a homomorphically encrypted database D_e which is first secretly reorganized into a database D_e^r using a secret watermarking key K_w. This reorganization is used in order to ensure that an attacker cannot get access to the secret watermark. To do so, a secret hash is associated to each tuple (line) of the database, and all tuples are reorganized according to their hash values and in ascending order. After reorganizing database tuples, D_e^r is divided into overlapping blocks (see [21]) and one bit of the watermark W is inserted into each block of D_e^r, so as to produce a reorganized and watermarked database D_e^{rw}. Once all blocks are watermarked, the database D_e^{rw} is reorganized back into the encrypted and watermarked database D_e^w. Notice that the watermark W can be randomly generated or generated using the identifier of the user.

Watermark Extraction from Encrypted Database and Integrity Verification. The watermark extraction can be conducted in a similar way as the watermark embedding (see Fig. 1). Indeed, to extract the watermark from a suspicious database $\widehat{D}_e^w$, this database is first reorganized into $\widehat{D}_e^{rw}$ using the secret watermarking key K_w, and $\widehat{D}_e^{rw}$ is divided into blocks. The hash is then computed for each block so as to get access the watermark bit. Once all watermark bits are extracted to form $\widehat{W}$, this information is compared to the original watermark W in order to conduct integrity verification. Any difference between these watermarks will indicate that the database D_e^w was illegally modified.

3 Privacy-Preserving GWAS Hybrid Cloud Deployment

In this section, we introduce a new hybrid model to deploy genetic analyses and handle two major security concerns, confidentiality and integrity of genetic data. We use four security techniques to fulfill the security requirements for data transfer and processing as well as result delivery to researchers. We build a comprehensive framework with a new deployment model for secure genetic data processing of GWAS. Concretely, we use a static version of our scheme for the dynamic watermarking of homomorphic encrypted databases [21]. In addition, we are using SGX coupled with homomorphic encryption for secure processing of χ^2 statistic test, thus following a computation protocol inspired by Sadat *et al.* [23]. Finally, we use our meta-data de-identification techniques through fragmentation [6].

3.1 System Infrastructure

Figure 2 shows a typical system architecture that we are interested in. It represents a collaboration of five types of parties:

– Genetic data owners, (E_i in the Fig. 2). The owners, *e.g.,* research labs, keep genetic data (represented, for instance, by means of VCF files) locally while

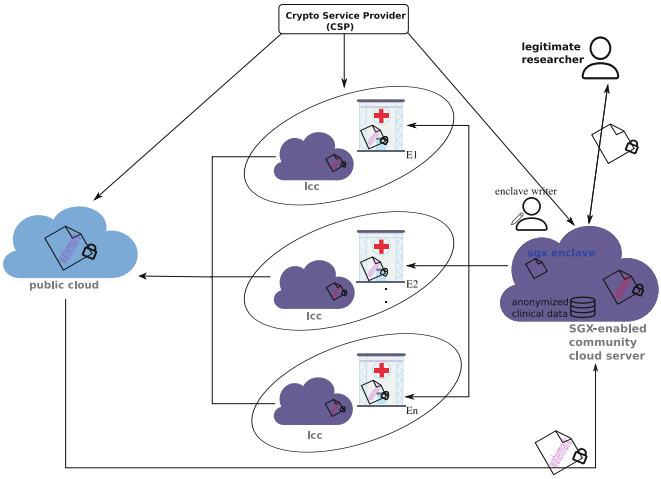

Fig. 2. PrivGen deployment architecture.

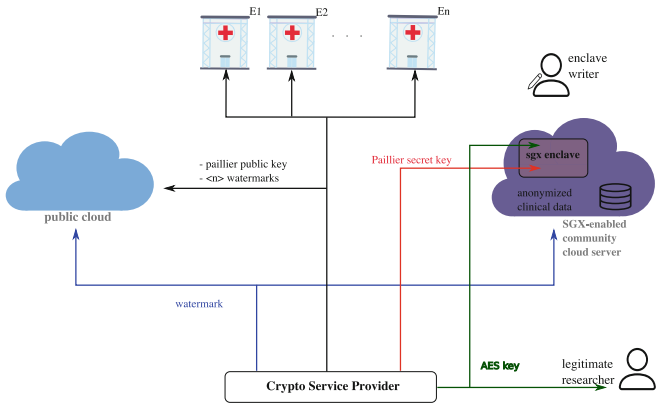

Fig. 3. Key distribution by the CSP.

they outsource the associated de-identified meta-data to a community cloud storage service.

– Local community cloud servers (LCCs). Due to the strong security guarantees of these clouds, they are well adapted to serve as independent providers for compute-intensive operations required by applications of data owners. Such infrastructures are well known to biomedical actors: the French Institute of Bioinformatics[1], for instance, provides a cloud infrastructure providing computing and storage resources for the biomedical and bioinformatics community.

[1] https://www.france-bioinformatique.fr/en/infrastructure-0.

- A hybrid cloud provider that provisions two infrastructure domains i) a community cloud (ComC), where data owners keep demographic/clinical data allowing researchers to get transparent access to it and ii) a public cloud (PubC) that handles homomorphic computations on encrypted contingency tables pooled from several owners.
- Researchers, who have authenticated access to the ComC storage platform for meta-data querying purposes. Actually, doing so affords researchers with an accurate selection of genetic data to form the case-control groups which is of main interest in study design.
- A Cryptography Service Provider (CSP), who handles watermarking and encryption key generation and proper distribution.
- An enclave program writer who is in charge of developing a secure module destined to run as an SGX enclave at the SGX-enabled ComC.

3.2 Key Distribution

The cryptography service provider is in charge of key generation and distribution for both encryption and watermarking operations. Indeed, the CSP generates a Paillier key pair (K_s, K_p); the public key is communicated to data owners which they use to encrypt the contingency tables, while the private key is securely shared with the SGX enclave (during the attestation protocol) where the global tables will be decrypted. The CSP also handles distribution of watermarking keys. More precisely, it generates a set of watermarking keys whose number is equivalent to the number of geneticists who use the platform, and it shares these keys with geneticists and the public cloud. The CSP also generates a watermarking key that will be used at the public cloud in order to watermark the resulting global contingency table. This step is added to ensure integrity during data transfer between public and community clouds; hence, the community cloud also receives the expected watermark from the CSP. Finally, the CSP generates an AES key which it securely shares with the SGX enclave and the researcher. This key is used to encrypt χ^2 values inside the enclave so that only legitimate researchers (who hold the decryption key) are able to read the final results. These elements are depicted in Fig. 3.

3.3 Attacker Model

In our secure GWAS application, we aim at protecting the confidentiality and integrity of genetic data and guaranteeing the generation of correct results as well as preventing the re-identification of the participating subjects. Given the key distribution schema presented above, these requirements can be achieved under a semi-honest attacker type for the HCP. Indeed, the HCP provider could straightforwardly insert noise into the data using the Paillier' public key which is inherently available had they been a dishonest party; this would output false results and thus violate the requirement of getting correct results. As to preserving data confidentiality, we need the enclave writer to be independent from the HCP whom we assume not to collude with to get access to Paillier' private key;

hence, as long as it is the CSP (fully trusted) who provisions the SGX enclave with secret keys, data confidentiality is preserved throughout the lifetime of the application; this also means that the SGX technology is fully trusted to meet the security functionalities it does afford. Data integrity might be compromised by the HCP if they ignore cases where integrity check fails, this again would result in generating false results at the last step of the analysis; under a semi-honest attacker model (that is realistic when considering a cloud provider who cares about their commercial reputation), this scenario is excluded.

Besides, we suppose researchers to be honest entities: a malicious or semi-honest researcher with background knowledge on some subjects may still be able to jeopardize their presence in the ComC database and hence infer their health status; such an attack can be countered by techniques like differential privacy and k-anonymity which we have not considered in this work.

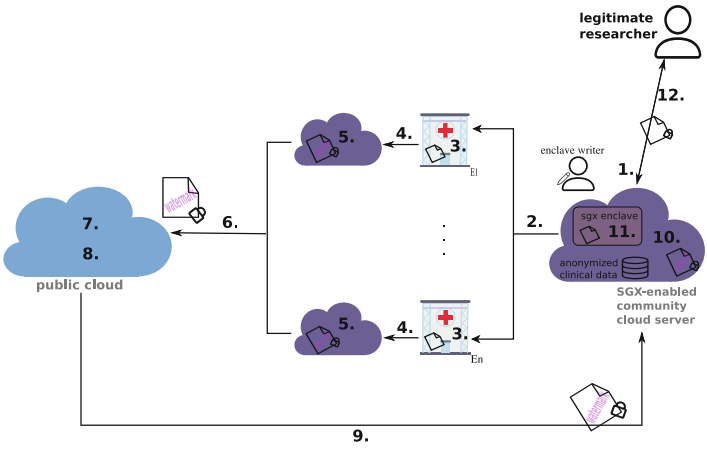

Fig. 4. GWAS workflow.

3.4 Protocol Workflow

We assume the initial state of the architecture is set by outsourcing anonymized clinical data for storage on the community cloud platform and that an authentication system is deployed to allow legitimate researchers to access and query the data. On the other hand, VCF files of genetic data are kept locally at their owners' sites. Hence, linking primary keys should be set so that these files can always be mapped to associated clinical data. This is for researchers who want to select genetic data based on clinical phenotypes and/or demographic attributes. The primary keys can be represented by the anonymous subject identifiers found in VCF files.

Using the architecture of Fig. 2 and assuming the above initial setting, a GWAS analysis can be performed as follows (cf. Fig. 4):

1. The researcher queries ComC for meta-data to form the case-control groups.
2. ComC executes the query, then forwards the primary database keys to the data owners.
3. The owners compute contingency tables from VCF files of involved individuals.
4. They prepare the resulting tables for encryption on their LCC servers.
5. The LCC servers encrypt the contingency tables with the (same) Paillier public key, then they watermark the tables with a secret watermark shared with the PubC.
6. The owners send the encrypted-watermarked tables to the public cloud servers.
7. Upon data reception, PubC first verifies data integrity by verifying the watermark to ensure the data has not been tampered with. If this check fails, a 'resend' request is sent.
8. The public cloud proceeds with computing the global contingency table homomorphically which it watermarks at the end with a secret watermark shared with the ComC.
9. The HCP moves the global contingency tables from the PubC to the ComC server.
10. The SGX-enabled host system of ComC verifies data integrity by ensuring that the inserted watermark is the expected one after which it forwards the rest of the computations to an initiated SGX enclave.
11. Inside the enclave, the tables are decrypted and the χ^2 test is computed after the global table was extended with totals (cf. Table 1), then the resulting values are independently encrypted using AES with a secret encryption key shared with the researchers.
12. Finally, the encrypted χ^2 result values are communicated to researchers who can decrypt the results and infer the p-values.

3.5 Deployment Choices

In this section, we justify the architectural choices advanced in Sect. 3.1. As a reminder, privacy concerns and meta-data storage/access are the major factors defining the architectural elements and the data flows across the infrastructure servers. We discuss our architectural choice with respect to scalability, security, performance and cost criteria. We show, in particular, why simpler architectures like a centralized server are not appropriate for our purposes.

Scalability. The public cloud deployment has been chosen because of the scalability and elasticity properties it affords so that both horizontal and vertical growth can be accommodated with ease. This, in turn, enables arbitrary numbers of data owners to be admitted and large numbers of SNPs to be analyzed.

At the community cloud infrastructure, only vertical scalability has to be supported: the number of global contingency tables to handle depends on the number of considered SNPs and not on that of the contributing entities. For these reasons, we believe the public cloud is the necessary element that guarantees horizontal scalability of the architecture. However, the community cloud is required to perform well even when the number of SNPs becomes huge. Our experimental results below show that our approach meets these requirements and provides a practical solution for scenarios with hundreds of thousands of SNPs.

Unlike the public cloud, the community cloud infrastructure is assumed to have much less computation resources. It is mainly used to permanently store meta-data and handle secure computations thanks to SGX support; the SGX part of the computations has to be strictly minimized. Therefore, we entrust the community cloud host system to check integrity by handling watermark detection operations; only decryption, the χ^2 test, and the final AES encryption are computed inside the enclave. This task allocation is quite realistic as integrity checks do not reveal any secret inputs but they are still very important for correct computation within the enclave.

Security and Privacy. We do not perform the SGX part of the computations on the public cloud (but allocate them to the community cloud) because we would like to avoid side-channel attacks on SGX processors. Since public cloud platforms are multi-tenant, intruders are much more likely to succeed in performing such attacks [14,18]. This arrangement requires (additional) network exchanges between the PubC and the ComC, but we do believe the increased level of security is worth this cost. Furthermore, the introduction of a community cloud allows to envision meta-data management and global access provides stronger guarantees to data owners. They, in turn, can then provide more flexible access to their data to researchers, in particular add data attributes to their shared meta-data that otherwise would not be available to researchers.

Performance. Following the protocol workflow depicted by Fig. 2, the contingency tables' totals are generated inside the enclave just before computing the χ^2 value. This is clearly more efficient than homomorphically generating totals directly in the public cloud or at the geneticists level, because the enclave would then need to perform later six additional decryption operations per SNP in case of a genotype contingency table and five in case of a table with allele counts.

Finally, the watermarking system we chose contributes to enhancing the overall performance of the preprocessing phase: the offline step of homomorphically encrypting contingency tables can result in storing the data for some time before the online process starts. Data integrity can still be checked continuously by data owners thanks to watermark embedding done just after the encryption. If ever the integrity check fails the extracted wrong watermark indicates which block of the data was modified; this means that the data owner can re-encrypt only the modified parts of it and re-watermark it with properly chosen bits (whether the same as in the first watermark or completely new ones). Doing so is much more

efficient than with a classical signature whose failure of integrity check leads to re-encryption of all the tables.

Business Model. It is noteworthy to mention that this architecture can be wholly ephemeral: both the global community and public clouds are elastic elements; this is useful in a case where *e.g.,* a project of a couple of months uses a community cloud for the same period but only needs the public cloud occasionally where joint computations are required on private genetic data. In general, such a hybrid cloud improves the security of implementation while leveraging a cost-effective business. In fact, community cloud deployment type tends to be more expensive than a public cloud: using both with markedly differentiated hardware resources permits to couple an affordable budget with an enhanced security level compared to pure public-cloud deployments.

4 Experimental Results

In the following we evaluate an implementation, especially its performance, of a GWAS analysis that executes a χ^2 test using our deployment scheme and framework to secure the analysis.

4.1 Experimental Setup

We have used the following global setup:

- *LCC servers* perform Paillier encryption and watermarking of contingency tables. The encryption uses multiple nodes and all available (in our experiments, 90) cores by splitting the VCF file and assigning each part to a different node for processing. The watermarking operations are executed in a parallel fashion on (in our experiments, 18 cores of) a single node.
- *PubC modules* serve to verify watermarks and compute global tables. Watermark detection is parallelized and the global tables computation uses all of the node's cores.
- *ComC modules* also verify watermarks (by the host system), perform Paillier decryption, compute χ^2 tests and AES encrypt the final results (within the SGX enclave). This part of the application uses one core only.

The VCF file we have worked with contained only 2952 SNPs; for experimental purposes, we have made copies of the available contingency tables to obtain tables of approximate size of 500K. We report execution times on the basis of 1024 bits Paillier keys, 256 bits AES key. The watermark sizes depend on the size of the input file. In order to minimize the number of watermarks to manage, we reorganize the encrypted contingency tables by using bigger chunks so that several tables are contained in a single file of $N \times 6$ entries where N is the number of SNPs of each chunk. In every chunk, each line represents a table of the consecutive values o_{11} — o_{12} — o_{13} — o_{21} — o_{22} — o_{23}, cf. Table 1.

Unfortunately, the watermarking algorithm performance was a bottleneck when we chose $N \approx 500$. Therefore, we have split the file with encrypted tables into smaller subfiles, then parallelized the insertion/detection functions. In our case, we chose $N = 5000$, thus, all the data was split into 101 files.

We launched all experiments five times each and report the average time as performance values.

Table 2 presents the hardware infrastructure setup used in our experiments.

Table 2. Experiments' hardware infrastructure.

	Nodes	Processor	Cores/Node
LCC	5	Intel Xeon Gold 5220	18
sgx-en. ComC	1 laptop	Intel Core i7 1.80 GHz	1
PubC	1	Intel Xeon Gold 5220	18

The multi-core servers are part of the French testbed infrastructure Grid5000[2]. We used the cluster "gros" located in the French city Nancy.

4.2 Results

We now present the computation times obtained for each module as previously stated without considering network latency. We report results for two cases: i) contingency tables comprise genotype counts, meaning that six values per table have to be processed, and ii) the contingency tables comprise allele counts, which means that only four values are handled across the lifetime of the application. Though execution times are naturally lower in the second case, real GWAS studies would rather consider contingency tables of genotypes [24], the reason being that testing with allele counts assumes that the two alleles are independent (Hardy-Weinberg principle) which is not always true. However, testing with genotype counts is possible even if one genotype lacks in the table (yielding one degree of freedom less); in the context of our framework this is equivalent to handling four values. In other words, our experiments conducted using allele counts also provides information about the performance of genotype-based scenarios where a genotype might be missing.

The results for genotype counts and allele counts for different numbers of geneticists are, respectively, reported in Tables 3 and 4. The execution times change slightly but significantly with increasing numbers of geneticists at the public cloud level because it homomorphically computes the global contingency table by aggregating tables from an increasing number of entities. At the data owners level, execution times do not change because the experiments were set in such a way that they launch their modules in parallel; a valid assumption for real scenarios given that this part of the solution is performed offline, that is, as

[2] https://www.grid5000.fr.

Table 3. Execution times for ∼500K SNPs (genotypes).

Data owners	lcc-crypt	lcc-wat	PubC	ComC
3	110 min	2 min	1 min 52 s	53 min
6	110 min	2 min	3 min 34 s	54 min
9	110 min	2 min	5 min 13 s	54 min
20	110 min	2 min	11 min 23 s	55 min

Table 4. Execution times for ∼500K SNPs (alleles).

Data owners	lcc-crypt	lcc-wat	PubC	ComC
3	75 min	1 min 23 s	1 min 8 s	36 min
6	75 min	1 min 22 s	2 min 6 s	36 min
9	75 min	1 min 22 s	3 min	36 min
20	75 min	1 min 22 s	6 min 45 s	36 min

part of a preprocessing phase. The enclave code also runs in constant time with respect to an increasing number of geneticists because the number of SNPs is fixed which means that the file size (5000×6 for genotype counts and 5000×4 for allele counts) is the same across the three experiments. From the tables above, we infer that the total execution time per SNP including all data flow steps for genotype counts ranges from 20 to 21 ms, while alleles account for approximately 14 ms.

The data flow related to asymmetric operations in our solution is generally the same as in the SAFETY framework [23]. Its authors reported execution times of the SGX enclave module for one genotype contingency table: homomorphic decryption and Fisher's Exact Test (FET) computation take over 7 ms run on an Intel Core i7 3.40 GHz processor. In our framework, Paillier's decryption, chi-square test computation and AES encryption of results take 6.5 ms approximately on the designated hardware (see Table 2). Actually, the chi-square test and FET formulae are different, but as these tests are computed in the clear, we believe the differences in performance is due to the fact that SAFETY delegates all decryption operations to the enclave including the decryption of total values, while we calculate totals in the enclave after the values have been decrypted.

It is very important to mention that the enclave code does not make use of all of the available cores. However, the SGX technology permits multi-threaded code to execute, we leave this extension as a future optimization which should largely enhance the community module performance. In addition, we could also parallelize the watermarking functions themselves so that many threads can insert/detect in/from a single file so that only one (or very few) watermarks have to be managed by the CSP.

5 Conclusion

In this paper, we have presented a new framework that harnesses the use of a hybrid cloud computing deployment model to secure collaborative Genome-Wide Association Studies. We used watermarking, encryption, Intel SGX hardware execution and de-identification in order to ensure integrity as well as confidentiality of genetic data and the associated meta-data. We have also reported results from a proof of concept implementation using the proposed framework. The results show that our approach enables efficient execution of biomedical analyses that can be scaled significantly. Overall, our results show the feasibility of our solution and that it can be used to comprehensively secure real-world biomedical analyses executed on grid-like architectures of already deployed resources.

As future work, we will optimize the three code modules that are part of our architecture, notably, by parallelizing the enclave code. In addition, we will extend our results to accommodate the inference of p-values at the enclave level. Furthermore, we will focus on requiring less resources from geneticists so that most of the heavy computations can be securely outsourced.

Acknowledgments. This work has received a French government support granted to the Labex CominLabs and managed by the ANR in the "Investing for the future" program under reference ANR-10-LABX-07-01, and to the Labex GenMed, ANR-10-LABX-0013, through the project PrivGen.

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

References

1. Armitage, P.: Tests for linear trends in proportions and frequencies. Biometrics **11**(3), 375–386 (1955)
2. Bellafqira, R., Coatrieux, G., Bouslimi, D., Quellec, G.: Content-based image retrieval in homomorphic encryption domain. In: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 2944–2947. IEEE (2015)
3. Bkakria, A., Cuppens, F., Cuppens-Boulahia, N., et al.: Preserving multi-relational outsourced databases confidentiality using fragmentation and encryption. JoWUA **4**(2), 39–62 (2013). https://hal.archives-ouvertes.fr/hal-01213956
4. Bogdanov, D., Kamm, L., Laur, S., Sokk, V.: Implementation and evaluation of an algorithm for cryptographically private principal component analysis on genomic data. IEEE/ACM Trans. Comput. Biol. Bioinf. **15**(5), 1427–1432 (2018)
5. Bonte, C., et al.: Privacy-preserving genome-wide association study is practical. In: IACR Cryptology ePrint Archive 2018, p. 955 (2018)
6. Boujdad, F.-z., Südholt, M.: Constructive privacy for shared genetic data. In: CLOSER 2018–8th International Conference on Cloud Computing and Services Science, Proceedings of CLOSER 2018, March 2018

7. Bouslimi, D., Coatrieux, G., Cozic, M., Roux, C.: A telemedicine protocol based on watermarking evidence for identification of liabilities in case of litigation. In: 2012 IEEE 14th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 506–509. IEEE (2012)
8. Canim, M., Kantarcioglu, M., Malin, B.: Secure management of biomedical data with cryptographic hardware. Trans. Info. Tech. Biomed. **16**(1), 166–175 (2012). https://doi.org/10.1109/TITB.2011.2171701. http://dx.doi.org/10.1109/TITB.2011.2171701. ISSN 1089–7771
9. Chen, F., et al.: Presage: privacy-preserving genetic testing via software guard extension. BMC Med. Genomics **10**(2), 48 (2017)
10. Chen, F., et al.: Princess: privacy-protecting rare disease international network collaboration via encryption through software guard extensions. Bioinformatics (Oxford, England) **33**(6), 871–878 (2017). https://doi.org/10.1093/bioinformatics/btw758. https://www.ncbi.nlm.nih.gov/pubmed/28065902. ISSN 1367–4811
11. Cho, H., Wu, D.J., Berger, B.: Secure genome-wide association analysis using multiparty computation. Nat. Biotechnol. **36**(6), 547 (2018)
12. Ciriani, V., De Capitani, S., Vimercati, D., Foresti, S., et al.: Combining fragmentation and encryption to protect privacy in data storage. ACM Trans. Inf. Syst. Secur. **13**(3), 22:1–22:33 (2010). https://doi.org/10.1145/1805974.1805978. http://doi.acm.org/10.1145/1805974.1805978. ISSN 1094–9224
13. Erlich, Y., Narayanan, A.: Routes for breaching and protecting genetic privacy. Nat. Rev. Genet. **15**(6), 409–421 (2014). https://doi.org/10.1038/nrg3723. ISSN 1471–0064
14. Götzfried, J., Eckert, M., Schinzel, S., Müller, T.: Cache attacks on intel SGX. In: Proceedings of the 10th European Workshop on Systems Security, EuroSec 2017, pp. 2:1–2:6. ACM, New York (2017). https://doi.org/10.1145/3065913.3065915. http://doi.acm.org/10.1145/3065913.3065915. ISBN 978-1-4503-4935-2
15. Khanduja, V., Chakraverty, S.: A generic watermarking model for object relational databases. Multimedia Tools Appl. **78**(19), 28111–28135 (2019). https://doi.org/10.1007/s11042-019-07932-3
16. Langmead, B., Nellore, A.: Cloud computing for genomic data analysis and collaboration. Nat. Rev. Genet. **19**(4), 208 (2018)
17. Lauter, K., López-Alt, A., Naehrig, M.: Private computation on encrypted genomic data. In: Aranha, D.F., Menezes, A. (eds.) LATINCRYPT 2014. LNCS, vol. 8895, pp. 3–27. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16295-9_1
18. Moghimi, A., Irazoqui, G., Eisenbarth, T.: CacheZoom: how SGX amplifies the power of cache attacks. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 69–90. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66787-4_4
19. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. ACM Trans. Storage (TOS) **2**(2), 107–138 (2006)
20. Nassar, M., Malluhi, Q., Atallah, M., Shikfa, A.: Securing aggregate queries for DNA databases. IEEE Trans. Cloud Comput. **7**(3), 827–837 (2017). https://doi.org/10.1109/TCC.2017.2682860. ISSN 2168–7161
21. Niyitegeka, D., Coatrieux, G., Bellafqira, R., Genin, E., Franco-Contreras, J.: Dynamic watermarking-based integrity protection of homomorphically encrypted databases – application to outsourced genetic data. In: Yoo, C.D., Shi, Y.-Q., Kim, H.J., Piva, A., Kim, G. (eds.) IWDW 2018. LNCS, vol. 11378, pp. 151–166. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11389-6_12

22. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16
23. Sadat, M.N., Al Aziz, M.M., Mohammed, N., Chen, F., Jiang, X., Wang, S.: Safety: secure gwAs in federated environment through a hybrid solution. IEEE/ACM Trans. Comput. Biol. Bioinform. (TCBB) **16**(1), 93–102 (2019)
24. Sasieni, P.D.: From genotypes to genes: doubling the sample size. Biometrics **53**(4), 1253–1261 (1997)
25. Sweenay, L.: Simple demographics often identify people uniquely. Carnegie Mellon. Data Privacy Working Paper 3 (2000). http://dataprivacylab.org/projects/identifiability/
26. Tramèr, F., Huang, Z., Hubaux, J.-P., Ayday, E.: 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, pp. 1286–1297. ACM, New York (2015)
27. Fei, Yu., Ji, Z.: Scalable privacy-preserving data sharing methodology for genome-wide association studies: an application to iDASH healthcare privacy protection challenge. BMC Med. Inform. Decis. Mak. **14**(1), S3 (2014)
28. Yung, C.K., et al.: ICGC in the cloud (2016)
29. Zeng, P., et al.: Statistical analysis for genome-wide association study. J. Biomed. Res. **29**(4), 285–297 (2015). https://doi.org/10.7555/JBR.29.20140007. https://pubmed.ncbi.nlm.nih.gov/26243515. 26243515[pmid]. ISSN 1674–8301