A Novel Secret Sharing Approach for Privacy-Preserving Authenticated Disease Risk Queries in Genomic Databases

Maitraye Das**
Technology and Social Behavior
Northwestern University
Evanston, IL, USA
Email: maitraye@u.northwestern.edu

Nusrat Jahan Mozumder,* Sharmin Afrose,†
Khandakar Ashrafi Akbar Shama,‡ and Tanzima Hashem§
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology, Dhaka, Bangladesh
Email: *deeptee.cse12@gmail.com, †sharmin.afrose.bd@gmail.com,
‡aninditaashrafi@gmail.com, §tanzimahashem@cse.buet.ac.bd

***Abstract*—Recent improvement in genomic research is paving the way towards significant progress in diagnosis and treatment of diseases. A disease risk query returns the probability of a patient to develop a particular disease based on her genomic and clinical data. Despite various innovative prospects, frequent and ubiquitous usage of genomic data in medical tests and personalized medicine may cause various privacy threats like genetic discrimination, exposure of susceptibility to diseases, and revelation of genomic data of relatives. Another major concern is on ensuring the reliability of the genome data and the correctness of the computed disease risk, which is known as authentication. We develop a novel secret sharing approach to protect privacy of sensitive genomic and clinical data, disease markers, disease name, and the query answer while ensuring authenticated result of the disease risk query. Experiments with real datasets show that our approach for authenticated disease risk queries achieves a high level of privacy with reduced processing and storage overhead.**

***Index Terms*—Genomic privacy, secret sharing, authenticated disease risk queries**

I. Introduction

Rapid advancement of efficient and cost-effective genome sequencing has opened the door for various novel research directions in genomics. In recent years, researchers have focused on revealing the correlation between genetic variants and an individual's predisposition to diseases or response to the treatment. Thus, genomic data has become popular for early diagnosis and proper treatment of diseases [25]. For example, people having family history of HIV, cancer, leukemia, heart disease, or diabetes may want to measure the risk of inheriting these diseases in advance so that proper diet and preventive treatment can be adopted [1]. Besides, accurate dosage of medicine can also be suggested according to patients' genetic makeup [19].

With this pervasive usage of genomic data in personalized medicine, privacy of an individual is going through potential risks, as genomic data may reveal sensitive information regarding an individual's ethnicity, ancestry, phenotypic traits, health conditions and susceptibility to specific diseases [16]. In addition, a person's genomic data can reveal sensitive information of the person's close relatives (possibly without their consents) due to hereditary nature of genome [17]. Therefore, to continue the growth of revolutionary applications on genomes, privacy protection is essential. We focus on protecting privacy of genome data while processing a *disease risk query*, i.e., the probability of an individual to develop a specific disease.

Besides protecting privacy of genome data, another major challenge is to authenticate a disease risk query. Processing a person's disease risk query involves outside entities like a data center, and thus raises concerns on the reliability of the genome data used for a disease risk query and the correctness of the computed disease risk. Authentication ensures that the disease risk is correctly computed using a person's actual genome data. We develop a novel secret sharing approach for privacy-preserving authenticated disease risk queries.

Gene sequencing is done by a *certified* Sequencing Institute (SI) [7], [8], [9], [12], [15], which may be directed by the government or any trusted party. In our approach, the SI distributes SNPs [4] (Single Nucleotide Polymorphism) of genome data among several authorized Distributed Databases (DDBs), where one DDB is located at the patient's device. The key idea of our approach is that SNPs remain hidden in an aggregate form, and the probability to develop a specific disease is computed by combining partial genetic scores for the specific disease from all the DDBs. If a dishonest DDB alters a patient's SNP data and provides a wrong partial genetic score, then our authentication technique can detect the alteration using an authentication key generated based on the stored SNP data at the DDB and thus verify the correctness of the computed disease risk. Additionally, we show that not a single SNP of a patient can be identified without involving the patient even if all the DDBs become compromised. The portion of data that our approach stores on a patient's device does not cause any significant overhead in terms of the storage size. On the other hand, our approach does not store the full data on a patient's device to ensure that it is also not possible to identify

**This work was conducted while the first author was a student at Bangladesh University of Engineering and Technology.

a patient's SNP from the patient's DDB without compromising the other DDBs.

Over the last years, though researchers have developed a few cryptographic approaches for privacy-preserving disease risk queries [7], [8], [9], [12], [14], [15], these approaches cannot authenticate the query answer. Another major limitation of these techniques is that they cannot answer a disease risk query accurately when different alleles of the same SNP in genomes are responsible for two or more different diseases. For example, allele C of SNP rs6313 holds higher risk for rheumatoid arthritis, whereas allele T of the same SNP contributes to depression, panic and stress response [5]. Specifically, existing approaches [7], [8], [9], [12], [15] store the frequency of one allele for an SNP (considering that this allele is always responsible for diseases) in encrypted form in a single Data Center (DC). The DC can only *partially decrypt* the frequency information when it receives a disease risk query from a Medical Unit (MU). Though all common SNPs have two possible allele variations, it is not possible for the DC to infer the frequency of the other allele in the SNP from the partially decrypted frequency of one allele. The DC sends *encrypted* frequency information (not the partially decrypted ones) to the MU. The MU can also *partially decrypt* the frequency information and thus, cannot infer the frequency of the other allele. Only possible way to infer the frequencies of both alleles is the collusion of the MU and the DC, which is not allowed since the collusion will eventually reveal the genome data to both parties and violate user privacy. These two limitations are not possible to overcome by any trivial computation. On the other hand, *our approach ensures privacy of genomes even if the dishonest MU and the DDBs collude and can evaluate disease risk queries when two alleles of the same SNP are responsible for two or more different diseases*. Recently, in [23], Turkmen *et al.* have proposed a cryptographic approach to authenticate computed disease risks. However, this approach also has not considered storing both alleles or a dishonest medical unit. More importantly, the authors have not performed any experiment to validate the performance of their approach.

Besides, existing approaches incur high storage overhead due to encryption and the overhead would be doubled if the encrypted frequencies of both alleles of an SNP are stored. Nowadays, the provision of low cost genome sequencing is attracting an increasing number of people to use disease risk queries. The number of SNPs responsible for different diseases is also increasing. Thus, reducing the storage overhead has become an important challenge for any privacy preserving approach for disease risk queries. Our approach offers a substantial improvement in reducing storage cost.

A disease risk query for a patient is processed using an MU's disease marker that consists of the SNPs associated with a particular disease, their risk alleles (i.e., which one of the two possible alleles of each SNP is responsible for this particular disease), and contribution factors of risk alleles. Though the SNPs associated with a particular disease and risk alleles are publicly known, it may happen that an MU wants to keep contribution factors of risk alleles confidential from others. On the other hand, it is possible to infer the name of a disease from the publicly available contents of the disease marker, i.e., the SNPs associated with a particular disease and risk alleles. However, a patient may not feel comfortable to disclose the disease name such as Alzheimer's to any party except the MU for treatment purposes. Thus, it is also essential to hide the number and IDs of SNPs and their risk alleles used in the disease risk query to protect privacy of the disease name. Our approach ensures the privacy of a patient's genomic and clinical data, an MU's disease marker, disease name, and the query answer, i.e., the probability of a patient to develop a particular disease.

To the best of our knowledge, we develop the first secret sharing approach for privacy preserving authenticated disease risk queries that eliminates the cryptographic overheads for processing encrypted genomes. In summary, the contributions of our paper are as follows:

- We propose a novel secret sharing approach to privately compute the probability of a patient to develop a specific disease without revealing genomic data, clinical data, the disease name and the query answer to others.
- We ensure that our proposed technique can evaluate disease risk queries when different alleles of the same SNP are responsible for different diseases.
- We authenticate the query results sent by dishonest DDBs to ensure the correctness of the disease risk.
- We protect privacy of genome data against the dishonest MU and its collusion with the DDBs.
- We provide solution to hide the MU's disease markers from others.

II. Preliminaries

A. Genomic Background

DNA consists of two complimentary polymer chains of four nucleotides: Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). If a single nucleotide in the DNA differs between members of the same species or paired polymer chains of an individual, the variation is called an SNP [4]. For example, DNA fragments $C\underline{T}GG$ and $C\underline{C}GG$ differ in a single nucleotide. Each SNP carries two alleles (i.e. two nucleotides); one at each polymer chain of the DNA. For each of the two alleles, an SNP generally has only two probable nucleotides among A, C, G, and T. Both probable nucleotides of an SNP can increase risks for two or more different diseases. For example, SNP rs6313 has two variations C and T [5]. Let f_i denote the number of risk allele r_i in the SNP S_i, where $f_i \in \{0, 1, 2\}$. That means, if C is the risk allele for any particular disease, f_i of three patients having genotype CC, CT, and TT are 2, 1, and 0 respectively.

B. Contribution of Clinical Data in Disease Risk

Along with genomic data, clinical factors of an individual can contribute significantly to her disease risks, especially for chronic diseases like Coronary Artery Disease, Diabetes etc. The clinical factors can include demographic information

(e.g., age, sex etc.), his family history of diseases, laboratory test results (e.g., cholesterol level, blood sugar level etc.). For this reason, clinical data should also be considered along with genomic data in the computation of the disease risks of an individual [22].

C. Computation of Disease Risk

The contribution factor of an SNP S_i is defined as $\beta_i = ln(OR_i)$, where OR_i represents the extent to which S_i is associated with a certain disease. Let λ be the number of SNPs associated with disease X, β_i be the contribution factor of S_i, and f_i be the number of risk allele for S_i. Let ϕ be the number of clinical factors associated with disease X, $\bar{\beta}_i$ be the contribution factor of the clinical data C_i and v_i be the value of C_i, where $v_i \in \{0, 1\}$. Note that we can easily convert clinical data to binary form (e.g., whether age > 50 or not). Let Pr be the probability of a patient P to develop disease X and Z be the total disease score. Following recent approaches [9], [15], [12], we have

$$Z = ln\left(\frac{Pr}{1-Pr}\right) = \sum_{i=1}^{\lambda} \beta_i \times f_i + \sum_{i=1}^{\phi} \bar{\beta}_i \times v_i \quad (1)$$
$$\Rightarrow Pr = \frac{e^Z}{1+e^Z}$$

III. Related Work

In recent years, researchers have focused on protecting privacy of genomic data while computing the probability of developing a particular disease. In [8], Ayday *et al.* have proposed privacy-preserving disease risk queries using modified paillier cryptosystem and proxy re-encryption. In [9], Ayday *et al.* have considered clinical data in addition to genomic data for evaluating a disease risk query. In [15], Danezis *et al.* have identified that it is possible to infer disease name from the IDs and number of SNPs used in a disease risk query and developed solutions to overcome this attack. In [12], Barman *et al.* have proposed countermeasures to genomic data retrieval attack by dishonest-but-covert medical unit based on the architecture of [8], [9]. Using techniques similar to [8], Turkmen *et al.* used message authentication code and verifiable computing to check correctness of disease susceptibility tests in [23]. All of these approaches store data in encrypted form in a semi-honest data center and require high computing power and storage facility [6]. On the contrary, we develop a secret sharing approach that does not need to store encrypted genomic data. We also provide necessary authentication measures considering dishonest databases and medical unit. Furthermore, these approaches assume that storage and medical units never collude and also fail to give the correct answer when two alleles of the same SNP are responsible for two or more different diseases. These limitations have been addressed in our approach.

All of the above approaches including ours consider the impact of multiple genetic variants, i.e., SNPs to compute the probability to develop a particular disease. Earlier approaches [11], [13] used a small genetic fragment for medical tests and developed different private string searching and sequence comparison techniques for this purpose. However, these techniques cannot produce accurate answer for a disease risk query as they do not take into account multiple genetic variants needed in many medical tests [22].

It is shown in [21] that secret sharing techniques are more efficient than encryption-based techniques for privacy-preserving data mining with respect to communication, computation and storage cost. Secure multi-party computation-based secret sharing techniques have been used to protect privacy in evaluating count and ranked queries [10] and in GWAS (Genome-Wide Association Studies) [18], [26]. However, to the best of our knowledge, no previous work adopts secret sharing techniques or naïve bit encoding [24] (the encoding that we used) to protect genomic privacy for disease risk queries.

IV. System Overview

Like existing systems [7], [8], [9], [15], [12], a trusted sequencing institute (SI) performs the sequencing of genomic data of a patient (P). P provides her sample (e.g. hair, saliva etc.) to the SI for genome sequencing. The SI distributes the SNPs of P and relevant information for authentication of genomic data among n independent databases (DDBs). We assume that the DDBs are run by separate authorities such as private companies, cloud storage services or non-profit organizations under the supervision of the government. The n^{th} DDB is stored in the patient's personal computer or mobile device. The SI sends data to all the DDBs except the n^{th} DDB in plain format. On the other hand, the SI encrypts genomic data and authentication key using TDES [20] before sending them to the patient and the patient decrypts the data before storing them to the n^{th} DDB. At this point, it may be argued that the SNP contents could be stored as a whole in the patient's device instead of n separate DDBs. However, patient's device can easily be hacked or stolen leaving the genomic data in risk. In our system, we ensure that even if the patient's device is hacked, genomic data is secure, unless other $n-2$ DDBs are also compromised (Section V-B). The system architecture is shown in Fig. 1.

A Medical Unit (MU) normally located at a health center, has the IDs of SNPs and clinical data responsible for various diseases, risk allele and contribution factor corresponding to each SNP or clinical factor. A pseudonym is assigned to each patient at the time of gene sequencing and used to store genomic data in the DDBs to hide the identity of a patient from adversaries. When a patient P wants to know her probability of developing a particular disease X, P sends her encrypted pseudonym to the MU. The MU decrypts the received data using TDES. We use TDES only for securing communication of the SI and the MU with P.

The MU sends P's pseudonym, the IDs of relevant SNPs, their risk alleles, and scaled contribution factors responsible for developing disease X and other randomly selected $l-1$ dummy diseases to all $n-1$ DDBs except the patient's device and makes X indistinguishable from l diseases (details in

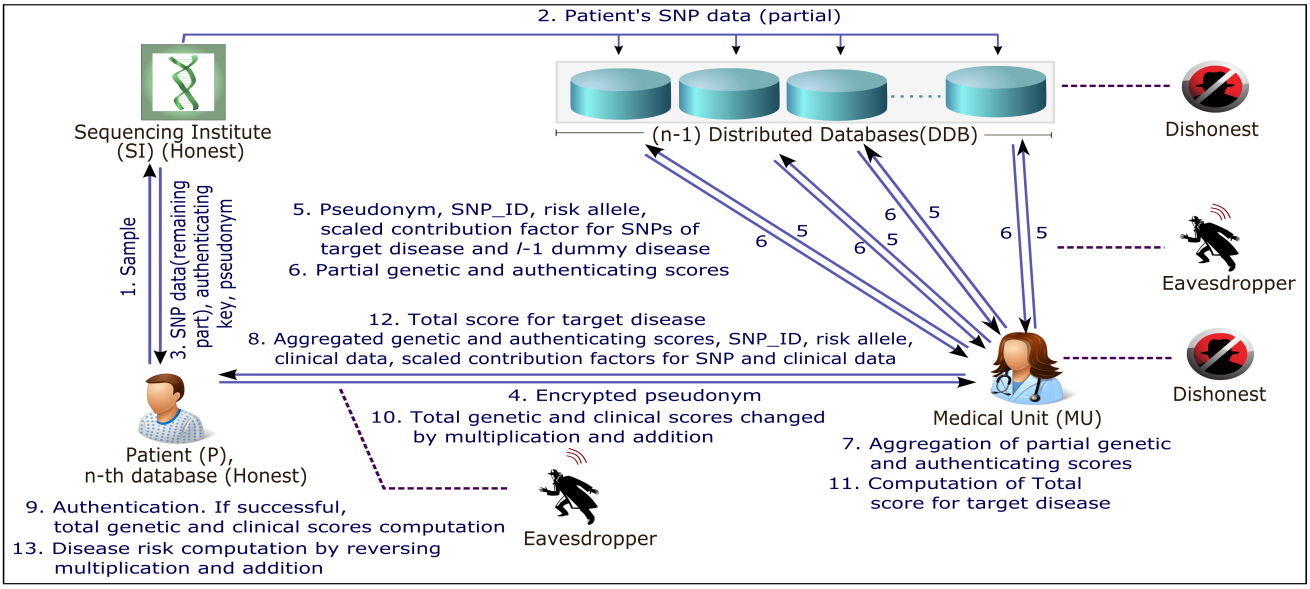


Fig. 1. System architecture of our secret sharing approach for privacy-preserving authenticated disease risk query

Section V-C1). The MU scales the contribution factors of the SNPs (β) by random constants to ensure that original β values cannot be inferred by the DDBs. Each DDB computes partial genetic and authenticating scores using P's genomic data stored in the database and scaled β values received from the MU, and sends back the partial scores to the MU. The MU separately sums up partial genetic and authenticating scores sent from $n-1$ DDBs. Along with these aggregated genetic and authenticating scores, SNPs of all the l diseases, their risk alleles and scaled contribution factors, the MU sends clinical data related to l diseases and their contribution factors scaled by random constants to the n^{th} DDB at the patient's device. Patient P verifies the correctness of the aggregated genetic scores using the authenticating scores sent by the MU and the authentication key stored in its database. A patient can detect if other $n-1$ DDBs or the MU alter the genomic data (see Theorem A.2 in Appendix). After authenticating the aggregated genetic scores, P calculates the total genetic and clinical scores, modifies these scores using multiplication and addition operations, and sends to the MU. The MU first scales back the genetic and clinical scores of the target disease X, and then sends the combined score to P. Finally, P accurately computes the disease risk probability by reversing the effect of previous multiplication and addition operations (see Theorem A.1 in Appendix).

We involve the patient to make sure that not a single SNP is disclosed to anyone without the consent of the patient even if the other DDBs along with the MU are compromised. One may argue that a patient may not agree to take the burden of authentication and storage. We note that our approach is also applicable if a patient does not store the n^{th} DDB, i.e., the n^{th} DDB is run by a separate authority like other $n-1$ DDBs. However, in this case, the patient's privacy is slightly reduced; SNPs of a patient can be identified and authentication process can fail if an adversary compromises $n-1$ DDBs (including the n^{th} DDB).

V. Our Approach

We discuss the steps of our approach in the following subsections.

A. Gene Sequencing

A patient (P) provides her sample, e.g. saliva, hair etc. to the SI. The SI sequences the sample and extracts SNPs from the raw genomic data. A pseudonym and an authentication key μ for P are generated and given to P, where μ is a constant. The pseudonym is used instead of P's actual name and identity to store her genomic data in the DDBs.

B. Storing Data in the Distributed Databases

SNPs are stored in n independent databases and all databases (DDBs) collectively give the actual SNP contents. Each SNP has a unique position and a unique ID and almost all common SNPs have only two probable nucleotides among A, C, G, and T for each of the two alleles. For example, SNP rs6313 has two variations C and T [5]. Other two variations A and G are not possible in SNP rs6313.

Each DDB stores nucleotides of two alleles of an SNP separately using naïve bit encoding as a bit string of length 2 (00, 01, 10, and 11 representing A, C, G, and T, respectively). The actual nucleotide of each allele of an SNP is stored on a randomly selected m DDBs, where $m \leq n-2$ and the other possible nucleotide on the remaining $(n-m-1)$ DDBs. In the n^{th} DDB located at the patient's device, we store both possible nucleotides (e.g., C and T). For each allele of an SNP, we also store a random weight such that the summation of weights from all the DDBs for the true nucleotide of the allele becomes 1 and the false one becomes 0 (so that its impact on the disease risk computation is nullified). Neither of the total weight (i.e., 1 or 0) can be inferred unless the patient's DDB is stolen and other $n-2$ DDBs are compromised. On the other hand,

though all $n-1$ DDBs store the same pseudonym for a single patient as the primary key, it is not possible to predict the total weight of an allele without knowing the weights stored in the n^{th} DDB at the patient's device even if all $n-1$ DDBs collude.

For authentication purpose, each DDB stores another value α for each allele of an SNP such that the summation of weights of that allele from all the $n-1$ DDBs equals the summation of α for that allele from all n DDBs including the patient's device scaled by the authentication key, μ.

Let $a_{j,k}$, $w_{j,k}$ and $\alpha_{j,k}$ denote the j^{th} allele of an SNP in the k^{th} DDB, its weight, and corresponding α value assigned to it, respectively. We note that $j \in \{1,2\}$ as each SNP has two alleles, $k \in \{1,2,\ldots,n-1\}$, $a_{j,k} \in \{00,01,10,11\}$ and $-100 < w_{j,k}, \alpha_{j,k} < 100$. The n^{th} DDB does not have the pseudonym but stores two possible nucleotides of each allele $a_{j,n,t}$ for $t \in \{1,2\}$, corresponding weights $w_{j,k,t}$, and authenticating values $\alpha_{j,k,t}$ for an SNP.

Consider SNP S_1 has two variations C and T and patient P has CT in her genome for S_1. Table I shows a possible distribution of weights in 5 DDBs. For the 1st allele, we have 01 in $a_{1,1}$, $a_{1,3}$ and $a_{1,5,2}$, and 11 in $a_{1,2}$, $a_{1,4}$ and $a_{1,5,1}$. The weights of 01 and 11 are 1 ($w_{1,1}+w_{1,3}+w_{1,5,2}$) and 0 ($w_{1,2}+w_{1,4}+w_{1,5,1}$), respectively. Thus, 01 (i.e., C) is true content of the first allele. Similarly, we can see that 11 (i.e., T) is true content of the other allele. Let $\mu = 6$. We can see that for allele 11, $w_{1,2}+w_{1,4} = (\alpha_{1,2}+\alpha_{1,4}+\alpha_{1,5,1}) \times 6 = 48$.

TABLE I
SAMPLE ENTRIES FOR SNP S_1, $k = DDB$ NO.

k		$a_{1,k}$	$w_{1,k}$	$\alpha_{1,k}$	$a_{2,k}$	$w_{2,k}$	$\alpha_{2,k}$
1		01	-12	-1	11	62	2
2		11	50	7	11	-2	2
3		01	0	-8	01	46	8
4		11	-2	-8	01	8	0
5	$t=1$	11	-48	9	11	-59	6
	$t=2$	01	13	7	01	-54	1

C. Computation of Disease Risk

1) Query Processing at the MU: To hide the identity of the target disease, X from a curious party at the DDBs or eavesdroppers, the MU chooses $l-1$ distinct dummy diseases $(Y_1, Y_2, \ldots, Y_{l-1})$ from different types of disease groups other than disease X, so that the protection provided to the patient is not mitigated. For example, if breast cancer is the targeted disease, the dummy diseases will be chosen such that they are not different types of cancers. Otherwise, the DDBs might conclude that the patient has some kind of cancer.

Next, IDs of SNPs associated with all the l diseases are retrieved with their corresponding risk alleles. Let $\mathbb{P}(X), \mathbb{P}(Y_1), \mathbb{P}(Y_2), \ldots, \mathbb{P}(Y_{l-1})$ be the sets of SNPs related to target disease X and dummy diseases $Y_1, Y_2, \ldots, Y_{l-1}$, respectively. The MU also retrieves the contribution factors of the SNPs related to the target disease, X from its database. For the SNPs of the dummy diseases, random values are generated as contribution factor, β. To hide the contribution factors from the adversaries, the MU scales the β_i value of each SNP, S_i belonging to the j^{th} disease set in the query message using a randomly generated constant c_j, where $j \in \{1,\ldots,l\}$. The MU does not disclose the value of c_j to others. Let the scaled β_i value of each SNP S_i be ε_i, such that $\varepsilon_i = \beta_i \times c_j$. Note that the scaling constants c_js are distinct for different diseases.

Consider an example, where the number of DDBs, $n = 5$ and $l = 2$. SNPs related to only one disease Y_1 are used as dummies along with SNPs of the target disease X. Let $\mathbb{P}(X) = \{S_1, S_4\}$ and $\mathbb{P}(Y_1) = \{S_2, S_3, S_5\}$.

All SNP sets related to different diseases with their relevant risk alleles (r_i) and scaled β_i values, i.e., ε_i are accumulated *randomly* to generate the final query message, M. The random organization restricts the DDBs to recognize which SNP set is related to the target disease and which ones to dummies. To scale back the query result derived from the DDBs, the MU saves index value, j of the target disease and constant c_j. Let this index value be γ and the constant be δ. The final query message, M is generated as follows:

$$S_2, 00, \varepsilon_2 : S_3, 01, \varepsilon_3 : S_5, 10, \varepsilon_5 \,|\, S_1, 11, \varepsilon_1 : S_4, 00, \varepsilon_4 \,|$$

As the 2nd SNP set is associated with the target disease X, the MU saves $\gamma = 2$ and $\delta = c_2$ to scale back the results sent by the DDBs. Finally, the MU sends M to each DDB except the n^{th} DDB at the patient's device.

2) Partial Genetic Score Calculation at the DDBs: Each DDB except the n^{th} DDB at the patient's device uses the query message, M, and patient P's pseudonym, N, to calculate partial genetic and authenticating scores for disease X. Algorithm 1 shows the pseudocode used by the k^{th} DDB to generate the partial scores. It produces return message, R_k as output that contains partial genetic and authenticating scores calculated by the k^{th} DDB.

After necessary parsing, Line 4 finds the ID of the SNP S_i, its risk allele r_i and scaled contribution factor ε_i related to each of the diseases in M. Using the pseudonym, N, Function *RetrieveValues* in Algorithm 1 retrieves the total weight ($\boldsymbol{\omega}_{i,k}$) and the sum of α values ($\boldsymbol{\alpha}_{i,k}$) for the risk allele, r_i of SNP S_i from the k^{th} DDB (Line 5). The function matches r_i with the stored alleles, $a_{1,k}$ and $a_{2,k}$ of S_i. If both the alleles match r_i, *RetrieveValues* returns the summation of corresponding weights $w_{1,k}$ and $w_{2,k}$ as $\boldsymbol{\omega}_{i,k}$ and the summation of values $\alpha_{1,k}$ and $\alpha_{2,k}$ as $\boldsymbol{\alpha}_{i,k}$. If one of these alleles matches r_i, *RetrieveValues* returns the corresponding weight as $\boldsymbol{\omega}_{i,k}$ and the corresponding α as $\boldsymbol{\alpha}_{i,k}$. If none of the alleles matches r_i, 0 is returned as $\boldsymbol{\omega}_{i,k}$ and $\boldsymbol{\alpha}_{i,k}$.

TABLE II
SAMPLE ENTRIES FOR SNP S_4, $k = DDB$ NO.

k		$a_{1,k}$	$w_{1,k}$	$\alpha_{1,k}$	$a_{2,k}$	$w_{2,k}$	$\alpha_{2,k}$
1		00	-5	-9	00	-5	-3
2		10	-56	-10	00	-25	-9
3		00	-13	1	10	18	-6
4		10	-10	-8	10	0	4
5	$t=1$	00	19	5	00	30	7
	$t=2$	10	66	7	10	-17	5

Consider the second SNP set $\langle S_1, 11, \varepsilon_1 : S_4, 00, \varepsilon_4 \rangle$ of the example in Section V-C1. Table II shows sample distributions

Algorithm 1 *CalculatePartialGeneticScore*

Input: M, N
Output: R_k, where k is the number of the DDB
1: **for** each disease set $T_j \in M$ **do**
2: $s_{j,k} \leftarrow 0$, $m_{j,k} \leftarrow 0$
3: **for** each SNP $S_i \in T_j$ **do**
4: $S_i, r_i, \varepsilon_i \leftarrow$ *Parse*(T_j)
5: $\boldsymbol{\omega}_{i,k}, \boldsymbol{\alpha}_{i,k} \leftarrow$ *RetrieveValues*(S_i, r_i, N)
6: $s_{j,k} \leftarrow s_{j,k} + \boldsymbol{\omega}_{i,k} \times \varepsilon_i$
7: $m_{j,k} \leftarrow m_{j,k} + \boldsymbol{\alpha}_{i,k} \times \varepsilon_i$
8: **end for**
9: R_k.*append*("$s_{j,k}, m_{j,k}$:")
10: **end for**
11: **return** R_k

of weight values in 5 DDBs for SNP S_4. From Tables I and II, we see that at the 1st DDB, the retrieved weight of risk allele of S_1 (11) and S_4 (00) are, $\boldsymbol{\omega}_1$ = 62, and $\boldsymbol{\omega}_4$ = (-5)+(-5) = -10 respectively. Therefore, the partial genetic score is, $s_{1,1} = 62 \times \varepsilon_1 - 10 \times \varepsilon_4$. Similarly, partial authenticating score is, $m_{1,1} = 2 \times \varepsilon_1 - 12 \times \varepsilon_4$. In this way, 1st DDB calculates partial genetic and authenticating scores for $l = 2$ combinations and sends back reply message, R_1 to the MU. A sample R_1 looks like $R_1 = s_{1,1}, m_{1,1} : s_{2,1}, m_{2,1}$.

3) Query processing at the MU for the n^{th} DDB: The MU extracts partial genetic and authenticating scores from the return messages R_k sent by each of the $n-1$ DDBs. Let $s_{j,k}$ and $m_{j,k}$ respectively be a partial genetic and an authenticating score sent by the k^{th} DDB for the j^{th} SNP set related to any particular disease, where $j \in \{1, \ldots, l\}$. The partial scores in the return messages are maintained in sequence with the SNP sets in the query message, M. The authentication process can detect if a dishonest DDB changes the order or value of the partial scores (see Theorem A.2 in Appendix). The MU separately adds up all the partial genetic and authenticating scores sent by $n-1$ DDBs to generate the sum $\eta_{j,s}$ and $\eta_{j,m}$, respectively. The SNP set for each disease in the query message, M sent to the $n-1$ DDBs are concatenated with these summation values to generate the new query message, $\bar{M}$ that will be sent to the n^{th} DDB.

The MU retrieves the set of clinical data, $\mathbb{N}(D)$ and contribution factors of these clinical data, $\bar{\beta}$, where D can be any of the l diseases in the query - target and dummy ones. Each set of clinical data related to a disease is randomly partitioned into two separate subsets. To hide the contribution factors (secret of the MU) from malicious parties, $\bar{\beta}_i$ of each clinical data C_i associated with the r^{th} subset of j^{th} disease is multiplied by a randomly generated constant $\bar{c}_{r,j}$ to generate the scaled contribution factor, $\bar{\varepsilon}_i$, such that $\bar{\varepsilon}_i = \bar{\beta}_i \times \bar{c}_{r,j}$, where $r \in \{1, 2\}, j \in \{1, \ldots, l\}$. Note that $\bar{c}_{r,j}$ values are distinct for different diseases. The MU saves the scaling constants $\bar{\delta}_r = \bar{c}_{r,j}$, where j^{th} disease is the target disease. Note that $\bar{\delta}_r \neq \delta$, where δ is the constant used to scale the contribution factors of the SNPs related to the target disease. Clinical data are partitioned into two subsets so that the n^{th} DDB cannot infer the contribution factors from the aggregated disease risk.

All the clinical data and their contribution factors are appended at the end of the SNP set for the related disease in the query message, $\bar{M}$. Finally, MU sends $\bar{M}$ to the n^{th} DDB at the patient's device. Continuing our previous example, we assume that $\mathbb{N}(X) = \{C_1, C_2, C_4\}$ and $\mathbb{N}(X)$ is partitioned into two subsets, $\mathbb{N}_1(X) = \{C_1, C_2\}$ and $\mathbb{N}_2(X) = \{C_4\}$. For the dummy disease, $\mathbb{N}_1(Y_1) = \{C_3\}$, and $\mathbb{N}_2(Y_1) = \{C_5\}$. Similar to the previous query message, M, a sample for the new query message, $\bar{M}$ can be as follows:

$\eta_{1,s}, \eta_{1,m}; S_2, 00, \varepsilon_2 : S_3, 01, \varepsilon_3 : S_5, 10, \varepsilon_5 ; C_3, \bar{\varepsilon}_3 :: C_5, \bar{\varepsilon}_5|$
$\eta_{2,s}, \eta_{2,m}; S_1, 11, \varepsilon_1 : S_4, 00, \varepsilon_4 ; C_1, \bar{\varepsilon}_1 : C_2, \bar{\varepsilon}_2 :: C_4, \bar{\varepsilon}_4|$

4) Authentication at the n^{th} DDB: After receiving the query message, $\bar{M}$, the n^{th} DDB at the patient's device authenticates the aggregated genetic score sent from the other $n-1$ DDBs and calculates the total genetic and clinical scores for all the l diseases. Algorithm 2 shows the pseudocode for this process. The input to this algorithm is the query messages, $\bar{M}$, the SNP set related to the target disease, $\mathbb{P}(X)$, the authentication key μ stored at the patient's device, and two randomly generated constants ρ and τ used to change the total scores by multiplication and addition. The output is the reply message $\bar{R}$ containing the total scores of l diseases. The SNPs associated with a particular disease and their risk alleles are normally available in public. Since patient P naturally knows the name of the target disease, X, we assume that $\mathbb{P}(X)$ is also known to her.

Function *GetIndex* in Algorithm 2 matches $\mathbb{P}(X)$ with the SNP sets in $\bar{M}$ to find the index, γ, of the target disease, X in the query message, $\bar{M}$. After necessary parsing, the algorithm finds the aggregated genetic score $\eta_{j,s}$, aggregated authenticating score $\eta_{j,m}$ and ID of the SNP S_i, its risk allele r_i and scaled contribution factor ε_i related to each of the diseases in $\bar{M}$. Similar to Algorithm 1, Function *RetrieveValues* in Algorithm 2 retrieves the total weight ($\boldsymbol{\omega}_{i,n}$) and the sum of α values ($\boldsymbol{\alpha}_{i,n}$) for the risk allele, r_i of SNP S_i from the n^{th} DDB at the patient's device (Line 6). The function matches r_i with the stored alleles, $a_{1,n,1}$, $a_{1,n,2}$, $a_{2,n,1}$ and $a_{2,n,2}$ of S_i. The weight $\boldsymbol{\omega}_{i,n}$ is calculated by summing those weight ($w_{i,n,t}$) values, whose corresponding allele encoding matches r_i, where $t \in \{1, 2\}$. Similarly, $\boldsymbol{\alpha}_{i,n}$ is calculated by summing the $\alpha_{i,n,t}$ values of the matched alleles. We note that the total number of risk allele r_i in the SNP S_i is, $f_i = \sum_{1 \leq k \leq n} \boldsymbol{\omega}_{i,k}$.

Line 7 multiplies $\boldsymbol{\alpha}_{i,n}$ values with the scaled contribution factor, ε_i of each SNP S_i and adds up with the aggregated authenticating score $\eta_{j,m}$. The parameter $\eta_{j,m}$ is multiplied by the authentication key μ and checked whether the multiplied value is equal to the aggregated genetic score $\eta_{j,s}$ (Line 9). If the result does not match, then the n^{th} DDB decides that the genetic scores are altered or disease sequence is changed by dishonest $n-1$ DDBs or a dishonest MU. Otherwise, if the aggregated genetic score $\eta_{j,s}$ is authenticated as correct, Line 10 checks if the j^{th} disease is the target disease, i.e., $j = \gamma$ or

Algorithm 2 *CalculateAuthenticatedScore*

Input: $\bar{M}, \mathbb{P}(X), \mu, \rho, \tau$
Output: $\bar{R}$

1: $\gamma \leftarrow$ *GetIndex*$(\bar{M}, \mathbb{P}(X))$
2: **for** each disease set $T_j \in \bar{M}$ **do**
3: $\eta_{j,s}, \eta_{j,m} \leftarrow$ *Parse*(T_j)
4: **for** each SNP $S_i \in T_j$ **do**
5: $S_i, r_i, \varepsilon_i \leftarrow$ *Parse*(T_j)
6: $\boldsymbol{\omega}_{i,n}, \boldsymbol{\alpha}_{i,n} \leftarrow$ *RetrieveValues*(S_i, r_i)
7: $\eta_{j,m} \leftarrow \eta_{j,m} + \boldsymbol{\alpha}_{i,n} \times \varepsilon_i$
8: **end for**
9: **if** $\eta_{j,s} = \eta_{j,m} \times \mu$ **then**
10: **if** $j = \gamma$ **then**
11: **for** each SNP $S_i \in T_j$ **do**
12: $\eta_{j,s} \leftarrow \eta_{j,s} + \boldsymbol{\omega}_{i,n} \times \varepsilon_i$
13: **end for**
14: $\eta_j \leftarrow (\eta_{j,s} \times \rho) + \tau$
15: **for** $r = 1$ to 2 **do**
16: $\bar{\eta}_{r,j,c} \leftarrow 0$
17: **for** each clinical data $C_i \in$ subset $N_{r,j}$ **do**
18: $C_i, \bar{\varepsilon}_i \leftarrow$ *Parse*$(N_{r,j})$
19: $v_i \leftarrow$ *ReceiveValue*(C_i)
20: $\bar{\eta}_{r,j,c} \leftarrow \bar{\eta}_{r,j,c} + v_i \times \bar{\varepsilon}_i$
21: **end for**
22: $\bar{\eta}_{r,j} \leftarrow (\bar{\eta}_{r,j,c} \times \rho) + \tau$
23: **end for**
24: **else**
25: $\eta_j, \bar{\eta}_{1,j}, \bar{\eta}_{2,j} \leftarrow$ *Random*()
26: **end if**
27: $\bar{R}$*.append*("$\eta_j, \bar{\eta}_{1,j}, \bar{\eta}_{2,j}$:")
28: **else**
29: $\bar{R}$*.append*("authentication error :")
30: **end if**
31: **end for**
32: **return** $\bar{R}$

not. If $j = \gamma$, Line 12 multiplies $\boldsymbol{\omega}_{i,n}$ values with the scaled contribution factor, ε_i of each SNP S_i and adds up with $\eta_{j,s}$ to generate the total genetic score. Line 14 multiplies the total genetic score with the constant ρ and adds to the constant τ to generate scaled genetic score, η_j for the j^{th} disease. The n^{th} DDB at the patient's device saves ρ and τ for final computation of the disease risk. This scaling is done so that the MU cannot infer the genetic score even if the patient decides to share the final disease risk with the MU for the purpose of treatment.

Similar to the SNP sets, each clinical data C_i and its scaled contribution factor $\bar{\varepsilon}_i$ are parsed from the query message. The value, v_i of C_i is received from patient, P. Recall that $v_i \in \{0, 1\}$. In Line 20, each v_i is multiplied with the scaled contribution factor, $\bar{\varepsilon}_i$ and summed up to generate the total clinical score for the r^{th} subset of the j^{th} disease , $\bar{\eta}_{r,j,c}$. Similar to Line 14, Line 22 generates scaled clinical score $\bar{\eta}_{r,j}$ using the same constants ρ and τ. The n^{th} DDB saves the values of γ^{th} genetic score, $\eta_{\gamma,s}$ and clinical scores $\bar{\eta}_{r,\gamma,c}$ to check whether a dishonest MU has forged contribution factors to infer genomic or clinical data.

In Line 25, random values are generated as η_j and $\bar{\eta}_{r,j}$ for a dummy disease. This is done so that a dishonest MU cannot generate score for any disease except the target disease without patient's consent. Scaled genetic and clinical scores for all the l diseases are sent in the return message, $\bar{R}$ to the MU.

5) Aggregation at the MU: The MU finds the total genetic score η_j, and the clinical scores $\bar{\eta}_{r,j}$ corresponding to the r^{th} clinical data subset of the j^{th} disease from the return message, $\bar{R}$ sent by the n^{th} DDB, where $j \in \{1, \ldots, l\}, r \in \{1, 2\}$. Recall that the index value, γ and the scaling constants, δ for genetic score and $\bar{\delta}_r$ for clinical scores related to the target disease, X are saved at the MU during query processing. Thus, γ^{th} scores, $\eta_\gamma, \bar{\eta}_{r,\gamma}$ correspond to the target disease. The MU scales back η_γ and $\bar{\eta}_{r,\gamma}$ using the constants, δ and $\bar{\delta}_r$ respectively and generates $\bar{Z}$ by adding the results as follows,

$$\bar{Z} = \eta_\gamma \times \delta^{-1} + \bar{\eta}_{1,\gamma} \times \bar{\delta}_1^{-1} + \bar{\eta}_{2,\gamma} \times \bar{\delta}_2^{-1}$$

Next, MU adds inverse of the scaling constants to generate a value Δ such that, $\Delta = \delta^{-1} + \bar{\delta}_1^{-1} + \bar{\delta}_2^{-1}$. For final computation of the total disease risk, MU sends $\bar{Z}$ and Δ to the n^{th} DDB at the patient's device.

6) Final computation at the n^{th} DDB: After receiving $\bar{Z}$ and Δ from the MU, the n^{th} DDB generates the final score, Z using the previously saved constants ρ and τ such that,

$$Z = (\bar{Z} - \Delta \times \tau) \times \rho^{-1}$$

Final score, Z is used to compute the probability of the patient to develop target disease, X using Equation 1.

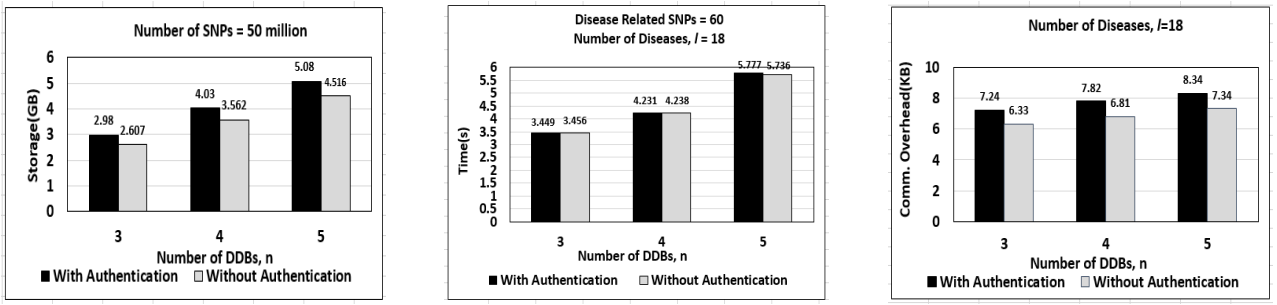
In the n^{th} DDB, value of Z is checked whether

$$Z = \frac{\eta_{\gamma,s}}{\Delta} \text{ or } Z = \frac{\bar{\eta}_{r,\gamma,c}}{\Delta} \text{ for } r \in \{1, 2\},$$

where $\eta_{\gamma,s}$ and $\bar{\eta}_{r,\gamma,c}$ are the genetic and clinical scores respectively, corresponding to the target disease, X and are saved in the n^{th} DDB (Section V-C4). If any of these values are equal to Z, the patient concludes that a dishonest MU has altered contribution factors to infer her SNP contents or clinical data and will not share the final score, Z with the MU.

VI. Results

In this section, we evaluate the effect of varying the privacy level for disease risk queries on the performance of our proposed system. The privacy level is expressed using the number of the DDBs (n), the number of diseases used in a query (l), and the total number of SNPs related to l diseases in the query. Values of these parameters are varied between ranges as n: 3-5, l: 5-25, and total number of SNPs: 50-75, where default values of n and l are 4 and 18, respectively. We use 0.3 million SNPs from a real SNP profile [2]. The relevant information, i.e., SNPs, their risk alleles, clinical data and contribution factors have been collected from [3]. We repeat every experiment for 100 disease risk queries and present the average result in terms of storage, computational and communication overhead. To represent the communication overhead independent of the used communication link, we measure the communication cost in terms of transferred data size among involved parties. We have



(a) (b) (c)

Fig. 2. Effect of n on (a) storage, (b) time and (c) communication overhead

performed experiment on our proposed system on Intel Core i5 CPUs with 2.7GHz processor under macOS using Eclipse 4.6 and MySQL database.

A. *Effect of* n

For evaluating authenticated disease risk query, each tuple of a DDB entry needs 8×8 (8 character pseudonym) + 8×10 (10 character SNP ID) + 2×2 (two 2 bit naïvely encoded alleles) + 2×8 (two 8 bit tiny integer weight of the two alleles) + 2×8 (two 8 bit tiny integer for authenticating value α of the two alleles) = 180 bits. Only exception is the patient's DDB for which each tuple needs $8\times10+4\times(2+8+8)$ i.e., 152 bits, as it does not have the pseudonym attribute but has both possible nucleotides and corresponding weights and α values for each of the two alleles. The patient's DDB also saves authentication key μ as an 8 bit tiny integer. In an unauthenticated system, there will be no authenticating value α in the DDBs. As such, each tuple of a DDB entry will need only 164 bits and each tuple in the patient's DDB will need only 120 bits. Thus to store 50 million SNPs, total storage is $(180(n-1)+152)\times 50\times10^6+8$ bits for an authenticated system, and $(164(n-1)+120)\times50\times10^6$ bits for an unauthenticated system. Again, in a system with no privacy measure, there are only two message transfers between the MU and a central data center for the computation of the genetic score. However, in a system with n number of DDBs, number of message transfers between the MU and the DDBs is $(2n+1)$ if the system is authenticated and $2n$ otherwise. As such the storage size and communication overhead increases linearly with the increase of n (Fig. 2a and Fig. 2c). We emphasize that the DDBs are linked to the MU with a parallel interface connections and all the DDBs compute partial genetic scores simultaneously. Hence, with the increase of n, the computational time is not affected significantly apart from the time needed for the connection setup and packet transfer. Fig. 2b shows that the computational time varies in ms range between authenticated and unauthenticated system.

B. *Effect of* l *and number of SNPs in a query*

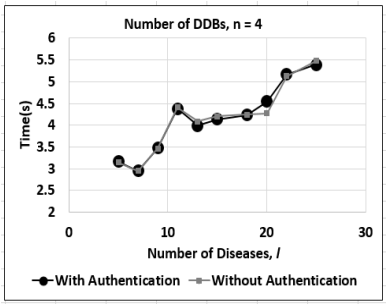
The time and number of bits needed to generate the query message at the MU and the return messages with partial genetic scores at the DDBs depend on the total number of SNPs that are subject to randomly chosen (l-1) diseases. Thus, with the increase of total number of SNPs, time and communication overhead increases linearly (Fig. 4a and Fig. 4b). However, Fig. 3a–3b show almost linear patterns with several peaks and valleys. The reason behind this behavior is that the number of SNPs related to a disease can vary in a large range. As such, smaller value of l may result in larger number of SNPs used in a query. Figures 3 and 4 show that communication overhead increases slightly in an authenticated system compared to an unauthenticated one and computational time remains almost same in both systems. Furthermore, we note that increasing l or total number of SNPs does not affect the storage size.

C. *Comparative Analysis*

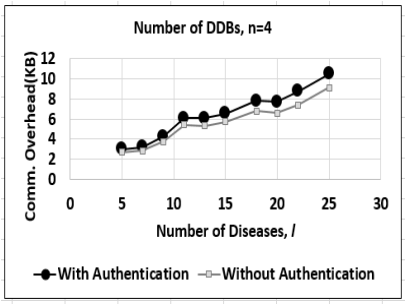
We have compared the performance of our system (authenticated and unauthenticated systems denoted as DA1 and DA2, respectively in graphs) with recent cryptographic approaches [8], [15], [12] (denoted as A1, A2 and A3, respectively in graphs). These approaches consider the effect of multiple SNPs on disease risk queries. However, none of these approaches authenticates the disease risk query.

1) Storage Overhead: In [8], two BCP ciphertexts (one for the SNP, other for its square) for approximately 50 million known SNPs are stored in encrypted form. Each BCP ciphertext is a pair of 4096-bit group elements. Thus, the total storage for 50 million SNPs is $2\times(50\times10^6)\times(2\times4096)$ bits, i.e., almost 100GB. In [15], all the 50 million SNPs are sent at once and the storage needed to encrypt all the SNPs takes $2\times(50\times10^6)\times(2\times193)$ bits, i.e., about 4.5GB. Both of these approaches store only the frequency of one allele in each SNP. If these approaches consider storing both alleles, the storage becomes double (Fig. 5a). The storage of [12] is similar to [8], as it follows the encryption method of [8]. On the contrary, the storage of our system depends on the number of DDBs, n. For $n=5$, which ensures a good privacy level, our authenticated and unauthenticated approaches require about 5.08GB and 4.516GB, respectively and the cost lies below [15] till $n\leq8$ and $n\leq9$, respectively to store 50 million SNPs.

2) Communication Overhead: For $n=5$, $l=18$ and the number of total SNPs related to l diseases = 68, communication overhead of our authenticated and unauthenticated systems are 8.34KB and 7.34KB, respectively. On the contrary, in [8], the data center needs to send two BCP ciphertexts for each SNP (one for the SNP, other for its square). If we consider these 68 SNPs, the communication overhead entails $2\times68\times(2\times4096)$ bits, i.e., 136KB which is significantly higher than the overhead incurred by our system (Fig. 5b). The

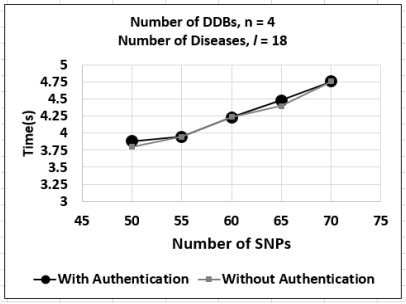


(a)

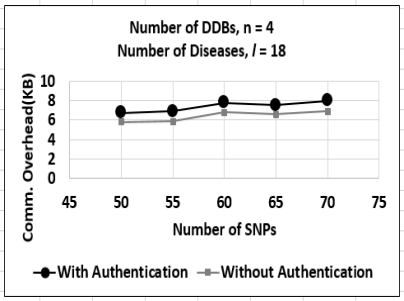


(b)

Fig. 3. Effect of l on (a) time and (b) communication overhead

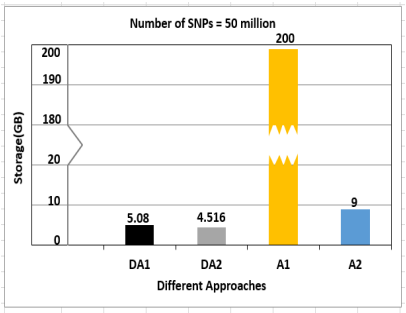


(a)

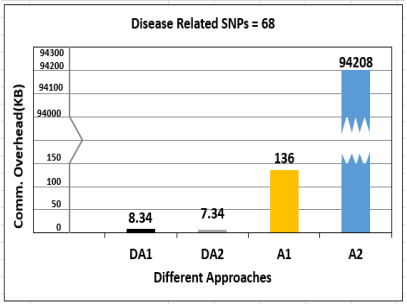


(b)

Fig. 4. Effect of number of SNPs on (a) time and (b) communication overhead



(a)



(b)

Fig. 5. Comparative analysis in terms of (a) storage and (b) communication overhead

approach proposed in [12] also incurs similar communication traffic as [8], since these two approaches use the same encryption method. Again, the approach in [15] always uses 1 million SNPs to hide the disease name for which its communication cost amounts to 92MB, which is extremely high.

VII. Conclusion

We introduced a novel secret sharing approach to evaluate privacy preserving authenticated disease risk queries that overcomes the limitations of existing approaches. Our approach can compute the probability of an individual to develop a disease when both the alleles of an SNP are responsible for two or more different diseases, and protect privacy of genome and clinical data even if the MU alters important parameters and colludes with the DDBs. Moreover, we ensure the correctness of the disease risk query by authenticating genomic data shared by the DDBs. Additionally, our approach protects the privacy of contribution factors, disease name, and the query answer. An important advantage of our approach is that the storage cost for SNPs is reduced significantly. Experiments show that our approach outperforms the existing approaches in terms of storage with a large margin. Furthermore, our approach provides a high level of privacy for a smaller value of n (i.e., 3) and incurs less computational and communication overheads.

References

[1] 23andMe, https://www.23andme.com/welcome
[2] ftp://ftp.ncbi.nih.gov/1000genomes/ftp/technical/reference/
[3] http://www.eupedia.com/genetics/medical_dna_test.shtml
[4] http://www.nature.com/scitable/definition/single-nucleotide-polymorphism-snp-295
[5] http://www.snpedia.com/index.php?title=Rs6313
[6] M. Akgün, A. O. Bayrak, B. Ozer, M. Ş. Sağıroğlu, "Privacy preserving processing of genomic data: A survey," Journal of Biomedical Informatics, 56, 2015, pp. 103-111.
[7] E. Ayday, J. L. Raisaro, J.-P. Hubaux, "Personal use of the genomic data: Privacy vs. storage cost," GLOBECOM, 2013, pp. 2723-2729.
[8] E. Ayday, J. L. Raisaro, J.-P. Hubaux, J. Rougemont, "Protecting and evaluating genomic privacy in medical tests and personalized medicine," WPES, 2013, pp. 95-106.

[9] E. Ayday, J. L. Raisaro, P. J. McLaren, J. Fellay, J.-P. Hubaux, "Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data," HealthTech, 2013.
[10] M. M. A. Aziz, M. Z. Hasan, N. Mohammed, D. Alhadidi, "Secure and Efficient Multiparty Computation on Genomic Data," IDEAS, 2016.
[11] P. Baldi, R. Baronio, E. D. Cristofaro, P. Gasti, G. Tsudik, "Countering GATTACA: Efficient and secure testing of fully-sequenced human genomes," CCS, 2011, pp. 691-702.
[12] L. Barman, M.-T. Elgraini, J. L. Raisaro, J.-P. Hubaux, E. Ayday, "Privacy Threats and Practical Solutions for Genetic Risk Tests," SPW, 2015, pp. 27-31.
[13] Y. Chen, B. Peng, X. Wang, H. Tang, "Large-scale privacy-preserving mapping of human genomic sequences on hybrid clouds," NDSS, 2012.
[14] E. D. Cristofaro, S. Faber, G. Tsudik, "Secure genomic testing with size- and position-hiding private substring matching," WPES, 2013, pp. 107-118.
[15] G. Danezis, E. D. Cristofaro, "Fast and Private Genomic Testing for Disease Susceptibility," WPES, 2014, pp. 31-34.
[16] J. Fowler, J. Settle, N. Christakis, "Correlated genotypes in friendship networks," National Academy of Sciences, 108 (5), 2011.
[17] M. Humbert, E. Ayday, J.-P. Hubaux and A. Telenti, "Addressing the Concerns of the Lacks Family: Quantification of Kin Genomic Privacy," CCS, 2013, pp. 338-347.
[18] L. Kamm, D. Bogdanov, S. Laur, J. Vilo, "A new way to protect privacy in large-scale genome-wide association studies," Bioinformatics, 29 (7), 2013, pp. 886-893.
[19] A. N. Mayer, D. P. Dimmock, M. J. Arca, D. P. Bick, J. W. Verbsky, E. A Worthey, H. J. Jacob, D. A. Margolis, "A timely arrival for genomic medicine," Genet Med, 13 (3), 2011, pp. 195-196.
[20] R. Merkle, M. Hellman, "On the security of multiple encryption," Communications of the ACM 24 (7), 1981, pp. 465-467.
[21] T. B. Pedersen, Y. Saygin, E. Savas, "Secret sharing vs. encryption-based techniques for privacy preserving data mining," Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality, 2007.
[22] M. Rotger, T. R. Glass, T. Junier, J. Lundgren, J. D. Neaton, E. S. Poloni, ..., P. E. Tarr, "Contribution of Genetic Background, Traditional Risk Factors, and HIV-Related Factors to Coronary Artery Disease Events in HIV-Positive Persons," Clinical Infectious Diseases, 57 (1), 2013.
[23] F. Turkmen, M. R. Asghar, Y. Demchenko, "iGenoPri: Privacy-Preserving Genomic Data Processing with Integrity and Correctness Proofs", PST(2016)
[24] S. Wandelt, M. Bux, U. Leser, "Trends in genomic compression," Current Bioinformatics, 2013.
[25] Y. Yang, D. M. Muzny, J. G Reid, M. N. Bainbridge, A. Willis, P. A. Ward, ..., C. M. Eng, "Clinical whole-exome sequencing for the diagnosis of mendelian disorders," New England Journal of Medicine, 369 (16), 2013, pp. 1502-1511.
[26] Y. Zhang, M. Blanton, G. Almashaqbeh, "Secure distributed genome analysis for GWAS and sequence comparison computation," BMC Medical Informatics and Decision Making, vol. 15 (5), 2015

APPENDIX

Theorem A.1 (Proof of Correctness)**.** *Let $\mathbb{P}(X)$ and $\mathbb{N}(X)$ be the sets of SNPs and clinical data related to a disease X, where $|\mathbb{P}(X)| = \lambda$ and $|\mathbb{N}(X)| = \phi$. For each SNP $S_i \in \mathbb{P}(X)$, β_i be the contribution factor of risk allele r_i and f_i be the total number of r_i in S_i. For each clinical data $C_i \in \mathbb{N}(X)$, $\bar{\beta}_i$ be the contribution factor and v_i be the value of C_i. Then the total score of a patient P for developing disease X is*

$$Z = \sum_{1 \le i \le \lambda} f_i \times \beta_i + \sum_{1 \le i \le \phi} v_i \times \bar{\beta}_i$$

Proof: Without loss of generality, we assume that each SNP set related to l different diseases sent by the MU to n DDBs has equal size λ. Recall that $\varepsilon_i = \beta_i \times c_j$ for the j^{th} disease, where $j \in \{1, \ldots, l\}$. Parameter $\boldsymbol{\omega}_{i,k}$ represents the total weight of the risk allele r_i of SNP S_i retrieved from the k^{th} DDB. Thus, the partial score, $s_{j,k}$ generated at the k^{th} DDB, is expressed as $s_{j,k} = \sum_{1 \le i \le \lambda} \boldsymbol{\omega}_{i,k} \times \varepsilon_i$. If authentication is successful at the n^{th} DDB, total genetic score $\eta_{j,s}$ is calculated as follows:

$$\eta_{j,s} = \sum_{1 \le k \le n-1} s_{j,k} + \sum_{1 \le i \le \lambda} \boldsymbol{\omega}_{i,n} \times \varepsilon_i = c_j \times \sum_{\substack{1 \le k \le n \\ 1 \le i \le \lambda}} \boldsymbol{\omega}_{i,k} \times \beta_i$$

Without loss of generality, we assume that each set of clinical data related to l different diseases has equal size ϕ and is partitioned into two subsets of equal size θ, i.e, $\phi = 2\theta$. For each clinical data C_i in the r^{th} subset of the j^{th} disease, $\bar{\varepsilon}_i = \bar{\beta}_i \times \bar{c}_{r,j}$, where $r \in \{1, 2\}$. The n^{th} DDB computes clinical score, $\bar{\eta}_{r,j,c}$ using the following equation:

$$\bar{\eta}_{r,j,c} = \sum_{1 \le i \le \theta} v_i \times \bar{\varepsilon}_i = \bar{c}_{r,j} \times \sum_{1 \le i \le \theta} v_i \times \bar{\beta}_i$$

The n^{th} DDB changes the genetic and clinical scores using constants ρ and τ such that $\eta_j = (\eta_{j,s} \times \rho) + \tau$ and $\bar{\eta}_{r,j} = (\bar{\eta}_{r,j,c} \times \rho) + \tau$. We note that $c_j = \delta$, $\bar{c}_{r,j} = \bar{\delta}_r$, and $\Delta = \delta^{-1} + \sum_{r=1,2} \bar{\delta}_r^{-1}$, where j^{th} disease is the target disease X. The MU calculates $\bar{Z}$ for $j = \gamma$ such that

$$\begin{aligned}
\bar{Z} &= \eta_\gamma \times \delta^{-1} + \sum_{r=1,2} \bar{\eta}_{r,\gamma} \times \bar{\delta}_r^{-1} \\
&= \frac{\eta_{\gamma,s} \times \rho}{\delta} + \frac{\tau}{\delta} + \sum_{r=1,2} \frac{\bar{\eta}_{r,\gamma,c} \times \rho}{\bar{\delta}_r} + \frac{\tau}{\bar{\delta}_r} \\
&= \rho \left(\frac{\eta_{\gamma,s}}{\delta} + \sum_{r=1,2} \frac{\bar{\eta}_{r,\gamma,c}}{\bar{\delta}_r} \right) + \tau \times \left(\frac{1}{\delta} + \frac{1}{\bar{\delta}_r} \right) \\
&= \rho \left(\sum_{\substack{1 \le k \le n \\ 1 \le i \le \lambda}} \boldsymbol{\omega}_{i,k} \times \beta_i \delta \times \frac{1}{\delta} + \sum_{\substack{r=1,2 \\ 1 \le i \le \theta}} v_i \times \bar{\beta}_i \bar{\delta}_r \times \frac{1}{\bar{\delta}_r} \right) + \tau\Delta \\
&= \rho \left(\sum_{1 \le i \le \lambda} f_i \times \beta_i + \sum_{1 \le i \le \phi} v_i \times \bar{\beta}_i \right) + \tau\Delta,
\end{aligned}$$

since we have $f_i = \sum_{1 \le k \le n} \boldsymbol{\omega}_{i,k}$ from Section V-C4. Finally, the n^{th} DDB calculates the total score, Z as follows:

$$Z = (\bar{Z} - \tau\Delta) \times \rho^{-1} = \sum_{1 \le i \le \lambda} f_i \times \beta_i + \sum_{1 \le i \le \phi} v_i \times \bar{\beta}_i$$

■

It is not possible to ensure the accuracy of disease risk queries if the MU uses inaccurate values of contribution factors for SNPs and clinical data. Hence, we focus on the integrity of SNP data for authentication purpose.

Theorem A.2 (Proof of Authentication)**.** *Let the number of DDBs be n. The n^{th} DDB can detect if the other $n-1$ DDBs or the MU alter SNP data used in a disease risk query.*

Proof: Each DDB stores weight w and a value α for each allele of an SNP. Let $\boldsymbol{\omega}_{i,k}$ and $\boldsymbol{\alpha}_{i,k}$ respectively be the total weight and the total value of α for risk allele r_i of SNP S_i in the k^{th} DDB. According to Section V-B, we have

$$\sum_{1 \le k \le n-1} \boldsymbol{\omega}_{i,k} = \mu \times \sum_{1 \le k \le n} \boldsymbol{\alpha}_{i,k}$$

If the DDBs change weight or α value for an SNP or the MU changes the sum of partial genetic and authenticating scores generated by $n-1$ DDBs arbitrarily, patient P can detect the changes, since authentication key μ is only known to P. ■