

# A Cryptographic Approach to Securely Share and Query Genomic Sequences

Murat Kantarcioglu, Wei Jiang, Ying Liu, and Bradley Malin, *Member, IEEE*

**Abstract**—To support large-scale biomedical research projects, organizations need to share person-specific genomic sequences without violating the privacy of their data subjects. In the past, organizations protected subjects' identities by removing identifiers, such as name and social security number; however, recent investigations illustrate that deidentified genomic data can be “re-identified” to named individuals using simple automated methods. In this paper, we present a novel cryptographic framework that enables organizations to support genomic data mining without disclosing the raw genomic sequences. Organizations contribute encrypted genomic sequence records into a centralized repository, where the administrator can perform queries, such as frequency counts, without decrypting the data. We evaluate the efficiency of our framework with existing databases of single nucleotide polymorphism (SNP) sequences and demonstrate that the time needed to complete count queries is feasible for real world applications. For example, our experiments indicate that a count query over 40 SNPs in a database of 5000 records can be completed in approximately 30 min with off-the-shelf technology. We further show that approximation strategies can be applied to significantly speed up query execution times with minimal loss in accuracy. The framework can be implemented on top of existing information and network technologies in biomedical environments.

**Index Terms**—Databases, genomics, homomorphic encryption, privacy, security.

## I. INTRODUCTION

THE PRACTICE of medicine is evolving toward personalized health care [1]. Already, findings from pharmacogenomic investigations indicate that variations in an individual's genotype influence the uptake and metabolism of pharmaceuticals [2], [3]. However, to realize cost-effective specialized services, scientists need to characterize the influence of genomic variation over a wide array of health features, such as clinical diagnostics and treatment response [4]. The integration of modern technologies into biomedical environments has enabled the collection of detailed genomic and clinical records [5], but the quantity of data necessary to conduct personalization studies is often beyond the capabilities of an individual researcher or institution [6]. As such, it is necessary for scientists to share private data collections in support of research on a larger scale. To facilitate data sharing, organizations in various countries, in-

cluding Estonia, Iceland, Japan, Mexico, Norway, Sweden, the United Kingdom, and the United States are establishing data repositories that centralize person-specific biomedical records for research purposes [7]–[9].

Despite the potential benefits to health care, person-specific genomic records must be shared in a manner that preserves the anonymity of the data subjects. This requirement is rooted in both social concerns and public policy. Many people fear that sensitive information learned from their medical and genomic records will be misused or abused [10], [11]. To mitigate such concerns, many countries have enacted policies that limit the sharing of a subject's genomic information in a personally identifiable form. In the United States, for instance, the National Institutes of Health (NIH) is drafting policy that will require scientists to share genomic data studied with NIH funding once “identifiable information” has been removed [12].

Consider the following scenario. Alice is a principle investigator located at the University of Texas Southwestern Medical Center and Bob is a principle investigator located at the Vanderbilt University Medical Center. Both Alice and Bob are independently funded by the NIH to collect data from hospital patients and conduct genome wide association studies on Alzheimer's disease. To comply with the NIH policy, at the completion of their studies, Alice and Bob need to share their data collections to a centralized repository, so that researchers around the country, such as Charlie at the National Institute on Aging can perform scientific investigations on the integrated data, such as “How many records contain a diagnosis of juvenile Alzheimer's and gene variant  $X$ ?” How can Alice and Bob share the biomedical records so that biomedical researchers can conduct their scientific investigations without revealing the identities of the data subjects? To summarize the problem, data collectors, such as Alice and Bob need to satisfy two goals when sharing genomic data:

- 1) *data utility: the data should be useful for scientific investigations;*
- 2) *data privacy: the data should not reveal the subjects' identities.*

Often, these goals are considered to be contradictory and existing privacy methods tend to favor one over the other. In this paper, however, we demonstrate that utility and privacy goals can be simultaneously satisfied for specific scientific endeavors.

### A. Genomic Data Privacy Techniques and Their Limitations

What is it about genomic data that makes it “identifiable”? To date, various privacy protection strategies have been designed to remove identifying information prior to sharing genomic data. For the most part, existing genomic data privacy techniques

Manuscript received February 15, 2007; revised June 30, 2007. First published June 10, 2008; current version published September 4, 2008.

M. Kantarcioglu and Y. Liu are with the Department of Computer Science, University of Texas, Dallas, TX 75080-3021 USA (e-mail: muratk@utdallas.edu; ying.liu@utdallas.edu).

W. Jiang is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907 USA (e-mail: wjiang@cs.purdue.edu).

B. Malin is with the Department of Biomedical Informatics, Vanderbilt University, Nashville, TN 37240 USA (e-mail: b.malin@vanderbilt.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TITB.2007.908465

can be roughly grouped into two approaches with distinct benefits and drawbacks: 1) *data deidentification* and 2) *data augmentation*.

Privacy protections based on “deidentification” advocate the removal, or encryption, of person-specific identifiers, such as name and social security number, initially associated with genomic records [13]–[17]. Deidentification enables data collectors to disclose all genomic information that has been collected, but it is an *ad hoc* process and provides no guarantees of privacy protection. In fact, it was recently shown that in many cases, knowledge gleaned from deidentified genomic data can be exploited to “reidentify” records to named subjects in publicly accessible resources through simple automated methods [18].

Data augmentation techniques provide exact guarantees of privacy protection. As an example, consider that a prime factor in reidentification is that a subject’s DNA is uniquely distinguishable. In particular, experimental evidence indicates that less than 75 single nucleotide polymorphisms (SNPs), features common to genomic studies, are sufficient to uniquely distinguish a subject’s DNA record in a population [19]. A formal model of privacy protection that addresses uniqueness is the generalization of a subject’s DNA sequence so that the resulting record is indistinguishable from other shared records [20], [21]. For instance, the DNA sequences *AACTAA* and *AAGTAC* can be generalized to the common *AA[C or G]TA[A or C]*. Privacy protection based on generalization is controlled by varying the number of records that are rendered indistinguishable. Though generalization formally prevents data reidentification, it changes the genomic records in ways that may limit their scientific usefulness.

### B. Contributions of This Research

In this paper, we propose an alternative approach to genomic data privacy protection that is based on cryptography. Our model ensures that: 1) the data utility of protected records is equivalent to that achieved by deidentification and 2) the data privacy is equivalent to that achieved by data augmentation schemes.

As an overview, our model works as follows. Data holders Alice and Bob transmit encrypted versions of their records to a third party’s data repository. The repository administrator executes queries on behalf of Charlie the researcher without decrypting any of the records. The results of the query are then sent to a third party who decrypts the aggregation of the result (i.e., How many records satisfied the query criteria?) and sends the answer to the scientist. This architecture incorporates two different third parties for security-related benefits. There is no opportunity to decrypt the data unless both third parties collaborate. As a result, the use of multiple third parties ensures that there is no single point of data compromise. Thus, if a hacker breaks into one of the third party’s computer systems, the hacker cannot learn the sensitive information in the encrypted records.

Recognize that though the data remains encrypted at all times, the results of queries themselves can violate privacy requirements. For instance, if the answer to Charlie’s query is such that there is only one record with DNA sequence “*AATCAATGAA*” and juvenile Alzheimer’s disease, then Charlie has uniquely

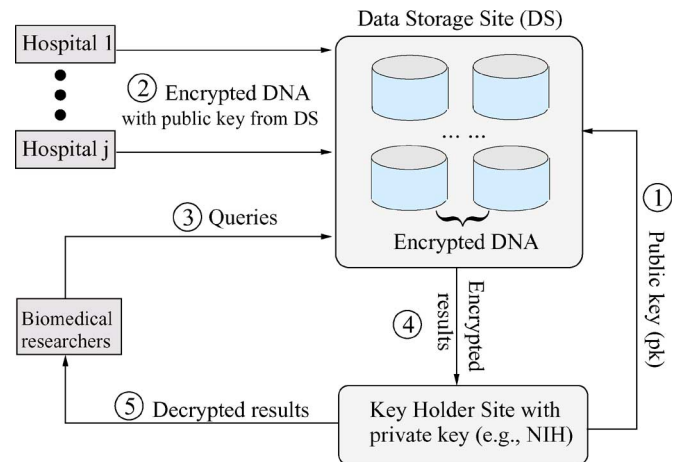


Fig. 1. General architecture of the proposed framework.

pinpointed an individual’s record. Thus, it is necessary for the third party to ensure that query results, or the combination of a series of query results issued by a researcher, do not permit the triangulation of an individual’s record. This process, known as query restriction, is necessary to ensure that our framework achieves identity protection; however, this topic has been studied extensively in the database security community [22], and thus, we neglect the presentation of query restriction in this paper.

The main contribution of our model is in the analysis of encrypted genomic data. To the best of our knowledge, there is no off-the-shelf product or literature that can be applied to satisfy this component of the framework. As such, this paper focuses on the cryptographic protocols that are necessary to build and query encrypted genomic databases. In addition, we provide experimental validation so that in our framework, queries can be answered efficiently for real world biomedical applications.

## II. METHODS

The goal of our research is to create a system that simultaneously: 1) stores DNA sequences in a database securely, 2) supports querying tasks that would be performed on the original sequences, 3) facilitates the DNA data holders to submit their records to our system without ever knowing the secret keys that can be used to decrypt the encrypted data, and 4) prevents a single point of failure to ensure that if a hacker breaks into any single site, he/she will not be able to learn the confidential DNA data. To achieve these goals, we designed an architecture that incorporates four types of participants: data holders, data users, a DS, and a KHS. In Fig. 1, we depict the relationship of these participants and a broad overview of the architecture.

For illustrative purposes, let us extend the scenario posed in Section I to correspond with the proposed framework. Imagine that the set of data holders are a set of hospitals (e.g., Vanderbilt University Medical Center and University of Texas Southwestern Medical Center) and that the set of data users are biomedical researchers (e.g., Charlie from the National Institute on Aging). For this research, we assume each hospital maintains one or more DNA records and that all hospitals collect

records on the same set of attributes (i.e., the same regions of the genome). Recall, in the earlier scenario, the data holders need to share the data with a third party for public dissemination purposes, which in the context of genome wide association studies in the United States will be the NIH. Yet, notice that in our framework, we incorporate two third parties: a data storage site (DS) and a key holder site (KHS). The additional third party is crucial to the security of the framework. The DS is where encrypted DNA is stored and processed, whereas KHS manages the cryptographic keys that are used for encryption and decryption of the genomic records stored in a database at DS, as well as the query results to biomedical researchers. Thus, if one of the third parties is compromised (e.g., a hacker breaks into the system), the decrypted DNA records are not revealed.

The distribution of the role of the third party provides additional benefits. In particular, notice that the majority of data management is pushed onto DS, whereas KHS serves as a final point of control in the system. Given the status of KHS, we recommend that the original third party, i.e., the NIH, plays the role of KHS. Now that we have mapped our participants from the original scenario to roles in the secure framework, the question remains, “Who plays the the role of DS?” This question can be qualified by recognizing that DS is constrained by several factors. First, the DS must be a trusted entity with no data of its own at stake, so that there is no conflict of interest. Second, the DS must have sufficient storage and bandwidth capacity in managing large databases with simultaneous access. We believe that this role can be assumed by a specialized information management company that is contractually bound to DS for liability purposes.

In the context of this paper, we assume that the participants are *noncolluding* and *semihonest*. By noncolluding, we mean that participants do not share information related to the protocol. Semihonest implies that all participants correctly follow the protocols, but they are free to use whatever information they see during the execution of the protocols in any way they choose. For a detailed description of the semihonest model in the context of formal security architectures, we refer the reader to [23]. We address the appropriateness of such assumptions in Section IV.

Before describing the details of the system architecture, we present several basic principles regarding the cryptographic protocols that we employ. For reference, Table I provides notations and abbreviations that we use throughout this paper.

#### A. Data Representation

Given a cryptographic basis, we need to define a mechanism by which genomic sequence data are encrypted. Since genomic sequence data are represented by the four letter alphabet of nucleotides  $\{A, C, G, T\}$ , each letter can be represented as a pair of bits and each genomic sequence can be represented as a series of binary values. For instance, Table II provides a mapping from a nucleotide alphabet to a two-bit binary value. Table III presents four DNA sequence samples (with a size of three nucleotides) in forms of the four letter alphabet and the corresponding binary representations, which are based on the mapping in Table II. Table IV shows how each record is encrypted using public

TABLE I  
COMMON NOTATIONS USED IN THE PAPER

$\theta^h = \{\theta_1^h, \dots, \theta_n^h\}$	A dataset of $n$ records in relational form, where each row indicates an individual's data and each column represents an attribute of the individual (e.g., a single nucleotide polymorphism (SNP)) at hospital $h$
$\{SNP_1, \dots, SNP_m\}$	the set of single nucleotide polymorphisms
$\theta^h[SNP']$	A projection of $\theta^h$ to the set of attributes contained in $SNP' \subseteq \{SNP_1, \dots, SNP_m\}$
$\theta_{ij}^h$	The value of attribute $SNP_j$ of the $i^{th}$ DNA sequence in $\theta^h$
$E_{pk}$	A public key based encryption function with public key $pk$
$D_{pk}$	The public key based decryption function related to $E_{pk}$ with private key $pr$
DS	Data storage site (i.e., location that maintains the encrypted genomic sequences)
HPE	Homomorphic Public-Key Encryption
KHS	Key holder site (i.e., manages the distribution of public keys and performs decryption of query results with private keys)

TABLE II  
ENCRYPTED DNA SEQUENCES: MAPPING

$A \rightarrow 00$
$C \rightarrow 01$
$G \rightarrow 10$
$T \rightarrow 11$

TABLE III  
ENCRYPTED DNA SEQUENCES: ORIGINAL DATA

$\theta_1^h$	$AAC$	000001
$\theta_2^h$	$AGT$	001011
$\theta_3^h$	$GTC$	101101
$\theta_4^h$	$GTC$	101101

TABLE IV  
ENCRYPTED DNA SEQUENCES: ENCRYPTING THE DATA

$E_{pk}(\theta_1^h)$	$E_{pk}(00)E_{pk}(00)E_{pk}(01)$
$E_{pk}(\theta_2^h)$	$E_{pk}(00)E_{pk}(10)E_{pk}(11)$
$E_{pk}(\theta_3^h)$	$E_{pk}(10)E_{pk}(11)E_{pk}(01)$
$E_{pk}(\theta_4^h)$	$E_{pk}(10)E_{pk}(11)E_{pk}(01)$

TABLE V  
ENCRYPTED DNA SEQUENCES: THE ENCRYPTED DATA

$E_{pk}(\theta_1^h)$	010011100100
$E_{pk}(\theta_2^h)$	101110100110
$E_{pk}(\theta_3^h)$	010111011100
$E_{pk}(\theta_4^h)$	110011100101

key encryption. Within each DNA sequence (e.g.,  $\theta_1^h$ ), each nucleotide is encrypted independently into a four-bit number, Table V indicates the encrypted DNA data.

#### B. Cryptographic Basics

To achieve a simple and flexible architecture, we utilize a “semantically secure” homomorphic public-key encryption (HPE) scheme. In an HPE scheme, each participant maintains a pair of cryptographic keys: a private key and a public key. Generally speaking, a participant keeps the private key secret and publicly publishes the public key. For example, if Alice wants to send a

message, or plaintext, to DS, Alice can encrypt the message into “ciphertext” using DS’s public key. The ciphertext can only be decrypted by the corresponding private key, so DS is the only entity that can decipher the message from Alice.

An encryption scheme is said to be semantically secure [24] if it is infeasible for an adversary, with finite computational capability, to extract information about a plaintext when in possession of the ciphertext and the corresponding public encryption key. The semantic security property implies that even the repeated encryption of the same message will be indistinguishable to an eavesdropper on the communication. In other words, if Alice and Bob both encrypt the same genomic sequence, say “GTC” ( $\theta_3^h$  and  $\theta_4^h$  in Table II) using DS’s public key, then the resulting ciphertexts  $E_{pk}(\theta_3^h)$  and  $E_{pk}(\theta_4^h)$  are different in binary format, e.g., “010111011100” is not equal to “110011100101.”

The HPE scheme we adopt in our architecture must be probabilistic and possess an additive homomorphic property. Informally, the additive homomorphic property allows us to compute the encrypted sum of two plaintext values through the corresponding ciphertext values. More formally, let  $E_{pk}(\cdot)$  denote the encryption function with public key  $pk$  and  $D_{pr}(\cdot)$  denote the decryption function with private key  $pr$ . An HPE scheme is probabilistic and additive homomorphic if the encryption function satisfies the following requirements.

- 1) *Constant multiplication*: Given a constant  $k$  and the encryption  $E_{pk}(m)$  of  $m$ , there exists an efficient algorithm to compute the public key encryption of  $km$ , denoted  $E_{pk}(km) := k \times_h E_{pk}(m)$  (here  $\times_h$  represents the multiplication operation of an encrypted value with a constant).
- 2) *Probabilistic*: If a message is encrypted twice with very high probability (almost 1), the two ciphertexts are different. For example, given a message  $m$ ,  $c_1 = E_{pk}(m)$  and  $c_2 = E_{pk}(m)$ , there is a high probability that  $c_1 \neq c_2$  and  $D_{pr}(c_1) = D_{pr}(c_2)$ .
- 3) *Additive homomorphic*: Given the encryptions  $E_{pk}(m_1)$  and  $E_{pk}(m_2)$  of  $m_1$  and  $m_2$ , there exists an efficient algorithm to compute the public key encryption of  $m_1 + m_2$ , denoted  $E_{pk}(m_1 + m_2) := E_{pk}(m_1) +_h E_{pk}(m_2)$  (here  $+_h$  represents the addition operation of two encrypted values).

The techniques we present in this paper can be applied within any additive HPE schemes, such as [25]–[28]. Note that RSA [29] is multiplicative homomorphic; as a result, it cannot be used in our framework. In addition, commonly known private key encryption schemes, such as [30] and [31], do not possess any homomorphic properties, so they are not applicable as well. Also, as we show in the next section, by using HPE systems, we do not need to decrypt the sensitive DNA data to answer certain queries. This is important because at any given time, if a hacker attacks any single site, he/she will not be able to learn the original sensitive DNA data. Unfortunately, this is not the case with private key encryption schemes such as [30] and [31]. These private key encryption schemes require the decryption of the encrypted data for query processing, which creates a potential vulnerability that hackers could exploit to learn the sensitive DNA values by attacking a single site. In our HPE framework,

we can prove that any attack that involves single site will not be successful in learning the sensitive DNA data. In this paper, we adopt the Paillier cryptosystem [28] for empirical analysis because it is efficient in comparison to other additive HPE systems. For completeness, we next provide a brief description of the homomorphic cryptosystem that we adopt for this paper. A Paillier cryptosystem that satisfies the aforementioned properties can be defined as follows.

- 1) *Key generation*: Let  $p$  and  $q$  be prime numbers where  $p < q$  and  $p$  does not divide  $q - 1$ . For the Paillier encryption scheme, we set the public key  $pk$  to  $n$  where  $n = pq$  and private key  $pr$  to  $(\lambda, n)$  where  $\lambda$  is the lowest common multiplier of  $p - 1, q - 1$ .
- 2) *Encryption with the public key*: Given  $n$ , the message  $m$ , and a random number  $r$  from 1 to  $n - 1$ , encryption of the message  $m$  is calculated as follows:  $E_{pk}(m) = (1 + n)^m r^n \bmod n^2$ .
- 3) *Decryption with the private key*: Given  $n$ , the ciphertext  $c = E_{pk}(m)$ , we calculate the  $D_{pr}(c)$  as follows:  $m = [((c^\lambda \bmod n^2) - 1)/n]\lambda^{-1} \bmod n$  where  $\lambda^{-1}$  is the inverse of  $\lambda$  in modulo  $n$ .
- 4) *Adding two ciphertexts ( $+_h$ )*: Given the encryption of  $m_1$  and  $m_2$ ,  $E_{pk}(m_1)$  and  $E_{pk}(m_2)$ , we calculate the  $E_{pk}(m_1 + m_2)$  as follows:

$$\begin{aligned} E_{pk}(m_1)E_{pk}(m_2) \bmod n^2 &= ((1 + n)^{m_1} r_1^{n_1}) \\ &\quad ((1 + n)^{m_2} r_2^{n_2}) \bmod n^2 \\ &= ((1 + n)^{m_1 + m_2} (r_1 r_2)^n) \\ &\quad \bmod n^2 \\ &= E_{pk}(m_1 + m_2). \end{aligned}$$

Note, due to the modular operation, ciphertext addition yields  $E_{pk}(m_1 + m_2 \bmod n)$ .

- 5) *Multiplying a ciphertext with a constant ( $\times_h$ )*: Given a constant  $k$  and the encryption of  $m_1$ ,  $E_{pk}(m_1)$ , we calculate  $k \times_h E_{pk}(m_1)$  as follows:

$$\begin{aligned} k \times_h E_{pk}(m_1) &= E_{pk}(m_1)^k \bmod n^2 \\ &= ((1 + n)^{m_1} r_1^{n_1})^k \bmod n^2 \\ &= (1 + n)^{km_1} r_1^{kn} \bmod n^2 \\ &= E_{pk}(km_1). \end{aligned}$$

### C. Security Framework

Here, we walk through the framework and describe how the cryptographic features are used to create and query a database of encrypted DNA sequences. Fig. 1 provides a high-level perspective of the process.

*Step 1 (Key generation)*: In the first step of the protocol, KHS provides DS with a public key.

*Step 2 (Data encryption)*: When Alice is ready to share her DNA sequences, DS sends Alice its public key. Alice then encrypts her genomic records using the public key and sends the results to DS. From a practical standpoint, we recommend that Alice assigns each

record a unique identifier for update purposes. Thus, if Alice wants to append information, or correct errors that exist in records stored at DS, she does not have to resend the entire data collection. Instead, she can communicate the new information to DS, who can then amend or replace the appropriate records. It is at DS where the encrypted data will be queried and mined by biomedical researchers. We assume that DS can validate the legitimacy of the encrypted genomic sequences from each of the data holders. Note, it is necessary to specify authentication and access control mechanisms so that only authorized data holders can send their data to DS. We recommend building our framework on top of existing authentication and access control mechanisms. Though necessary in application, these issues are beyond the scope of this paper and are addressed elsewhere. We refer the reader to [32] for general architectures and [33] and [34] for implementation examples in biomedical settings.

*Step 3 (Query issuance):* The set of queries that can be issued are known to Charlie, the biomedical researcher. After the data are encrypted and stored at DS, Charlie can send a query for the database to DS. In Section II-D, we define an example of the types of queries that can be issued.

*Step 4 (Query processing):* Based on the query received, the DS performs the requested computations and sends the intermediate encrypted results to KHS.

*Step 5 (Result decryption):* Using the private key, the KHS decrypts the result and sends it back to Charlie.

Since data stored at DS are semantically secure, the DS can learn the contents of the encrypted data only when in possession of the corresponding private key. Yet, the DS does not know the corresponding private key because KHS keeps it secret. The KHS only issues DS a public key. Therefore, the data stored at DS are inherently secure against DS and researchers, such as Charlie. Nonetheless, to ensure a secure protocol within the proposed architecture, we need to prevent the private data from leaking to KHS. We prove this with respect to queries and aggregate results in the following section. Also note that in our framework, all the necessary cryptographic operations could be achieved in the background in such a way that Alice, Bob, and Charlie do not need to know any cryptographic details.

#### D. Secure Count Queries

One of the most common tasks that genome-based researchers need to perform is to determine how many samples satisfy certain characteristics. For example, researchers are interested in learning if there exist associations between various SNPs in the DSP1 gene and an individual's diagnosis with Alzheimer's disease [35]. Similar SNP-disease association studies are becoming common in human genomics research [36], [37]. The architecture described earlier provides a framework for the integration

of databases from disparate data holders, so that biomedical researchers may conduct research investigations over databases of larger populations. Yet, from a data mining perspective, for a researcher to discover association rules, he needs to learn the frequency of each itemset, e.g., the combination of values over a set of SNPs. The support of an itemset (e.g.,  $\text{SNP}_1 = A \wedge \text{SNP}_2 = T \wedge \text{Alzheimer's Diagnosis} = \text{Positive}$ ) can be found by first counting the number of records the itemset occurs in, and then, normalizing this quantity by the total number of records in the database. Other data mining tools, such as Naive Bayes models, Bayesian networks, and decision trees can also be learned by calculating the frequencies of certain events.

Frequencies for standard data mining applications can be calculated through traditional count queries. Unfortunately, count queries were not designed to be executed over homomorphically encrypted data. So, how can a database answer a count query on encrypted values without decrypting the data? In this section, we describe a SECURE-COUNT protocol that securely calculates the frequency of user-specified patterns without decrypting the data stored by DS.

In many cases, the genomic data under investigation corresponds to a set of SNPs, each of which can be represented as a binary variable [38]. Without loss of generality, we assume the underlying genomic sequences consist only of SNPs, such that the database contains only binary values. Let us elaborate on the earlier example: Charlie wants to learn how many records at DS contain a particular combination of SNPs, such that  $\{\text{SNP}_{j_1} = A \wedge \dots \wedge \text{SNP}_{j_k} = T\}$  where  $j_1, \dots, j_k$  is an arbitrary subsequence of a DNA data. Recall, we mapped nucleotides to bits, so  $A = b_1$  and  $T = b_k$ . Thus, such queries are represented as:  $\text{SECURE-COUNT} \left( \sigma_{\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k} \right)$ . To evaluate the query, the DS needs to calculate if  $\{\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k\}$  is satisfied for each tuple without revealing  $\{\text{SNP}_{j_1} = A \wedge \dots \wedge \text{SNP}_{j_k} = T\}$ . To formalize the problem, let  $\theta_{i,j_1}^h$  be the value of attribute  $\text{SNP}_{j_1}$  for tuple  $i$  in the SNP sequence table  $\theta^h$ . Again without loss of generality, let us further assume that  $b_1 = b_2 = \dots = b_t = 1$  and  $b_{t+1} = b_{t+2} = \dots = b_k = 0$  in the aforementioned formula.

In our architecture, the DS only has access to the  $E_{pk}(\theta_{i,j}^h)$  values, i.e., the encrypted SNPs. Though homomorphic encryption enables the addition of two encryptions modulo  $n$ , for a large  $n$ , it is nontrivial to compute a Boolean formula [39]. To prevent unrealistic computation times, the DS can check if the selection condition is satisfied for a given encrypted genomic sequence by converting it to an algebraic equation that can be calculated using the fundamental properties of homomorphic encryption. In Appendix I, we prove that the selection conditions can be satisfied in an algebraic form. Using the results given in Appendix I, our secure count protocol can be divided into two major parts. Using the DS-Count protocol, the DS calculates the algebraic equations on the encrypted data. Similarly, using the KHS-Count protocol, the KHS can decrypt the results of the algebraic equations to calculate the final query result. The DS-Count and KHS-Count protocols leverage the following observations.

**Protocol 1 DS-COUNT**


---

**Require:** SECURE-COUNT  $\left(\sigma_{SNP_{j_1}=b_1 \wedge \dots \wedge SNP_{j_k}=b_k}(\theta^h)\right)$ ,  
 $E_{pk}(\theta^h)$

- 1: **for all**  $E_{pk}(\theta_i^h) \in E_{pk}(\theta^h)$  **do**
- 2:  $E_{pk}(S(i)_1^t) = E_{pk}(\theta_{i_{j_1}}^h) +_h \dots +_h E_{pk}(\theta_{i_{j_t}}^h)$ ;  
 {Homomorphic addition of the encrypted attributes}
- 3:  $E_{pk}(S(i)_{t+1}^k) = E_{pk}(\theta_{i_{j_{t+1}}}^h) +_h \dots +_h E_{pk}(\theta_{i_{j_k}}^h)$ ;  
 {Homomorphic addition of the encrypted Attributes}
- 4:  $R_i \leftarrow ((E_{pk}(S(i)_1^t +_h E_{pk}(-t)) \times_h r_{i1}) + E_{pk}(S(i)_{t+1}^k \times_h r_{i2}, \text{ for randomly chosen } r_{i1}, r_{i2}$ ;  
 {Calculation of the encrypted algebraic formula}
- 5: **end for**
- 6:  $R \leftarrow \pi(R_1, R_2, \dots, R_\alpha)$ , where  $\pi$  is a random permutation, and send  $R$  to KHS

---

- 1) If a selection condition is satisfied for  $\theta_i^h$ , then the sum of  $SNP_{j_1}$  to  $SNP_{j_t}$  must be equal to  $t$  and the sum of all other SNP values must be zero.
- 2) Given two random nonzero values  $r_1, r_2 \bmod n$ , if  $a_1 r_1 + a_2 r_2 = 0 \bmod n$ , then  $a_1 = 0$  and  $a_2 = 0$  with high probability (see Appendix I).

Using these observations, for each  $\theta_i^h$ , the DS calculates:

- 1) the sum of the encrypted  $SNP_{j_1}$  to  $SNP_{j_t}$  values minus  $t$  and 2) the sum of all the other encrypted SNP values using homomorphic encryption. The DS then multiplies each of the previous summations with some random values  $r_1$  and  $r_2$ . Using the second observation given earlier, the KHS can ascertain whether a selection condition is satisfied or not with high probability. At the same time, the KHS does not learn anything other than the final result because the DS randomly orders the algebraic equation results.

Protocol 1 details the algorithm by which DS and KHS can answer a count query issued by a biomedical researcher. In this protocol,  $S(i)_a^b$  corresponds to  $\sum_{v=a}^b \theta_{i_{j_v}}^h$ , which is the sum of the  $a$ th through the  $b$ th bits of the SNP sequence. First, the DS calculates the  $(S(i)_1^t - t) r_1 + S(i)_{t+1}^k r_2 \bmod n$  values for each SNP sequence using the homomorphic encryption properties. More specifically, in protocol 1, the DS first calculates the encrypted sum of all  $SNP_{j_1}$  to  $SNP_{j_t}$  values [i.e.,  $E_{pk}(S(i)_1^t)$ ] (line 2) and the encrypted sum of all  $SNP_{j_{t+1}}$  to  $SNP_{j_k}$  [i.e.,  $E_{pk}(S(i)_{t+1}^k)$ ] (line 3) for each SNP sequence using homomorphic encryption properties. Second, using randomly chosen  $r_{i1}$  and  $r_{i2}$  values, the DS calculates the required algebraic equation for the  $i$ th SNP sequence and set it to  $R_i$  (line 4). Finally, the DS sends all  $R_i$  values to KHS.

From the theorems in Appendix 1, we know that if  $D_{pr}(R_i) = 0$  [note that  $D_{pr}(R_i)$  is equal to the value of the algebraic equation], then  $i$ th SNP sequence satisfies the query. In protocol 2, the KHS basically checks whether  $D_{pr}(R_i) = 0$  and count the number of rows  $D_{pr}(R_i) = 0$ . In other words, the KHS increments the counter  $c$  when  $D_{pr}(R_i) = 0$ . Since all the  $R_i$  values are randomly permuted, the KHS will not learn which DNA sequence satisfies the given query.

**Protocol 2 KHS-COUNT**


---

**Require:**  $R$  from DS

- 1:  $c \leftarrow 0$ ;
- 2: **for all**  $R_i \in R$  **do**
- 3: **if**  $D_{pr}(R_i) == 0$  **then**
- 4:  $c \leftarrow c + 1$ ; {If decryption of the encrypted value is zero then increment c}
- 5: **end if**
- 6: **end for**
- 7: *return*  $c$  to the user

---

*E. Communication and Computation Complexity*

Let us assume there are  $\alpha$  encrypted genomic sequences stored at DS, the size of a query from a biomedical researcher is  $k$ , and  $s$  is the number of bits necessary to represent  $n$ . Since  $k$  is a much smaller value than a randomly chosen value  $r \in \{1, \dots, n\}$ , a single exponentiation with the exponent  $r$  is more expensive than  $k$  multiplications. Therefore, to characterize the upper bound of the computational complexity, we calculate the number of exponentiations required by the SECURE-COUNT protocol (protocol 1).

First, there are two exponentiations required for each encrypted genomic sequence, so the number of exponentiations for the SECURE-COUNT protocol is bounded by  $O(\alpha)$ , or the total number of encrypted genomic sequences. Second, the DS sends  $\alpha$  encrypted query results to KHS, and each encrypted value is at most  $s$ -bits long. Thus, the communication complexity of the SECURE-COUNT protocol is bounded by  $O(\alpha s)$  bits, or  $O$  ("the total number of DNA records" times "the length of the encrypted result").

## III. RESULTS

The prior section defined the framework and how queries are executed within the framework. In this section, we prove that the framework is both secure and handles queries efficiently for small datasets. For large datasets, we prove that the query run-time can be approximated with minimal information loss.

*A. Security Analysis*

In this research, we assume that the security of a DNA sequence is compromised if the DNA sequence is revealed, or inferred, by a participant, given the observed information. Formally, we define security from the perspective of secure multi-party computation (see Definition 3.1 given later). Let us orient this definition in the context of the SECURE-COUNT protocol.<sup>1</sup> Recognize that the result  $R$  of the query issued by Charlie, which KHS receives from DS, consists of encryptions of either 0's or random values. As a consequence, it can be proven that KHS can only learn the query result, i.e., the number of 0-encryptions, but nothing else regarding the encrypted data stored at DS.

<sup>1</sup>Details of the security definitions and underlying models can be found in [23].

*Definition 3.1 (Secure multiparty computation):* Let  $x_i$  be the input of party  $i$ ,  $\prod_i(f)$  be the set of messages that are received and sent by party  $i$  during the execution of the protocol  $f$ , and  $c$  be the result computed from  $f$ . The protocol that computes  $f$  is secure if  $\prod_i(f)$  can be simulated from  $\langle x_i, c \rangle$  and distribution of the simulated image is computationally indistinguishable from  $\prod_i(f)$ .

In the context of biomedical research, Definition 3.1 states that if we can simulate what a participant sees during the execution of the protocol using only the participant's input and the final result, then the participant could not have learned anything beyond what it already knew. In the case of the SECURE-COUNT protocol, the input for DS corresponds to encrypted SNPs and Charlie's query. The output for DS corresponds to the encrypted count query result. To show that DS learned nothing other than the final result, we will show that what DS has seen during the execution of the secure count protocol can be simulated by its input and the final output.

To formally prove the security of the SECURE-COUNT protocol, we adopt the simulation argument defined in Definition 3.1. Recognize that DS only sees the encrypted genomic sequences, query, and the encrypted query result. Therefore, what DS sees can be simulated in polynomial time. Now, we need to show that, from KHS's point of view, the execution image of the SECURE-COUNT protocol can be simulated, and the simulated execution image is computationally indistinguishable from the actual execution image. Protocol 3 provides such a simulator. Protocol 3 generates encrypted  $R'$  values based on public key ( $pk$ ), the domain size of ciphertexts ( $n$ ), and the total number of encrypted DNA sequences ( $\alpha$ ).

Let  $\Pi_S$  be the view produced from the simulator, then according to protocol 3,  $\Pi_S = R'$ . Let  $\Pi_R$  be the view during the actual execution of the SECURE-COUNT protocol, then corresponding to protocol 2,  $\Pi_R = R$ . Note that in the actual execution of the protocol,  $R$  is the set of encrypted algebraic formula results for each  $\theta_i^b$  and KHS receives  $R$  from DS. We will show that KHS does not learn anything other than the final result by proving that  $\Pi_S$  and  $\Pi_R$  are indistinguishable. In other words, we will show that the protocol execution can be simulated by only KHS's input and the output. To prove  $\Pi_S$  and  $\Pi_R$  are computationally indistinguishable, we first prove the following claim.

*Claim 3.1:* The distributions of  $R'$  and  $R$  are computationally indistinguishable.

*Proof:* Let  $\alpha$  be the number of encrypted genomic sequences, and without loss of generality, assume  $R = (R_1, \dots, R_\alpha)$  are identically distributed random variables drawn from some distribution  $F_n$  and  $R' = (R'_1, \dots, R'_\alpha)$  are identically distributed random variables drawn from some distribution  $F'_n$ . Because the encryption scheme  $E_{pk}$  is semantically secure,  $R_i$  and  $R'_i$  ( $1 \leq i \leq \alpha$ ) are computationally indistinguishable. In addition, both  $R_1, \dots, R_\alpha$  and  $R'_1, \dots, R'_\alpha$  are polynomial-time constructible (or can be produced in polynomial time). Therefore, based on the polynomial-time sampling theorem presented in [23],  $R_1, \dots, R_\alpha$  is computationally indistinguishable from  $R'_1, \dots, R'_\alpha$ . ■

---

### Protocol 3 Simulator of KHS

---

**Require:**  $pk, \alpha, n, c$ ;  $pk$ : the public key,  $\alpha = |R|$ ,  $n$ : the domain size of ciphertexts and  $c$ : the final result

- 1: **for**  $i = 1$  to  $\alpha$  **do**
  - 2:   if  $i \leq c$ ,  $R'_i \leftarrow E_{pk}(0)$ ;
  - 3:   otherwise,  $R'_i \leftarrow E_{pk}(r)$  for randomly chosen  $r \in \{1, \dots, n-1\}$ ;
  - 4: **end for**
  - 5:  $R' \leftarrow \pi(R'_1, R'_2, \dots, R'_\alpha)$ ,  $\pi$ : random permutation;
  - 6: *return*  $R'$
- 

The variables  $R$  and  $R'$  differentiate between  $\Pi_R$  from  $\Pi_S$ ; however, based on Claim 3.1,  $\Pi_S$  is computationally indistinguishable from  $\Pi_R$ . As a consequence, the SECURE-COUNT protocol satisfies Definition 3.1. This result implies that what KHS has seen during the execution of the secure count protocol could be simulated by its input and the final count query result. Therefore, the protocol execution does not provide any new knowledge to KHS that could not be inferred by the final result.

### B. Experimental Run-Time Analysis

Since the commencement of the Human Genome Project, researchers have reported great numbers of SNPs. The availability of quality SNP markers makes candidate-gene, candidate-region, and whole-genome association studies possible. Linkage disequilibrium (LD) techniques have been widely applied for developing high-quality SNP marker maps [40], [41]. When applied to disease-gene mapping, LD is evaluated through association analysis that requires the comparison of allele or haplotype frequencies between the affected (e.g., diagnosis of Alzheimer) and the control individuals (e.g., no diagnosis of Alzheimer). Toivonen *et al.* [38] proposed a data mining method for LD mapping called haplotype pattern mining (HPM).

We evaluate the efficiency and accuracy of HPM within our framework. Following the work of [38], we use a simulated SNP dataset that was applied in their evaluation of the HPM algorithm. The dataset was generated with the following characteristics. First, an isolated founder population with an initial size of 300 was grown to 100 000 individuals over a course of 500 simulated years. Each individual's sequence was assigned one pair of homologous chromosomes, the length of each was 100 cM. Marker loci were simulated with a density of 3 SNPs per 1 cM nucleotides. The frequency of each SNP allele was set to 0.5. The goal of the HPM algorithm is to determine, for a given threshold  $x$  and a set of patterns  $P$ , if  $\pm\chi^2(P) \geq x$  is true or not. Given the disease-associated chromosomes ( $A$ ) and control chromosomes ( $C$ ) that either match a given pattern ( $P$ ) or not ( $N$ ), then  $\pm\chi^2(P)$  is defined as

$$\frac{(\delta_{AP}\delta_{CN} - \delta_{AN}\delta_{CP})^2}{\delta_A\delta_C\delta_P\delta_N}\delta$$

where  $\delta_{ij}$  is the number of chromosomes with properties  $i$  and  $j$ ,  $\delta_i$  is the number of chromosomes with property  $i$ , and  $\delta$  is the total number of chromosomes. Since  $\pm\chi^2(P)$  is contingent only on relative frequencies, it can be calculated using count

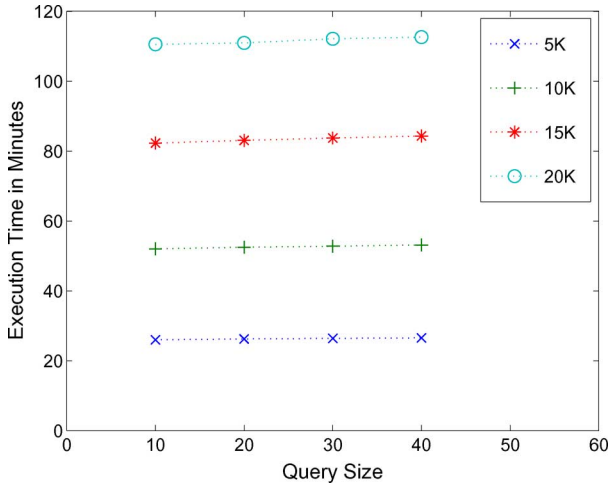


Fig. 2. Execution time for count queries on various datasets with different query sizes.

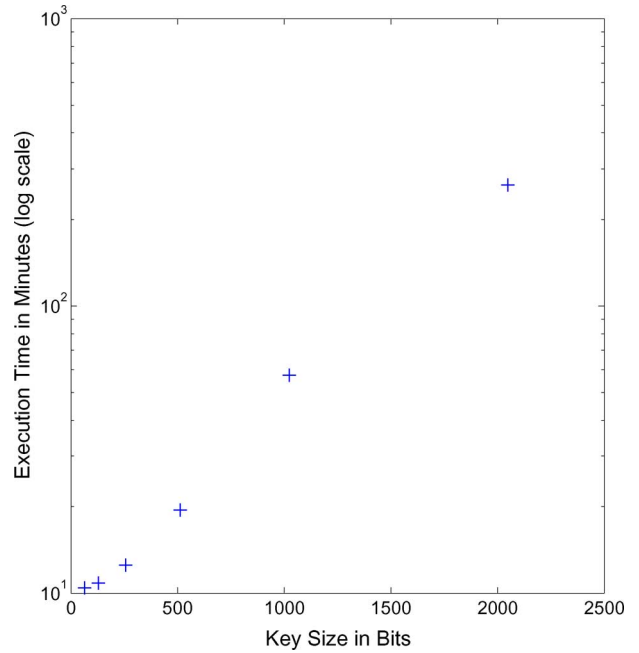
queries. To evaluate HPM in the context of a distributed environment, such as a set of hospitals, we combine a number of simulated datasets as used in [38], where each dataset contains 400 genomic sequence records, or chromosomes. Each record contains more than 300 SNPs represented in binary form.

We implemented our protocols in Java and ran our experiments on an off-the-shelf desktop with an Intel Pentium D 3.4 GHz processor with 2 GB memory. We used 1024 bit Paillier encryption for our experiments;  $n$  is 1024 bits long in our computations. In practice, we envision that there will exist a fast network connection between DS and KHS. Thus, to simplify our analysis, our implementation simulated DS and KHS on the same computer.

In our secure count query experiments, we used four binary datasets with 5000, 10 000, 15 000, and 20 000 tuples and four different query sizes that involve 10, 20, 30, and 40 binary attributes. Fig. 2 shows the execution time of each count query in minutes. For instance, in a database of 5000 records and a query that consists of 10 SNPs, the query will complete in approximately 25 min.

For the binary attributes, the SECURE-COUNT protocol requires only two exponentiations. At the same time, the number of attributes does not change the number of exponentiations. Rather, it only affects the number of homomorphic additions. Since each exponentiation is almost 1000 times more expensive than a homomorphic addition, a small increase in the number of attributes in the query does not significantly affect the execution time. As expected, execution time is linear in the number of tuples. Thus, when we increase our query to 40 SNPs on a database of 5000 records, the execution time increases to approximately 30 min. Unfortunately, the privacy protection provided by our architecture is not free. The same queries executed on the *unencrypted data* has running time that changes from 1 to 3 s. This increase in running time is due to expensive cryptographic operations.

The performance of the architecture is also influenced by the length of keys used for encryption and decryption. To investigate the effect of the key length, we repeated the experiments



be used to speed up a secure count query without sacrificing accuracy. Our results with a simple sampling strategy show that running count queries over 50 000 randomly chosen SNP sequences may be sufficient to precisely estimate the original count query result.

Let us define a random variable  $T_i^Q$  where  $T_i^Q = 1$  if the  $i$ th SNP sequence satisfies the selection criteria given in the query  $Q$  and 0 otherwise. Let us define  $f^Q = \sum_{i=1}^{\alpha} T_i^Q / \alpha$  to be the true frequency of the tuples that satisfies the selection criteria of the count query  $Q$ . In other words, the result of the count query  $Q$  is  $f^Q \alpha$ . Now let us calculate the result to our query by using random  $q$  SNP sequences chosen with replacement from the original  $\alpha$  sequences. Let  $\tilde{f}^Q$  be the estimated count  $f^Q$  using  $q$  randomly chosen SNP sequences. The Hoeffding inequality [43] implies Theorem 3.1, which bounds the probability where the difference between the query's true result and the approximated result is greater than a configurable parameter  $\epsilon$ .

*Theorem 3.1 [43]:* Given  $f^Q$  and  $\tilde{f}^Q$  calculated over  $q$  random samples, for any  $\epsilon > 0$

$$\Pr[|f^Q - \tilde{f}^Q| \geq \epsilon] \leq 2e^{-2\epsilon^2 q}.$$

To better understand the implications of the aforementioned theorem, consider the following example. Assume that we have a research database that has one million SNP sequences in it. Also assume that we choose  $q = 50\,000$  and  $\epsilon = 0.007$ . Furthermore, using the random 50 000 samples, assume that we observe 14 000 of the sampled sequences satisfy the query criteria. Based on the aforementioned observation, we estimate that the query is satisfied with frequency  $14\,000/50\,000 = 0.28$ . Using Theorem 3.1, we know that the true frequency is between 0.273 and 0.287 with probability close to 1. This means that the actual number of sequences that satisfy the query will be in the range  $0.273 \times 10^6 = 273\,000$  and  $0.287 \times 10^6 = 287\,000$  with very high probability.

This theorem is useful because it provides biomedical researchers and administrators the opportunity to weigh the costs and benefits of run-time versus error. For instance, in the previous example, when we set  $q = 50\,000$  SNP sequences and  $\epsilon = 0.007$ , we can calculate an upper bound of the probability that error is bigger than  $\epsilon$ . Fig. 4 shows the change in the upper bound probability for varying  $\epsilon$  values for  $q = 50\,000$ . As the figure indicates, for  $\epsilon$  values near 0.007, the upper bound on the probability that error is bigger than  $\epsilon$  approaches zero. This result indicates that, through sampling-based approaches, we can securely compute a highly accurate estimate of a count query's result in a reasonable amount of time.

#### IV. DISCUSSION

The proposed framework and the associated query illustrate how genomic data can be collected and queried in an encrypted manner. There are several limitations to the current implementation, however, which we now address.

##### A. Issues of Trust

In the proposed framework, we assumed that all participants are noncolluding and semihonest, such that they execute the

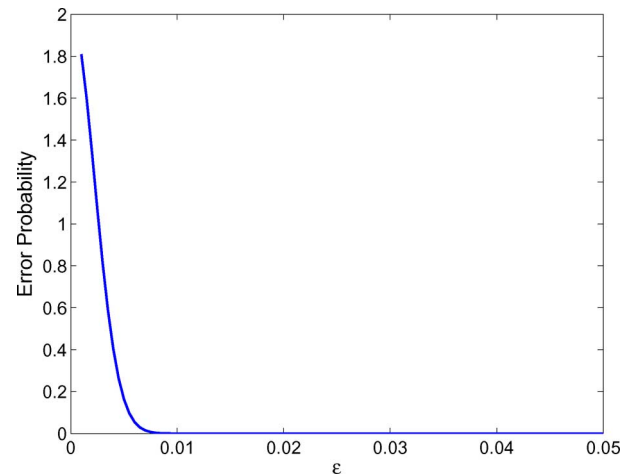


Fig. 4. Relationship between  $\epsilon$  (error) and upper bound on error probability.

protocol correctly, but may use what they observe to learn more than what they knew at the start of the protocol. In essence, collusion is only a problem when the DS colludes with the KHS. Thus, the trustworthiness of the system is completely dependent on our ability to trust the third parties. In biomedical and health care environments, a requirement of trust in third parties is not an unreasonable assumption. For instance, there are many real world applications where semihonest behavior is expected, such as daily administrative activities that take place between health care providers and insurance agencies providing reimbursement for patient care. In such cases, we assume that the insurance agency does not supply a patient's medical information to a nonprovider. In the context of genomic data privacy protection, third parties have been proposed in real world applications [14], [16].

Nevertheless, collusion between the DS and KHS can be prevented to a certain degree by utilizing "threshold decryption". The main idea behind this concept is that the private key can be distributed between  $n$  entities, and decryption can only be performed successfully when at least  $t$  out of these  $n$ , where  $1 \leq t \leq n$ , entities provide their portion of the key [44]. In other words, any subset of these entities, whose size is less than  $t$ , cannot decrypt the ciphertext that is encrypted via the corresponding public key. Therefore, collusion can only be achieved when there are more than  $t$  malicious entities who have shares of the private key. In general, the larger the values  $t$  and  $n$  are, the more difficult the collusion can succeed.

In addition, we recognize that the potential exists for participants in our architecture to deviate from the protocol to learn information. In the event that participants require more strict protections, we note that any semihonest protocol can be transformed to account for "malicious behavior" [23]. Yet, if a semihonest protocol is transformed into a malicious-resistant protocol using a generic model, the increased quantity of computation necessary to secure a protocol from malicious participants is often beyond what is acceptable for real world applications. A more reasonable and computationally feasible model of protection is not to prevent malicious behavior, but to detect when

such behavior has occurred, so that we may hold culprits accountable for their actions. Recent research has demonstrated that such models can be designed for simple data mining applications [45]. In the future, we anticipate applying such methods in our architecture.

### B. From Theory to Practice

This paper focused on the theoretical basis of a cryptographic framework for genomic data privacy. For this approach to be applied in the real world, it must be integrated into the wide variety of information technology infrastructures that are emerging in biomedical environments. One of the drawbacks of the theoretical presentation of our solution is that most biomedical sites have minimal experience in the integration of such a framework in their infrastructure. We believe that such a dilemma is relatively easy to overcome.

Though this paper adopted a theoretical approach, we have presented our research in a platform-independent nature. The primary reason for doing so is that we do not believe each site will need to redevelop the framework for their infrastructure. Rather, we are confident that our framework can be implemented on top of existing information infrastructures. As such, we believe that a generic implementation can be developed, in which existing infrastructures can set the appropriate inputs to the framework, and then, let the system run in the background.

## V. CONCLUSION

In this paper, we presented a cryptographic framework by which person-specific genomic sequence data can be stored and queried in an encrypted setting. In contrast to formal privacy models for genomic data that “perturb” or “generalize” records, our methods ensure that data are shared in its most specific state. We demonstrated that the architecture can support frequency counts without decrypting the genomic sequences. Beyond a theoretical basis, we experimentally validated that the architecture is efficient, in terms of time required for query processing, for real world applications. Though this research presented a secure framework, it does not address privacy violations that can be extracted from the query results. This can be handled through query restriction models, and we intend on addressing this issue in future work.

### APPENDIX I

#### ALGEBRAIC VERIFICATION OF A SELECTION CONDITION

In Section II-D, we claimed that DS can accurately compute the count result for a biomedical researcher’s query without decrypting the stored SNP sequences. In this section, we prove this claim and illustrate how DS can use algebraic properties of the homomorphic encryption scheme to verify if a particular sequence satisfies certain selection criteria. Our claims could be seen as a special case of the Schwartz–Zippel theorem applied to counting queries [46]. In the following theorems, let  $S(i)_a^b$  denote the sum  $\sum_{v=a}^b \theta_{i_v}^h$ . First, we show in Theorem 1.1 that if a given sequence  $\theta_i^h$  satisfies a certain query  $\{\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k\}$ , then the following algebraic

equation must be satisfied  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ , for randomly chosen  $r_1, r_2 \in \{1, \dots, n-1\}$  and for sufficiently large  $n$ .

*Theorem 1.1:* If  $\{\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k\}$  is satisfied for encrypted genomic sequence  $\theta_i^h$ , then  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ , for randomly chosen  $r_1, r_2 \in \{1, \dots, n-1\}$ .

*Proof:* Since each of the first  $t$  SNPs in the selection formula is equal to 1, summation of them must be  $t$ . Therefore,  $S(i)_1^t$  is equal to  $t$ . In addition, each of the last  $k-t$  SNPs is equal to 0; as a result,  $S(i)_{t+1}^k$  must be 0. This implies  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ . ■

Next, Theorem 1.2 indicates that, for randomly chosen  $r_1, r_2 \in \{1, \dots, n-1\}$ , the aforementioned algebraic equation is satisfied with probability at most  $1/(n-1)$  if  $\theta_i^h$  does not satisfy the query. In practice,  $n$  can be as large as  $2^{1024}$ . Therefore, in real world applications, the value  $1/(n-1)$  is negligible.

*Theorem 1.2:* If for any encrypted genomic sequence  $\theta_i^h$  does not satisfy  $\{\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k\}$ , then  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$  with probability at most  $1/(n-1)$  for randomly chosen  $r_1, r_2 \in \{1, \dots, n-1\}$ .

*Proof:* If  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ , this implies that either both  $(S(i)_1^t - t)$  and  $S(i)_{t+1}^k r_2$  are equal to zero or both are nonzero but  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ . Clearly, if both  $(S(i)_1^t - t)$  and  $S(i)_{t+1}^k$  are equal to zero, then  $\text{SNP}_{j_1} = b_1 \wedge \dots \wedge \text{SNP}_{j_k} = b_k$  is satisfied. Let  $a_1$  denote the  $(S(i)_1^t - t)$  and  $a_2$  denote the  $S(i)_{t+1}^k$ . Given  $(S(i)_1^t - t) \neq 0$  (i.e.,  $a_1 \neq 0$ ) and  $S(i)_{t+1}^k \neq 0$  (i.e.,  $a_2 \neq 0$ ), we can calculate the probability that  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$  for random  $r_1, r_2 \in [1, \dots, n-1]$  (next, we assume that all operations are done in mod  $n$  and let  $Y = a_1 r_1 + a_2 r_2$ ):

$$\begin{aligned} \Pr[Y = 0] &= \sum_{u=1}^{u=n-1} (\Pr[a_1 r_1 = -u \mid a_2 r_2 = u] \\ &\quad \Pr[a_2 r_2 = u]) \\ &= \sum_{u=1}^{u=n-1} (\Pr[a_1 r_1 = -u] \Pr[a_2 r_2 = u]) \\ &= \sum_{u=1}^{u=n-1} \left( \frac{1}{n-1} \times \frac{1}{n-1} \right) \\ &= \frac{1}{n-1}. \end{aligned}$$

This implies that if the query is not satisfied, our algebraic formula is equal to zero with probability  $1/(n-1)$ . ■

The aforementioned observation enables the calculation of count queries on the encrypted SNP data. Basically to check whether  $\theta_i^h$  satisfies a certain query, we need to check whether  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ , for randomly chosen  $r_1, r_2 \in \{1, \dots, n-1\}$ . If  $(S(i)_1^t - t)r_1 + S(i)_{t+1}^k r_2 = 0 \pmod n$ , this implies that  $\theta_i^h$  satisfies the query with at least probability  $1 - 1/(n-1)$ .

## REFERENCES

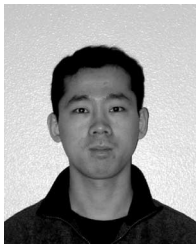
- [1] M. West, G. Ginsburg, A. Huang, and J. Nevins, "Embracing the complexity of genomic data for personalized medicine," *Genome Res.*, vol. 16, pp. 559–566, May 2006.
- [2] W. Evans and M. Relling, "Pharmacogenomics: Translating functional genomics into rational therapeutics," *Science*, vol. 286, pp. 487–491, 1999.
- [3] A. Roses, "Pharmacogenetics and pharmacogenomics in the discovery and development of medicines," *Nature*, vol. 38, pp. 815–818, 2000.
- [4] D. Roden, R. Altman, N. Benowitz, D. Flockhart, K. Giacomini, J. Johnson, R. Krauss, H. McLeod, M. Ratain, M. Relling, H. Ring, A. Shuldiner, R. Weinshilboum, and S. Weiss, "Pharmacogenomics: Challenges and opportunities," *Ann. Internal Med.*, vol. 145, pp. 749–757, 2006.
- [5] U. Sax and S. Schmidt, "Integration of genomic data in electronic health records—Opportunities and dilemmas," *Methods Inf. Med.*, vol. 44, pp. 546–550, 2005.
- [6] D. Gurwitz, J. Lunshof, and R. Altman, "A call for the creation of personalized medicine databases," *Nature Rev. Drug Discov.*, vol. 5, no. 1, pp. 23–26, 2006.
- [7] Anonymous, "Medicine's new central bankers," *The Economist*, vol. 377, no. 8456, pp. 28–30, Dec. 2005.
- [8] A. Engeland and A. Sogaard, "Conor (cohort norway)—En oversikt over en unik forskningsdatabank," *Norsk Epidemiologi*, vol. 13, pp. 73–77, 2003.
- [9] V. Barbour, "UK Biobank: A project in search of a protocol?," *Lancet*, vol. 361, pp. 1734–1738, 2003.
- [10] E. Clayton, "Ethical, legal, and social implications of genomic medicine," *New England J. Med.*, vol. 349, pp. 562–569, 2003.
- [11] M. Rothstein and P. Epps, "Ethical and legal implications of pharmacogenomics," *Nature Rev. Genetics*, vol. 2, pp. 228–231, 2001.
- [12] National Institutes of Health, "Request for information (RFI): Proposed policy for sharing of data obtained in NIH supported or conducted genome-wide association studies (GWAS)," National Institutes of Health, Bethesda, MD, no. NOT-OD-06-94, Aug. 2006.
- [13] L. Burnett, K. Barlow-Stewart, A. Proos, and H. Aizenberg, "The 'GeneTrustee': A universal identification system that ensures privacy and confidentiality for human genetic databases," *J. Law Med.*, vol. 10, no. 4, pp. 506–513, May 2003.
- [14] G. de Moor, B. Claerhout, and F. de Meyer, "Privacy enhancing techniques—The key to secure communication and management of clinical and genomic data," *Methods Inf. Med.*, vol. 42, no. 2, pp. 148–153, 2003.
- [15] D. Gaudet, S. Arsenault, C. Belanger, T. Hudson, P. Perron, M. Bernard, and P. Hamet, "Procedure to protect confidentiality of familial data in community genetics and genomic research," *Clin. Genetics*, vol. 55, pp. 259–264, 1999.
- [16] J. Gulcher, K. Kristjansson, H. Gudbjartsson, and K. Stefansson, "Protection of privacy by third-party encryption in genetic research in Iceland," *Eur. J. Human Genetics*, vol. 8, no. 10, pp. 739–42, 2000.
- [17] K. Hara, K. Ohe, T. Kadowaki, N. Kato, Y. Imai, K. Tokunaga, R. Nagai, and M. Omata, "Establishment of a method of anonymization of DNA samples in genetic research," *J. Human Genetics*, vol. 48, no. 6, pp. 327–330, 2003.
- [18] B. Malin, "An evaluation of the current state of genomic data privacy protection technology and a roadmap for the future," *J. Amer. Med. Inf. Assoc.*, vol. 12, no. 1, pp. 28–34, 2005.
- [19] Z. Lin, A. Owen, and R. Altman, "Genomic research and human subject privacy," *Science*, vol. 305, no. 5681, p. 183, 2004.
- [20] Z. Lin, M. Hewitt, and R. Altman, "Using binning to maintain confidentiality of medical data," in *Proc. Amer. Med. Inf. Assoc. Ann. Symp.*, San Antonio, TX, 2002, pp. 454–458.
- [21] B. Malin, "Protecting genomic sequence anonymity with generalization lattices," *Methods Inf. Med.*, vol. 44, no. 5, pp. 687–692, 2005.
- [22] N. R. Adam and J. C. Wortmann. (1989, Dec.). Security-control methods for statistical databases: A comparative study. *ACM Comput. Surveys* [Online], 21(4), pp. 515–556. Available: <http://doi.acm.org/10.1145/76894.76895>
- [23] O. Goldreich. (2004). General cryptographic protocols. *The Foundations of Cryptography*. Cambridge, U.K.: Cambridge Univ. Press [Online]. 2. Available: <http://www.wisdom.weizmann.ac.il/oded/PSBookFrag/prot.ps>
- [24] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Security*, vol. 28, pp. 270–299, 1984.
- [25] J. C. Benaloh, "Secret sharing homomorphisms: Keeping shares of a secret secret," in *Advances in Cryptography, CRYPTO'86: Proceedings* (Lecture Notes in Computer Science), vol. 263, A. Odlyzko, Ed. New York: Springer-Verlag, 1986, pp. 251–260.
- [26] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proc. 5th ACM Conf. Comput. Commun. Security*, San Francisco, CA ACM, 1998, pp. 59–66.
- [27] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *Advances in Cryptology—Eurocrypt '98* (Lecture Notes in Computer Science 1403). New York: Springer-Verlag, 1998, pp. 308–318.
- [28] P. Paillier, "Public key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology—Proceedings Eurocrypt '99* (Lecture Notes in Computer Science, no. 1592). New York: Springer-Verlag, 1999, pp. 223–238.
- [29] R. L. Rivest, A. Shamir L. Adleman. (1978). A method for obtaining digital signatures and public-key cryptosystems. *CACM* [Online], 21(2), pp. 120–126. Available: <http://doi.acm.org/10.1145/359340.359342>.
- [30] Data encryption standard (DES). (1988, Jan. 22). National Institutes of Standards and Technology, Tech. Rep. FIPS PUB 46-2 [Online]. Available: <http://www.itl.nist.gov/fipspubs/fip46-2.htm>
- [31] NIST. (2001). Advanced encryption standard (AES). National Institute of Standards and Technology, Tech. Rep. NIST Special Publication FIPS-197 [Online]. Available: <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.
- [32] B. Lampton, M. Abadi, M. Burrows, and E. Wobber, "Authentication in distributed systems: Theory and practice," *ACM Trans. Comput. Syst.*, vol. 10, pp. 265–310, 1992.
- [33] C. Georgiadis, I. Mavridis, and G. Pangalos, "Healthcare teams over the Internet: Programming a certificate-based approach," *Int. J. Med. Inf.*, vol. 70, pp. 161–171, 2003.
- [34] F. Wozak, T. Schabetsberger, and E. Ammenwerth, "End-to-end security in telemedical networks—A practical guideline," *Int. J. Med. Inf.*, vol. 76, pp. 484–490, 2007.
- [35] Y. Meng, C. Baldwin, A. Bowirrat, K. Waraska, R. Inzelberg, R. Friedland, and L. Farrer, "Association of polymorphisms in the angiotensin-converting enzyme gene with Alzheimer disease in an Israeli Arab community," *Amer. J. Human Genetics*, vol. 78, pp. 871–877, 2006.
- [36] B. Kirk, M. Feinsod, R. Favis, R. Kliman, and F. Barany, "Survey and summary: Single nucleotide polymorphism seeking long term association with complex disease," *Nucleic Acids Res.*, vol. 30, pp. 3295–3311, 2002.
- [37] F. de la Vega, A. Clark, A. Collins, and K. Kidd, "Design and analysis of genetic studies after the HapMap project," in *Proc. 12th Pacific Symp. Biocomput.*, 2006, pp. 451–453.
- [38] H. Toivonen, P. Onkamo, K. Vasko, V. Ollikainen, P. Sevon, H. Mannila, H. Herr, and J. Kere, "Data mining applied to linkage disequilibrium mapping," *Amer. J. Human Genetics*, vol. 67, pp. 133–145, 2000.
- [39] R. Cramer, I. Damgard, and J. B. Nielsen, "Efficient multiparty computation from homomorphic threshold cryptography," in *Proc. IACR Eurocrypt (EUROCRYPT 2001)*, pp. 280–300.
- [40] A. Ching, K. S. Caldwell, M. Jung, M. Dolan, O. S. H. Smith, S. Tingey, M. Morgante, and A. J. Rafalski, "SNP frequency, haplotype structure and linkage disequilibrium in elite maize inbred lines," *BMC Genet.*, vol. 3, 2002, Paper 19.
- [41] F. de la Vega, D. Dailey, J. Ziegler, J. Williams, D. Madden, and D. Gilbert, "New generation pharmacogenomic tools: A SNP linkage disequilibrium map, validated SNP assay resource, and high-throughput instrumentation system for large-scale genetic studies," *Biotechniques*, vol. 32, pp. 48–50, 2002.
- [42] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone (1996, Oct.). *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press [Online]. Available: <http://www.cacr.math.uwaterloo.ca/hac/>
- [43] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Statist. Assoc.*, vol. 58, pp. 13–30, 1963.
- [44] R. Cramer, I. Damgard, and J. B. Nielsen. (2001). Multiparty computation from threshold homomorphic encryption. *Lecture Notes in Computer Science* [Online], 2045 p. 280 Available: [citeseer.ist.psu.edu/article/cramer00multiparty.html](http://citeseer.ist.psu.edu/article/cramer00multiparty.html)
- [45] W. Jiang and C. Clifton, "Transforming semi-honest protocols to ensure accountability," in *Proc. 5th IEEE Int. Workshop Privacy Aspects Data Mining*, 2006, pp. 524–529.
- [46] R. Motwani and P. Raghavan, "Algebraic techniques," in *Randomized Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 1995.



**Murat Kantarcioglu** received the B.S. degree in computer engineering from the Middle East Technical University (METU), Ankara, Turkey, in 2000, and the M.S. and Ph.D. degrees in computer science from Purdue University, West Lafayette, IN, in 2002 and 2005, respectively.

He is currently an Assistant Professor of computer science at the University of Texas, Dallas. His current research interests include the intersection of privacy, security, data mining, and databases; security and privacy issues raised by data mining; distributed data mining techniques; security issues in databases; applied cryptography and secure multiparty computation techniques; use of data mining for intrusion and fraud detection.

Dr. Kantarcioglu is a member of the Association for Computing Machinery (ACM).



**Wei Jiang** received B.S. degrees in both computer science and mathematics from the University of Iowa, Iowa City, Iowa, in 2002, and the Master's and Ph.D. degrees in computer science from Purdue University, West Lafayette, IN, in 2004 and 2008, respectively.

He will be an assistant professor in the Department of Computer Science at Missouri University of Science and Technology, Columbia, MO. His current research interests include privacy-preserving data mining and integration, privacy issues in a federated

search environment, and text sanitization techniques.



**Ying Liu** received the B.S. degree in environmental biology from Nanjing University, Nanjing, China, in 1995, and the Master's degree in bioinformatics and computer science and the Ph.D. degree in computer science from Georgia Institute of Technology, Atlanta, in 2002 and 2005, respectively.

He is now a tenure-track Assistant Professor in the Department of Computer Science, Department of Molecular and Cell Biology, the University of Texas, Dallas, where he is engaged on text mining biomedical literature to discover gene-to-gene relationships.

His current research interests include bioinformatics, computational biology, data mining, text mining, and database system. He is the author or coauthor of more than 40 published peer-reviewed research papers in various journals and conferences. He is engaged in text mining of medical literature databases, creation of databases for biological applications, computational systems biology, and data mining for better understanding of genomic/proteomic and medical data.

Dr. Liu has been a Program Co-Chair/Conference Co-Chair and a Program Committee Member of several international conferences/workshops. He is a member of the IEEE Computer Society and the Association for Computing Machinery (ACM).



**Bradley Malin** (S'04–M'06) received the B.S. degree in molecular biology, the Master's degree in knowledge discovery and data mining, the second Master's degree in public policy and management, and the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburgh, PA, in 2000, 2002, 2003, and 2006, respectively.

He is currently an Assistant Professor of biomedical informatics in the School of Medicine, Vanderbilt University, Nashville, TN, with a secondary appointment in the Department of Electrical Engineering and

Computer Science, School of Engineering. He is the author or coauthor of numerous scientific articles on biomedical informatics, data mining, and data privacy. From 2004 to 2006, he was the Managing Editor of the *Journal of Privacy Technology* (JOPT). His current research interests include privacy in health and genetic databases, surveillance in electronic medical record systems, and model-based clinical information systems.

Dr. Malin is the recipient of several awards from the American and International Medical Informatics Associations for his research in DNA databases and privacy. He has chaired various workshops on privacy and data mining for the IEEE and the Association for Computing Machinery (ACM). He was the Guest Editor of a special issue of *Data and Knowledge Engineering* on selected work from the 2006 IEEE International Workshop on Privacy Aspects of Data Mining.