

A Privacy-Preserving Framework for Integrating Person-Specific Databases

Murat Kantarcioglu¹, Wei Jiang², and Bradley Malin³

¹ University of Texas at Dallas, Department of Computer Science
muratk@utdallas.edu

² Purdue University, Department of Computer Science
wjiang@cs.purdue.edu

³ Vanderbilt University, Department of Biomedical Informatics
b.malin@vanderbilt.edu

Abstract. Many organizations capture personal information, but the quantity of records needed to detect statistically significant patterns is often beyond the grasp of a single data collector. In the biomedical realm, this problem has pressed regulatory agencies to require funded investigators to share research-derived data to public repositories. The challenge; however, is that shared records must not reveal the identity of the subjects. In this paper, we extend a secure framework in which data holders contribute and query encrypted person-specific data stored on a third party's server. Specifically, we develop protocols that enable data holders to merge personal records, thus creating larger profiles and diminishing duplication. The repository administrator can merge records via encrypted identifiers without decrypting or inferring the contents of the joined records. Our model is more practical than prior secure join methods because each data holder needs only a single interaction with the central repository. We further present an extension to the protocol that permits the revelation of k -anonymous demographics, such that the administrator can perform joins more efficiently with the guarantee that each record can be linked to no less than k individuals in the population. We prove the privacy preserving features of our protocols and experimentally evaluate their efficiency in a real world Census dataset.

1 Introduction

The tensions between data sharing and data privacy are felt in many environments. In this paper, we focus on recent issues in the biomedical community, which illustrates real policy and technology challenges, but also opportunities for solutions. Consider, in the United States, the National Institutes of Health (NIH) expects the timely sharing of final research data from investigators receiving \$500,000 or more in any year of an NIH-funded grant [1]. The goal is to facilitate the dissemination of data generated with NIH funds for use by other researchers. More recently, the NIH recognized that the integration of information technology into healthcare, in combination with the decreasing cost of DNA sequencing and storage technologies, has enabled the collection of detailed

patient-specific health and genetic data [2]. Thus, in its policy for genome wide association studies (GWAS), the NIH specifies that all data derived through NIH-funded GWAS studies, whether it be DNA sequences or information derived from an individual's medical record, must be shared to an NIH-managed centralized repository [3]. These are noble endeavors, designed to facilitate information reuse, but they are challenging because, at the same time, investigators must ensure that the anonymity of their subjects is not compromised. Failure to do so will have adverse legal and social consequences that significantly harm public support of biomedical research. It is important to recognize this problem is not unique to the United States. Across the globe, organizations are working to integrate person-specific DNA and clinical information from disparate facilities in the hopes of generating statistically significant research results [4,5,6].

In earlier work, we began to address this challenge with the introduction of a privacy enhancing framework that permits data holders to submit and query biomedical data housed in a centralized repository managed by a third party [7]. The framework enables data holders to store person-specific data, such as DNA sequences, on a third party's server in an encrypted format. The cryptographic basis of the framework is homomorphic, which enables the third party to execute queries for researchers, such as count queries, without decrypting the records. For example, a researcher can ask "How many DNA sequence contain pattern X ?" and, while the third party will learn the result (i.e., the fraction of records that satisfy the criteria), it cannot learn which records satisfied the criteria.

The knowledge gained through count queries; however, does not support certain biomedical applications. For instance, in the healthcare realm, patients are mobile and their data can be collected by multiple locations, such as when a patient visits one hospital for primary care and a second hospital to participate in a clinical trial [8]. To facilitate robust biomedical investigations and prevent duplication of entries, it is beneficial to merge data that corresponds to the same patient. In traditional databases, such merges are achieved through joins on common attributes. The framework presented in [7]; however, is based on semantic security, so equivalent identifiers will appear different after encrypted. Therefore, we need to develop a new protocol to achieve join queries.

We exploit the fact that many organizations, such as healthcare providers, collect identifying information on their consumers. For instance, it is common for hospitals to use a patient's Social Security Number (SSN) and/or demographics for administrative purposes [9,10]. Such identifiers have been validated as excellent keys for data merging [11], but their disclosure is strictly prohibited by most organizations' policies and federal regulations. Despite restrictions, we can share SSNs, and other identifiers, in an encrypted manner for data merging purposes when the encryptions are semantically secure [12]. In this paper, we demonstrate how to join patient-specific identifiers within an existing encrypted framework. Moreover, by utilizing the concept of k -anonymity

[13,14], we propose an approach to speed up encrypted joins by revealing person-specific specific features to limit the number of potential joins that must be evaluated.

The rest of the paper is organized as follows: Section 2 highlights the existing work that is most related to the protocols proposed in this paper, including an overview of our secure framework. In Section 3, we develop a secure equi-join procedure within the context of the framework. Then, in Section 4 presents a more efficient protocol using data k -anonymization techniques. Section 5 provides an empirical analysis of the protocol with a real world Census dataset. Section 6 discusses some lessons learned, and Section 7 concludes the paper.

2 Related Work

To date, several generic privacy preserving data integration frameworks [15,16] have been proposed. These frameworks generally involve three basic privacy-preserving components: 1) schema matching, 2) joins, and 3) query processing. For the most part, research has been performed to address specific challenges in each component. For example, in [17], a cryptographic protocol for schema matching has been proposed. In addition, several equality joins have been proposed in the past for different settings [18,19,20,21]. Usually, these protocols involve expensive cryptographic operations and they are not directly scalable for large data sets. To enable more efficient solutions, hash-based noise addition techniques [22] and anonymization based approaches have been [23] proposed. Compared to previous work; however, the protocols we introduce in this paper require participants to have only single interaction. Also, in our work, we enable each data holder's records to be incrementally added to the central data repository.

In prior work, we introduced a framework to support integration and querying of a database of encrypted genomic sequences and affiliated patient-specific data [7]. The goal of the framework is to enable 1) the secure transfer and centralized storage of person-specific DNA sequences in a database and 2) the support queries and data mining tasks as they would be performed on the original sequences. To achieve these goals, the framework incorporates four types of participants: data holders, data users, a data site, and a key holder site. As a running example, imagine that the set of data holders are hospitals and that the set of data users are biomedical researchers. We assume each hospital maintains some demographic information (e.g., sex, age), clinical information (e.g., medical diagnosis), and DNA records. We further assume that the participants do not collude and are semi-honest [24], such that all participants can use information they observe to infer knowledge, but they do not deviate from the framework's specification. The data site (DS) and key holder site (KHS) are crucial to the security components of the architecture. Specifically, KHS manages the keys that encrypt patient information and queries and the keys to decrypt the query results. In contrast, the encrypted DNA and patient data is stored and processed at DS. We summarize the participants' roles and the framework in Appendix A.

3 Secure Queries and the Equi-Join

We focus on how to execute equi-join queries on the encrypted data stored at DS. Such queries are necessary for adding and integrating new datasets to the database already stored at DS. Here, we present a novel protocol to perform secure equality joins, termed as *Secure-Equi-join*, and applicable to non-interactive environments with independently encrypted attributes. The proposed protocol uses the Paillier cryptosystem (key properties are presented in Appendix B).

We adopt the following notation for this paper: $\theta^h = \{\theta_1^h, \dots, \theta_\alpha^h\}$ is dataset of α records in relational form, where each row (e.g., θ_i^h) indicates an individual's data. θ_{ij}^h represents the value of the j^{th} attribute of the i^{th} individual in θ^h . E_{pk} and D_{pr} respectively are the Paillier's encryption and decryption functions with public key pk and private key pr . $\theta^{h_1} \bowtie \theta^{h_2}$ indicates the join of datasets θ^{h_1} and θ^{h_2} on common attributes (e.g., encrypted primary key).

Protocols 1 and 2 depict the pseudo-code of *Secure-Equi-join* as executed by DS and KHS, respectively. We assume a patient's record in a database is associated with identifying attributes, such as Social Security Number (SSN) or various demographics. The *Secure-Equi-join* is initiated by a hospital to encrypt the tuples in its database, θ^{h_1} , which is then sent to DS. After receiving encrypted tuples from two hospitals, $E_{pk}(\theta^{h_1})$ and $E_{pk}(\theta^{h_2})$, DS calculates $\theta^{h_1} \bowtie \theta^{h_2}$.

To evaluate the equi-join, DS securely calculates if two encrypted records are equivalent. Without loss of generality, we assume the join is performed using attributes $j_1 \dots j_m$. Let θ_{ij}^h be the value of the j^{th} attribute of i^{th} tuple of θ^h . DS must inspect whether two encrypted tuples $E_{pk}(\theta_{i_1}^{h_1})$ and $E_{pk}(\theta_{i_2}^{h_2})$ match. Using the homomorphic properties of Paillier encryption, DS checks if $(\theta_{i_1 j_1}^{h_1} = \theta_{i_2 j_2}^{h_2}) \wedge \dots \wedge (\theta_{i_1 j_m}^{h_1} = \theta_{i_2 j_m}^{h_2})$ is true by checking if $M_{i_1, i_2} = \sum_{v=1}^m (\theta_{i_1 j_v}^{h_1} - \theta_{i_2 j_v}^{h_2}) \cdot r_v = 0 \pmod n$, where r_1, \dots, r_m are non-zero random values. DS calculates $E_{pk}(M_{i_1, i_2})$ on encrypted data via evaluating:

$$E_{pk}(M_{i_1, i_2}) = (+_h)_{v=1}^m \left[\left(E_{pk}(\theta_{i_1 j_v}^{h_1}) +_h (E_{pk}(\theta_{i_2 j_v}^{h_2}) \times_h (-1)) \right) \times_h r_v \right]$$

When the decrypted value of $E_{pk}(M_{i_1, i_2})$ is 0, then two records correspond to the same patient with high probability. The main reason behind this observation is the fact that if all the attributes match then for each v , $(\theta_{i_1 j_v}^{h_1} - \theta_{i_2 j_v}^{h_2}) = 0$, and then M_{i_1, i_2} is 0. As proven below, if any of the attributes fail to match then it is highly unlikely that M_{i_1, i_2} is 0.

3.1 Correctness of Equi-Join Protocol

Here, we prove that $M_{i_1, i_2} = 0$ gives the correct join result with high probability. We first derive a lemma that states the probability of computing a 0 through homomorphic addition, when there is at least one non-zero value, is very low.

Lemma 1. *Given fixed $a_1, \dots, a_m \in \{0, \dots, n-1\}$ with at least one non-zero a_j value and uniformly randomly chosen $r_1, \dots, r_m \in \{1, \dots, n-1\}$. Let $S_m = \sum_{i=1}^m a_i \cdot r_i \pmod n$, then $\Pr[S_m = 0] \leq \frac{1}{n-1}$.*

Algorithm 1. DS-Equi-Join

Require: Encrypted datasets $E_{pk}(\theta^{h_1})$ and $E_{pk}(\theta^{h_2})$; j_1, \dots, j_m are join attributes

- 1: **for all** $E_{pk}(\theta_{i_1}^{h_1}) \in E_{pk}(\theta^{h_1})$ **do**
- 2: **for all** $E_{pk}(\theta_{i_2}^{h_2}) \in E_{pk}(\theta^{h_2})$ **do**
- 3: **for** $v = 1$ **to** m **do**
- 4: $E_v \leftarrow \left(E_{pk}(\theta_{i_1 j_v}^{h_1}) +_h (E_{pk}(\theta_{i_2 j_v}^{h_2}) \times_h (-1)) \right) \times_h r_v$
- 5: **end for**
- 6: $E_{pk}(M_{i_1, i_2}) \leftarrow E_1 +_h E_2 +_h \dots +_h E_m$
- 7: **end for**
- 8: **end for**
- 9: Send all permuted $E_{pk}(M_{i_1, i_2})$ values to KHS

Algorithm 2. KHS-Equi-Join

Require: $E_{pk}(M_{i_1, i_2})$'s from DS

- 1: **for all** $E_{pk}(M_{i_1, i_2})$ **do**
- 2: **if** $D_{pr}(M_{i_1, i_2}) = 0$ **then**
- 3: (i_1, i_2) matches
- 4: **end if**
- 5: **end for**
- 6: Send all matching (i_1, i_2) pairs to DS

Proof. Let us assume a_j is not equal to zero (it exists due to the initial assumption) and all operations are modulo a large prime n .¹ Given any $x \in \{0, \dots, n - 1\}$, we can easily state the following inequality:

$$Pr[a_j \cdot r_j = -x] = Pr[r_j = -x \cdot (a_j)^{-1}] = \begin{cases} 0, & x = 0 \\ \frac{1}{n-1}, & \text{else} \end{cases} \leq \frac{1}{n-1}$$

Using the above inequality, we have: $Pr[S_m = 0] = \sum_{x=0}^{n-1} (Pr[a_j \cdot r_j = -x | S_m - a_j \cdot r_j = x] \cdot Pr[S_m - a_j \cdot r_j = x])$. Thus, for any x , $Pr[a_j \cdot r_j = -x] \leq \frac{1}{n-1}$,

$$Pr[S_m = 0] \leq \frac{1}{n-1} \left(\sum_{x=0}^{n-1} Pr[S_m - a_j \cdot r_j = x] \right) \tag{1}$$

Since all operations are modulo n , $S_m - a_j \cdot r_j$ will only takes values between $\{0, \dots, n - 1\}$, this implies that:

$$\left(\sum_{x=0}^{n-1} Pr[S_m - a_j \cdot r_j = x] \right) = 1 \tag{2}$$

¹ To uphold protocol security, we recommend choosing values of n , the modular base, on the order of 1024 bits. If, for instance, we join two tables with 10 million tuples each, the expected number of mismatches is significantly smaller than one (i.e., $\frac{10^7 \cdot 10^7}{2^{1024}-1}$). Thus, for any database with the less than 2^{512} tuples, the error introduced by our scheme can be made arbitrarily small by increasing size of n .

Equations (1) and (2) concludes our proof. □

Lemma 1 provides intuition regarding the general properties of homomorphic addition. In the context of our protocol, this lemma can be used to prove Theorem 1. Basically, assume that we use homomorphic encryption to subtract the two patients' values in the same attribute (e.g., date of birth). Then, if the values match, the homomorphic subtraction will be a random value, or a false non-match with very low probability.

Theorem 1. *Given two encrypted tuples $E_{pk}(\theta_{i_1}^{h_1})$ and $E_{pk}(\theta_{i_2}^{h_2})$, if $\theta_{i_1}^{h_1}$ and $\theta_{i_2}^{h_2}$ matches, then $M_{i_1,i_2} = 0$ (M_{i_1,i_2} is defined as above); on the other hand, if $M_{i_1,i_2} = 0$ then $\theta_{i_1}^{h_1}$ and $\theta_{i_2}^{h_2}$ matches with probability at least $1 - \frac{1}{n-1}$.*

Proof. Due to the definition of M_{i_1,i_2} , if $\theta_{i_1}^{h_1}$ and $\theta_{i_2}^{h_2}$ matches then for all v , $(\theta_{i_1 j_v}^{h_1} - \theta_{i_2 j_v}^{h_2}) = 0$. This implies that $M_{i_1,i_2} = 0$. Let us consider the case where $M_{i_1,i_2} = 0$ but $\theta_{i_1}^{h_1}$ and $\theta_{i_2}^{h_2}$ does not match. This implies for some non-zero $a_v = (\theta_{i_1 j_v}^{h_1} - \theta_{i_2 j_v}^{h_2})$ values, $\sum_{v=1}^m (a_v \cdot r_v) = 0 \pmod n$ for non-zero uniformly randomly chosen $r_v \in \{1, \dots, n-1\}$. According to Lemma 1, the probability of such an event is less than $\frac{1}{n-1}$. This implies that if $M_{i_1,i_2} = 0$ then two tuples match with probability bigger than $1 - \frac{1}{n-1}$. □

3.2 Security of the Equi-Join Protocol

The protocol is secure within the framework with respect to DS because it does not have access to the private keys. In addition, due to semi-honest model, we assume that DS follows the protocol and only asks KHS for the decryption for the properly constructed $E_{pk}(M_{i_1,i_2})$ values. Thus, we consider security with respect to KHS. Specifically, KHS observes only encrypted values of either 0, which corresponds to a match, or a random value, which corresponds to a non-match. Since the encryption scheme is semantically secure, KHS cannot learn anything regarding the corresponding values of the encrypted data.

3.3 Communication and Computational Cost

Assume *Secure-Equijoin* is performed using m attributes, and let $|\theta^{h_a}|$ indicate the number of tuples in θ^{h_a} . According to Protocol 1, for each tuple pair, we perform $2m - 1$ homomorphic additions, m modulo inverses and m homomorphic multiplications. Since each homomorphic multiplication is equivalent to an exponentiation, which is much more expensive than the other operations, we define the computational complexity in terms of the number of exponentiations. Each tuple pair requires m exponentiations and there are $|\theta^{h_1}| \cdot |\theta^{h_2}|$ such pairs; as a result, the number of exponentiations for the *Secure-Equijoin* protocol is bounded by $O(|\theta^{h_1}| \cdot |\theta^{h_2}| \cdot m)$.

For each tuple pair, DS sends the M_{i_1,i_2} value to KHS. Assuming an s -bit long n value, the communication complexity is bounded by $O(|\theta^{h_1}| \cdot |\theta^{h_2}| \cdot s)$.

4 k -Anonymity for Secure Equi-Join

The *Secure-Equijoin* protocol is impractical with large datasets because it requires testing each new and existing record as a potential join, such that the running time increases quadratically with the number of tuples. To overcome this limitation of the protocol, we propose a method that relaxes the semantically secure protections afforded by the homomorphic cryptosystem to anonymity sets of size k . We append non-encrypted patient-specific values (e.g., demographics) to encrypted data (e.g., DNA) in a manner that satisfies a formal privacy model. Specifically, hospitals disclose non-encrypted patient-specific data in a manner that satisfies k -anonymity (basic properties are presented in Appendix C).

4.1 k -Anonymity as Hash Key

In essence, k -anonymized values serve as hashed keys by which DS can partition encrypted tuples into buckets that are much smaller than the number of tuples in the database. In doing so, DS can perform the secure equi-join procedure on the homomorphically encrypted identifiers, such as SSNs, without testing every combination of tuples in the cross-product of the submitted databases. Moreover, by k -anonymizing the data, we ensure that every tuple in a bucket is linkable to no less than k patients. Thus, after joining encrypted values, DS is unable re-identify a tuple to less than k patients.

To implement this model, we assume the hospitals' databases contain a common set of quasi-identifying attributes, such as a patient's residential zip code and age. Each hospital encrypts all remaining attributes via the public key of DS. The hospitals then k -anonymize the quasi-identifying values of their datasets.

4.2 Joins with Equivalent Populations

First, we consider the case when all hospitals have data on the same population. In this scenario, each hospital k -anonymizes its dataset (with the same anonymization algorithm, k values and generalization schema) and submits the result to DS. When DS performs a join, it constructs buckets corresponding to each combination of k -anonymous values. For each bucket, DS executes the *Secure-Equijoin* protocol. At the completion of the protocol, every tuple from each location will be joined with data stored at DS.

Claim. The joined database resulting from *Secure-Equijoin* at DS is k -anonymous with respect to the attributes in the quasi-identifier.

Proof Sketch. The union of the joined quasi-identifying values is equivalent to the quasi-identifying values of any tuples involved in the join. Since there are k or more tuples in each bucket, after all tuples are joined with their corresponding partners, their quasi-identifying values do not change. Thus, the resulting database is k -anonymous. \square

4.3 Joins with Overlapping Populations

Next, we address how to join data when hospitals collect records on overlapping populations of patients. In this case, hospitals cannot k -anonymize their databases independently, as was performed in the prior section. If this occurs, the same patient's data can be k -anonymized in different ways at different hospitals. As a consequence, data that is joined at DS could violate the k -anonymity model. For instance, consider the record $\{25, 47906\}$ defined over the attributes *AGE* and *ZIP CODE* and the generalization hierarchies in Figure 3 (See Appendix C). This tuple could be k -anonymized to $\{[23, 45], 479**\}$ at one hospital and $\{25, 47***\}$ at another hospital. In each submitted database, the tuples are k -anonymous, however, after joining the encrypted values, DS can infer that the corresponding demographics must be $\{25, 479**\}$, which is more specific than both of the submitted tuples. If the number of tuples with the combination of these demographic values is less than k , then the join violates k -anonymity.

Algorithm 3. k -Equijoin

Require: k : anonymity threshold; V_1, \dots, V_m : a set of value generalization hierarchies

- 1: DS: Send $T[Q]$ to hospital h
 - 2: Hospital h :
 - (1) Compute $C \leftarrow \text{Get-Candidate}(T^h, T[Q], V_1, \dots, V_s)$
 - (2) Anonymize C based on $T[Q]$ and send C to DS
 - 3: DS: Compute $C' \leftarrow \text{Equi-Join}(C, T)$ and send C' to hospital h
 - 4: Hospital h :
 - (1) Compute $\Gamma \leftarrow (T^h - C) \cup C'$
 - (2) k -anonymize Γ and send it to DS
-

To prevent this inference leak, we present a protocol that enables hospitals to coordinate and, subsequently, ensure all data stored at DS satisfies the k -anonymity framework. We call this protocol *k-Equijoin* and its key steps are presented in Protocol 3. Before delving into the details of the protocol, we provide an informal overview. Let T represent the database stored at DS. We partition T into $T[Q]$ and $T[\hat{Q}]$. The first component, $T[Q]$, represents the projection of T on the quasi-identifier attributes. The second component, $T[\hat{Q}]$, represents the encrypted portion of T . Similarly, data at hospital h is represented as $T^h[Q]$ and $T^h[\hat{Q}]$. To initiate the protocol, h submits a request to DS to transfer its patient-specific records. At this point, we must consider two scenarios: 1) a base case in which h is the first hospital to submit data and 2) a general case in which h is not the first submitter. In the base case, DS has yet to receive data from any hospital, so h will k -anonymize the quasi-identifying attributes in its database and encrypt the remaining attributes. Then, h will send T^h to DS for storage. In the more general case, DS has already received and stored data from one or more hospitals. So, hospital h partitions his data into records that DS: 1) may have already received from other hospitals and 2) definitely has not received.

Hospital h will k -anonymize the first set of records in the same schema as they were submitted to DS by other hospitals. The second set of records, which we call Γ , can be k -anonymized by h without regard to records at DS because they are the first to be submitted. Thus, h generates and sends $\Gamma[Q]$ to DS.

Now, we present the protocol more formally. To produce consistent data, the degree of k -anonymization and the generalization algorithms are fixed for the execution of the protocol. In addition, we assume there exists a fixed set of value generalization hierarchies available to the hospitals. Without loss of generality, we assume that some data is already stored at DS. k -Equijoin utilizes a function called *Get-Candidate* to produce a set of data tuples in T^h whose projection on quasi-identifier attributes can be anonymized to some tuples in $T[Q]$. For example, let V_1 and V_2 be the value generalization hierarchies (VGHS) presented in Figure 3 (Appendix C), which correspond to the attributes *AGE* and *ZIP CODE*. Using set representation, let $t = \{[46, 90], 475**\}$ be one of the records in $T[Q]$. Based on the two VGHS and t , we compute a set γ such that any value in γ can be generalized to some value in t based on the given VGHS. Consequently, given t , $\gamma = \{50, 53, 70, 75, 80, 47500, 47535\}$. Suppose $t^h = \{50, 54339\}$ is one of the records in $T^h[Q]$. Since $t^h \not\subseteq \gamma$, t^h cannot be generalized to t . On the other hand, if $t^h = \{50, 47500\}$, then t^h can be generalized to t , and t^h is called a candidate for the future join process at DS. The set C contains all possible candidates computed according to $T[Q]$ and VGHS.

In Step 1 of k -Equijoin, DS sends the k -anonymous portion of the centralized database to hospital h .² Next, in Step 2, h computes the set of its tuples that could potentially join to tuples in the centralized database at DS. Then, h k -anonymizes his local database $T^h[Q]$ and sends the k -anonymized portion along with the corresponding encrypted portion to DS. Then, in Step 3, after DS receives C , DS will locate the tuples in C that can potentially join to its data using the *Secure-Equijoin* protocol. After this computation, DS notifies hospital h which tuples can definitely not be joined with existing records, denoted as the set C' . Finally, in Step 4, h k -anonymizes the remaining tuples with those in C' , and sends the k -anonymized tuples to DS.³

Claim. The database at DS after all hospitals execute k -Equijoin is k -anonymous with respect to the attributes in the quasi-identifier.

Proof Sketch. Tuples that can be joined successfully at DS do not violate k -anonymity. Since hospital h locally k -anonymizes the other data tuples (denoted as Γ in Algorithm 3), these tuples create new buckets at DS. Each of the new buckets contains at least k tuples and each tuple in the new buckets do not violate k -anonymity. Similarly, the extension of this protocol to all hospitals

² Note, before sending $T[Q]$, DS can eliminate all duplicates in $T[Q]$ to reduce communication costs at least by k times.

³ Note that a small number of tuples may not be k -anonymized. When this occurs, h will not send these tuples to DS. However, these data can be combined with future collected data. If the size of combined data is greater than k , h can initiate k -Equijoin protocol again with DS.

preserves the k -anonymity criteria. When multiple locations submit their data to DS, the k -*Equijoin* protocol is executed sequentially; i.e., hospital 1 executes the protocol, followed by hospital 2, and so on. Each execution of the k -*Equijoin* protocol with a new hospital preserves the k -anonymity of the database at DS. Since each execution of the protocol preserves the k -anonymity criteria, the final database at DS must be k -anonymous as well. This holds for all hospitals and the database at DS that results from the sequential execution of k -*Equijoin* must satisfy k -anonymity. \square

We hypothesize that k -*Equijoin* will reduce the number of cryptographic match evaluations that DS must perform in comparison to the *Secure-Equijoin* protocol because tuples corresponding to the same patient must reside in the same bucket. We experimentally investigate this hypothesis below.

5 Experiments

To evaluate the proposed protocols, we selected the U.S. Census income dataset, which is publicly available on the UC Irvine Machine Learning Repository [25]. This dataset contains person-specific records that were extracted from the 1994 and 1995 Current Population Surveys. It contains 286,775 tuples without missing values. There are 40 demographic and employment-related attributes.

5.1 Secure-Equijoin

We prototyped our protocols in Java and executed the secure join query experiments using relations of 100, 200, 300, and 400 tuples extracted from the Census income dataset. Please note that for relations of size 100, we need to do compare 10000 tuple pairs. Similarly for data set size 400, we need to compare 160000 tuple pairs. For simplicity, we executed join queries of the form $\theta^{h_1} \bowtie \theta^{h_2}$ with different numbers of attributes in the equi-join criteria. The experimental results are summarized in Figure 1 and indicate that join queries are computationally expensive and thus very time consuming. As expected, the running time of the join protocol increases linearly with the number of attributes in the join and quadratically with the size of the relation. For instance, it took around an hour to compute an integration of two datasets with 100 patients each (i.e., a join operation that involves 10000 tuple pair comparisons) across four attributes.

From a practical perspective, we investigated the degree to which specialized software implementations could decrease the time necessary to complete secure equi-joins. We simulated the homomorphic encryption-decryption function in the C programming language with the GMP library. Our results indicate that we can achieve an order of 10 speed-up. This implies that we can complete two million tuple pair comparisons in less than a day, which may be an acceptable amount of time for some biomedical research queries. Yet, as the size of the database grows, the savings afforded by specialized code is significantly outpaced by the increased time required to evaluate possible tuple pairs in the homomorphic space. Thus, scaling the basic join protocol to large datasets is not

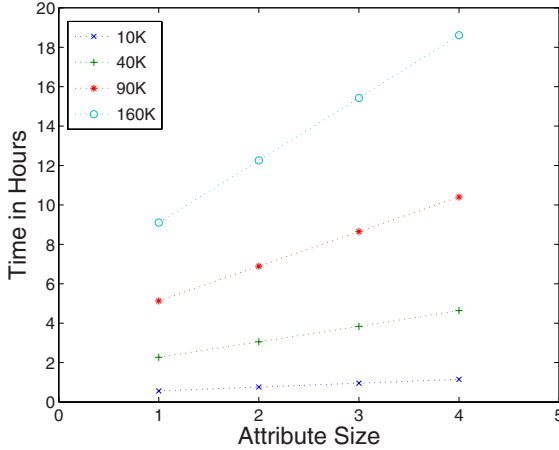


Fig. 1. Execution Time for Join Queries

feasible for the large databases that will be employed in biomedical data mining endeavors.

5.2 *k*-Equijoin

To evaluate the effect of *k*-anonymous demographics on secure joins, we applied the Datafly [26] algorithm, which generates *k*-anonymous datasets through generalization hierarchies using a greedy heuristic. We selected 4 of the 40 attributes in the dataset to represent the quasi-identifier (i.e., the generalizable attributes). Table 2 summarizes the statistics regarding the *k*-anonymous income dataset with *k* equal to 5, 10, 15, and 20. To orient the reader with respect to the results in this table, let θ^{h_1} refer to the income dataset and θ^{h_2} refer to a particular hospital’s dataset. Table 2 can be interpreted as follows: given $k = 20$, $\forall \theta_{i_2}^{h_2} \in \theta^{h_2}$, the expected number of tuples in θ^{h_1} that potentially match θ^{h_2} is 196. In other words, when the quasi-identifying attributes of θ^{h_2} is 20-anonymous, the average number of exponentiations is bounded by $196 \cdot |\theta^{h_2}|$ instead of $|\theta^{h_1}| \cdot |\theta^{h_2}|$, which occurs when we do not apply *k*-anonymity. This result implies that for $|\theta^{h_1}| = 286,775$ and $|\theta^{h_2}| = 1000$, by applying *k*-anonymity, we can reduce the number of exponentiations needed from $286,775,000 = 286,775 \cdot 1000$ to $196,000 = 196 \cdot 1000$. We have increased efficiency by almost 1500! Thus, we have confirmed our hypothesis that when *k* is not large, the number of secure equality checks required by the equi-join protocol can be reduced drastically.

6 Discussion

The limiting factor in the applicability of our join protocols is the computational power needed for exponentiations and the bandwidth necessary for

Table 1. Census Dataset Description

Attribute	Values	VGH Height
Age	91	5
Marital Status	7	3
Race	5	2
Sex	2	2

Table 2. Anonymization Statistics

k	Min	Max	Avg	Med	Std
5	5	1964	122	19	313
10	10	1964	150	30	341
15	15	1964	174	41	363
20	20	1964	196	51	379

communication between the data site (DS) and key holder site (KHS). We believe that our protocols will be more efficient when implemented in secure computer hardware. Here, we suggest several potential hardware-based improvements.

First, significant efficiency gains for our protocols can be achieved through cryptography accelerators that are tailored to execute expensive exponentiation operations. Based on reported results with hardware accelerators, the combination of more efficient software implementations (e.g., in the GMP library of the C language) with hardware accelerators could substantially decrease the time needed to complete an exponentiation in comparison to our Java-based experiments. This implies that secure joins of relations, without the use of k -anonymous keys, on databases of 10,000 could be achieved in less than a day. We leave the implementation of our algorithms using crypto accelerators as a future work.

Second, we can decrease the communication cost by co-locating KHS and DS. Specifically, we envision a system in which the functions of the KHS are performed by a secure co-processor that resides on the same server as DS. A secure co-processor is a single-board computer consisting of a CPU, memory and special-purpose cryptographic hardware contained in a tamper-resistant shell; certified to level 4 under FIPS PUB 140-1 (One example of such a secure co-processor is the IBM 4758 Cryptographic co-processor [27]). When installed on the server, it is capable of performing local computations that are completely hidden from the server. If tamper is detected, the secure co-processor clears the internal memory. The implementation of KS functionality through a secure co-processor on the same machine as DS will decrease the communication cost.

7 Conclusions

In this paper, we presented a framework by which person-specific biomedical data can be stored and queried in a centralized encrypted repository. We demonstrated that the administrator of the repository can perform joins of encrypted databases without decrypting or inferring the contents of the joined records. Furthermore, we presented an efficient extension to the join protocol that reveals patient-specific demographics in a manner that satisfies a formal privacy model, i.e., k -anonymity. In doing so, we allow the administrator to perform efficient joins with the guarantee that each record can be linked to no less than k patients in the population. This research is notable in that it demonstrates how centralized biomedical data repositories can be integrated and

updated with data distributed healthcare organizations without violating privacy regulations. In future research, we intend to implement this research in real world settings and extend it to secure computer architectures, such as secure coprocessors.

References

1. National Institutes of Health: Final NIH statement on sharing research data. NOT-OD-03-032 (2003)
2. National Institutes of Health: Genome-wide studies in biorepositories with electronic medical record data. RFA-HG-07-05 (2007)
3. National Institutes of Health: Policy for sharing of data obtained in nih supported or conducted genome-wide association studies. NOT-OD-07-88 (2007)
4. Benkner, S., Berti, G., Engelbrecht, G., Fingberg, J., Kohring, G., Middleton, S., Schmidt, R.: Gemss: grid-infrastructure for medical service provision. *Methods of Information in Medicine* 44, 177–181 (2005)
5. Anonymous: Medicine's new central bankers. *The Economist* (2005)
6. Barbour, V.: UK Biobank: a project in search of a protocol? *Lancet* 361, 1734–1738 (2003)
7. Kantarcioglu, M., Jiang, W., Liu, Y., Malin, B.: A cryptographic approach to securely share and query genomic sequences. *IEEE Transactions on Information Technology in Biomedicine* (in press, 2008)
8. Malin, B., Sweeney, L.: How (not) to protect genomic data privacy in a distributed network: using trail re-identification to evaluate and design anonymity protection systems. *Journal of Biomedical Informatics* 37, 179–192
9. Helliker, K.: A new medical worry: identity thieves find ways to target hospital patients. *Wall Street Journal* (2005)
10. Quantin, C., Allaert, F., Avillach, P., Fassa, M., Riandey, B., Trouessin, G., Cohen, O.: Building application-related patient identifiers: what solution for a european country? *Int. J. Telemed Appl.*, 678302 (2008)
11. Grannis, S., Overhage, J., McDonald, C.: Analysis of identifier performance using a deterministic linkage algorithm. In: *Proceedings of the 2002 American Medical Informatics Annual Fall Symposium*, pp. 305–309 (2002)
12. Berman, J.: Zero-check: a zero-knowledge protocol for reconciling patient identities across institutions. *Archives of Pathology and Laboratory Medicine* 128, 344–346 (2004)
13. Sweeney, L.: k -Anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 557–570 (2002)
14. Samarati, P.: Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering* 13, 1010–1027 (2001)
15. Clifton, C., Kantarcioglu, M., Foan, A., Schadow, G., Vaidya, J., Elmagarmid, A.: Privacy-preserving data integration and sharing. In: *Proc. of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery* (2004)
16. Bhowmick, S., Gruenwald, L., Iwaihara, M., Chatvichienchai, S.: Private-ye: A framework for privacy preserving data integration. In: *Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW 2006)*. IEEE Computer Society, Los Alamitos (2006)

17. Scannapieco, M., Figotin, I., Bertino, E., Elmagarmid, A.: Privacy preserving schema and data matching. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (2007)
18. Agrawal, R., Asonov, D., Kantarcioglu, M., Li, Y.: Sovereign joins. In: ICDE 2006: Proceedings of the 22nd International Conference on Data Engineering (ICDE 2006). IEEE Computer Society, Washington (2006)
19. Kissner, L., Song, D.: Privacy preserving set operations. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005)
20. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Eurocrypt 2004, Interlaken, Switzerland, International Association for Cryptologic Research (IACR) (2004)
21. Emekci, F., Agrawal, D., El Abbadi, A., Gulbeden, A.: Privacy preserving query processing using third parties. In: Proceedings of ICDE 2006, Atlanta, GA (2006)
22. Pon, R., Critchlow, T.: Performance-oriented privacy-preserving data integration. In: Data Integration in the Life Sciences, pp. 240–256. Springer, Heidelberg (2005)
23. Inan, A., Kantarcioglu, M., Bertino, E., Scannapieco, M.: A hybrid approach to private record linkage. In: Proceedings of the 24th Int'l Conf. on Data Engineering - ICDE 2008 (2008)
24. Goldreich, O.: General Cryptographic Protocols. In: The Foundations of Cryptography, vol. 2. Cambridge University Press, Cambridge (2004)
25. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
26. Sweeney, L.: Guaranteeing anonymity when sharing medical data, the datafly system. In: Proceedings of the 1997 American Medical Informatics Association Annual Fall Symposium, pp. 51–55 (1997)
27. IBM: IBM PCI cryptographic coprocessor (2004), <http://www.ibm.com/security/cryptocards/html/pcicc.shtml>
28. Paillier, P.: Public key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
29. Sweeney, L.: Achieving k -anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10, 571–588 (2002)

A Secure Architecture

The join protocols proposed in this paper are based on a secure framework [7]. To orient the reader, we briefly walk through the framework and describe the cryptographic mechanisms. Figure 2 summarizes the system.

Step 1 (Key Generation). KHS generates a <public, private> key pair and provides DS with the public key.

Step 2 (Data Encryption). Hospitals encrypt their records using the public key and send the results to DS.

Step 3 (Query Issuance). After the data is encrypted and stored at DS, a researcher sends a query for the database to DS.

Step 4 (Query Processing). DS executes the requested query and sends the encrypted results to KHS.

Step 5 (Result Decryption). KHS decrypts the result using the private key and sends it to the biomedical researcher.

We demonstrated that the framework supports aggregation queries, which are crucial to biomedical data mining tasks. Specifically, we proved that, through the homomorphic properties of Paillier encryption, we can execute count queries without revealing anything other than the query result. We would like to stress that our proposed system is secure under semi-honest model [24]. In the semi-honest model, the participating parties (e.g., KHS and DS) are assumed to follow the prescribed protocol and those parties only try to infer private information by using what is revealed during the protocol execution. In our context, semi-honest model implies that DS only asks KHS to decrypt *encrypted query results* as prescribed by the protocol. The semi-honest model, widely used in many different data mining tasks, is realistic for our purposes because changing complex protocols buried into large software without being detected could be hard. In addition, due to legal concerns, owners of the DS and KHS may not be willing to accept the potential liability of “not following the prescribed protocols”.

Data stored at DS is semantically secure, so DS can learn the actual values only with the corresponding private key. However, KHS only issues DS a public key. KHS keeps the private key secret and does not share it with DS. As a result, DS is unable to discover the original patient information. Therefore, the data stored at DS are inherently secure against DS, as well as any biomedical researcher that issues queries in the framework.

B Homomorphic Cryptography

To achieve a simple and flexible architecture, we utilize a semantically secure public-key encryption scheme. The public key encryption scheme adopted in our architecture is probabilistic and possesses a homomorphic property. The homomorphic property allows us to compute the encrypted sum of two plaintext values through the corresponding ciphertexts. Formally, let $E_{pk}(\cdot)$ and $D_{pr}(\cdot)$ represent the encryption function with public key pk and the decryption function with private key pr , respectively. A secure public key cryptosystem is probabilistic and homomorphic if the encryption function satisfies the following features:

Constant Efficient: Given a constant k and a ciphertext $E_{pk}(m)$ of m , we can efficiently compute a ciphertext of km , denoted as $E_{pk}(km) := k \times_h E_{pk}(m)$.

Probabilistic: Given a message m , $c_1 = E_{pk}(m)$ and $c_2 = E_{pk}(m)$, $D_{pr}(c_1) = D_{pr}(c_2)$ but $c_1 \neq c_2$ with high probability.

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 as $E_{pk}(m_1 + m_2) := E_{pk}(m_1) +_h E_{pk}(m_2)$.

Our framework can be applied within any additively homomorphic cryptosystem. In this paper, we situate the framework within the Paillier cryptosystem [28] because it has relatively wide-scale adoption and standardization.

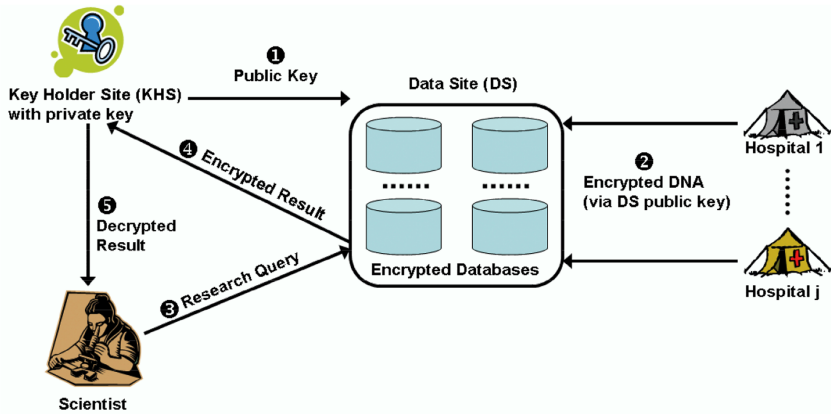


Fig. 2. General Architecture

C k -Anonymity

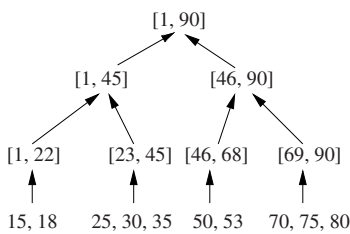
Here, we briefly review k -anonymity [13,29]. Let QI be a set of quasi-identifier attributes that can be used with certain external information to identify a specific individual, T be a dataset represented in a relational form and $T[QI]$ is the projection of T to the set of attributes contained in QI .

AGE	ZIP CODE
25	54339
75	47500
50	47535
30	54788

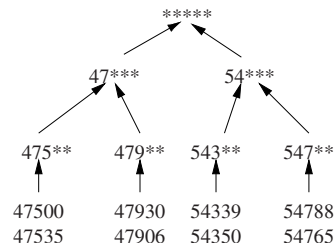
(a) Original Data

AGE	ZIP CODE
[23, 45]	54***
[46, 90]	475**
[46, 90]	475**
[23, 45]	54***

(b) 2-Anon. Data



(c) VGH of AGE



(d) VGH of ZIP CODE

Fig. 3. Data tables and value generalization hierarchies

Definition 1. $T[QI]$ satisfies k -anonymity if and only if each record in it appears at least k times.

The criteria for k -anonymity can be achieved via a number of mechanisms. In this paper, we concentrate on *generalization* [29]. In generalization, values are replaced by more general ones, according to a value generalization hierarchy (VGH). Figure 3 contains VGHs for the attributes *AGE* and *ZIP CODE*. According to the VGH of *AGE*, we say that 25 can be generalized to [23, 45].

As an example, consider Figure 3(a). Here, we show a dataset T with quasi-identifier $QI = \{AGE, ZIP\ CODE\}$. By generalization according to the VGHs, we can derive dataset in Figure 3(b) ($T[Q]$), which satisfies 2-anonymity.