A Database De-identification Framework to Enable Direct Queries on Medical Data for Secondary Use

B. S. Erdal1,2; J. Liu1; J. Ding1; J. Chen1; C. B. Marsh2; J. Kamal1; B. D. Clymer2
1Information Warehouse, The Ohio State University Medical Center, Columbus, Ohio, USA;
2Electrical and Computer Engineering, The Ohio State University, Columbus, Ohio, USA;
3Internal Medicine, The Ohio State University Medical Center, Columbus, Ohio, USA.

Objective: To qualify the use of patient clinical records as non-human-subject for research purpose, electronic medical record data must be de-identified so there is minimum risk to protected health information exposure. This study demonstrated a robust framework for structured data de-identification that can be applied to any relational data source that needs to be de-identified.

Methods: Using a real world clinical data warehouse, a pilot implementation of limited subject areas were used to demonstrate and evaluate this new de-identification process. Query results and performances are compared between source and target system to validate data accuracy and usability.

Results: The combination of hashing, pseudonyms, and session dependent randomizer provides a rigorous de-identification framework to guard against 1) source identifier exposure; 2) internal data analyst manually linking to source identifiers; and 3) identifier cross-link among different researchers or multiple query sessions by the same researcher. In addition, a query rejection option is provided to refuse queries resulting in less than preset numbers of subjects and total records to prevent users from accidental subject identification due to low volume of data.

This framework does not prevent subject re-identification based on prior knowledge and sequence of events. Also, it does not deal with medical free text de-identification, although text de-identification using natural language processing can be included due its modular design.

Conclusion: We demonstrated a framework resulting in HIPAA Compliant databases that can be directly queried by researchers. This technique can be augmented to facilitate inter-institutional research data sharing through existing middleware such as caGrid.

1. Introduction

Enabling clinical and translational research is dependent on availability of the electronic medical record (EMR) systems as the data sources [1, 2]. The use of EMR for research is governed by federal privacy regulations such as HIPAA [3] and the Common Rule [4]. In order to use identifiable data in research, an Institutional Review Board (IRB) approval must be obtained. Even though the IRB procedures can be extensive enough to impede opportune research and discoveries, these rules are in place to ensure complete safeguard of Protected Health Information (PHI).

Generally, when PHI for subjects within a dataset is removed, EMR data can be treated as non-human data. To address the concern of timeliness in research activities, the IRB at The Ohio State University Medical Center (OSUMC) has granted the Information Warehouse [5] (IW) the “Honest Broker” status (the Honest Broker Protocol [6], or HBP) that allows IW analysts to provide de-identified clinical data for research (Fig. 1).

Researchers only need to sign a data use agreement before gaining access to de-identified data. While the HBP greatly accelerated research access to data, IW analysts are overwhelmed with new data requests that need to be manually de-identified and delivered. This manual process also carries the risk of accidental PHI exposure. These limitations and/or problems can be remedied by the creation of a de-identified IW (DIW) [7] in conjunction with query tools that allow researchers to build complex queries without specific knowledge of any query languages.

In this paper we report on our de-identification processes (Fig. 2) that meet HIPAA regulations as well as more restricted rules set forth by OSU IRB. Following Figure 2, these processes can be explained as follows. Beginning with identifiable data inside the IW repository, all data go through a set of de-identification algorithms (step 1) and the resulting data are stored in a separate database as a limited dataset with dates unchanged in the DIW (step 2). These data can become available to researchers when a limited dataset is requested (step 3). Alternatively, all de-identified surrogate identifiers (DSI) and dates are further de-identified using a 2-step scheme both in the database and in each user session, leading to a completely de-identified dataset (step 4). These processes are generalizable: they can be used to export data into destination databases (or warehouses) while satisfying destination security requirements through different parameter sets; they can also be used to publish data from existing systems without...
exposing source identifiers. In our efforts, we are describing processes that de-identify data in order to enable use of data in such research related endeavors. These processes operate at database system level, and the idea is to make databases themselves protect the data from both internal and external threats and still enable incremental data updates.

2. Background

Preserving patient privacy and confidentiality while sharing [8] and enabling the secondary use [9, 10] of medical data brings forward the problem of de-identification [9, 10]. This has been the focus of much research in recent years [11–17]. As it is well recognized by Narayanan and Shmatikov [18] unnecessary de-identification may destroy the utility of datasets. However, as noted by Cavoukian and El-Emam [19], this does not mean we should stop de-identifying, but in addition we should incorporate careful risk assessments [16, 19, 20].

Many efforts have been made to de-identify data in conformance with HIPAA and IRB regulations that cover both structured [11, 15, 21, 22] and unstructured [23–25] clinical data. Methods have been described using a hash function to build a de-identified bio-repository [21]. Other methods have been reported on either de-identification schemes [17] or trying to guarantee anonymity [26, 27]. De-identification of medical free text has been studied and published with great progress in the past years [28]. Here, we would like to note that we consider de-identification of MRI and CT images a structured data de-identification problem which can be solved by proper handling of DICOM headers, and we consider removal of information from pixels of images (e.g. removal of patient names from ultrasound images) an optical character recognition (OCR) problem which is outside the scope of this paper.

De-identification methods for structured data can be divided into two categories: 1) heuristic methods, and 2) statistical methods. When heuristic methods are employed, usually all PHI containing field values are removed or replaced in a procedural manner. Well known examples in the literature include works of Roden et al. [21] and Pulley et al [29], where both cases utilize a one-way hash algorithm (SHA-512) in order to de-identify their data; in addition patients in their health system are presented with opt-out forms so they can be excluded from the repositories if they choose. There are also cryptographic approaches in the literature which demonstrate returning aggregate results without giving any patient information [30, 31] and performing secure join queries on encrypted datasets originating from multiple institutions [32]. Statistical methods as reported in studies of Sweeney [27, 33, 34] form the basis of many other works in this category [26]. In k-anonymity [27] Sweeney describes an algorithm that protects the anonymity of an individual in a dataset by ensuring existence of at least k records with the same characteristics (e.g. at least four other records have the same values) belonging to other individuals in the same dataset. Also in this category many differential privacy based methods [35] are defined in order to govern the inclusion or removal of database fields or items to datasets. Generally, the probability of re-identification versus the effects of added noise on statistical meaning derived from the datasets is calculated and acted upon.

De-identification of unstructured medical text is an active research area, and much advancement has been made. The research community has even organized challenges such as the i2b2 challenge where de-identification and concept extraction from the medical text was evaluated using Precision, Recall, and Balanced F-measure \((2 \times \text{Precision} \times \text{Recall}/(\text{Precision} + \text{Recall}))\) against community prepared and annotated gold standard medical text corpus [28]. Promising methods include systems combining dictionaries with pattern matching [36] as well as statistical systems [37]. In i2b2 challenge, the best performing system was the work of Wellner et al. where they combined two toolkits for named entity recognition (Carafa and LingPipe) [37], and they were able to achieve precision, recall and f-measure all greater than 96%. Later, using the same i2b2 corpus Uzuner et al. were able to produce higher precision (99%), recall (97%) and F-measures (98%); they used an open source Support Vector Machines (SVM) library called LIBSVM during their implementation [38].

In a recent review by El Emam et al. [16], the risk of re-identification is shown to be quite low on a properly de-identified dataset. Empirical tests conducted by statistical team experts assembled at U.S. Department of Health and Human Services’ Office of the National Coordinator for Health Information Technology has shown that risk of re-identification is as low as 0.013 percent when HIPAA safe harbor methods are used. Their tests were conducted under realistic conditions using a 15,000-patient set [39]. It is also worth noting that potential leaks or attacks can be caused by rogue employees (inside jobs) [19].
In cryptography, the protocol for one party to interactively answer another’s question without revealing the underlying hidden secret is called a zero-knowledge protocol [40]. If we were to give a medical dataset related example, one way to achieve zero-knowledge protocol could be adding a random number to patient record identifiers while answering a question. Here, we would be producing a new set of unique random codes for each patient for the given question and this approach could potentially be used as a building block for developing HIPAA and Common Rule compliant query methods [41].

2.1 The OSUMC Honest Broker Protocol

The IW’s “Honest Broker” status as a provider of de-identified clinical data for research purposes was approved as an annually reviewed procedural protocol by The Ohio State University IRB in April 2006.

IW data analysts prepare the de-identified data by removing PHI and deliver the dataset as a bundle of coherently linked files to the researcher. In a de-identified dataset, dates are recorded as time intervals from a patient’s first visit. Alternatively, actual dates, zip codes, and ages over 89 can be included in a limited dataset [18]. To safeguard against accidental identification, resulting de-identified or limited datasets with less than 25 records are not delivered; this number is determined by the OSU IRB protocol (can be adjusted for other institutions). Under certain circumstances, limited dataset is preferred over de-identified datasets because of the inclusion of temporal elements in studies. Also, queries based on previous results are not allowed to prevent users from performing longitudinal studies using either limited or de-identified dataset. DSI are changed between each query. This approach, while allowing retrospective analysis, inhibits forward longitudinal study of particular subjects. Furthermore, should an inadvertent identification occur due to investigator’s prior knowledge of a particularly unique data pattern, the Data Use Agreement stipulates that the investigator must immediately seek IRB oversight.

3. Design Objectives

Our first objective is providing a framework that is simple enough that it can be easily modified in support of potential policy changes from our IRB, HIPAA or structural changes in source database(s). Moreover, there are other honest broker protocols in the literature, such as the University of Michigan [43] and the University of Pittsburgh [44] honest broker systems. The design of our framework should be generic enough that it can, as a whole or in part, be adoptable/adaptable in these systems based on specific institutional needs.

To support both research and education, it is desirable to have a completely de-identified database (DIW) that is similar in data structure to the identifiable version (IDB). This DIW must be HIPAA compliant in terms of the masking or removal of PHI, and should ideally be capable of defending against re-identification attacks both internal and external. Structural similarity between IDB and DIW is beneficial to ensure that previously developed applications designed for use with the IDB remain applicable for the DIW with minimal modification.

4. System Description

In general, our de-identification framework is constructed as a methodology for extracting, transferring and loading (ETL) data from multiple sources into a comprehensive de-identified database (or data warehouse). Quite often sources of data within large medical institutions reside across multiple systems; thus, even when they are collected in a single repository (e.g. data warehouse) their origins are usually federated. This means potentially multiple record identifiers could have been used for a given patient. Hence, while removing PHI from records special attention is needed in order to maintain the record consistency across systems.

Following our system architecture depicted in Fig. 2 we describe how we create a de-identified version of our source database system.

In this model, data flow is one way only and the objective is to sever all possible linkages from destination to the source system in terms of patient identifiers, while the source can still safely update the destination. During our description we follow...
the direction of data travel through our de-identification pipeline. We first explain the operations taking place in the source database system, followed by operations on the destination database system.

### 4.1 Operations on the Source System

As shown in the left side of Figure 3 these operations take place in an "Source Encryption Account" (SEA) which is an account that has read privileges (SQL SELECT) on all database schemas (excluding the SYSTEM account) chosen to be de-identified. SEA can be further safeguarded by using data encryption functions within the database, if available. None of the database accounts (except the SYSTEM account) can see contents of the SEA. Using its stored procedures, the SEA can execute functions which can hash given strings, generate random numbers of variable sizes, or write data to master tables. The hashing and random number generator functions mentioned here are implemented as JAVA Stored Procedures using open source functions of java.security package (java.security.MessageDigest and java.security.SecureRandom). These are cryptographically strong pseudo-random numbers which are minimally compliant with the statistical random number generator tests [45] specified in FIPS 140–2, Security Requirements for Cryptographic Modules, section 4.9.1 [46].

First, looking at the tables to be de-identified, SEA creates master tables for each unique identifier type such as Medical Record Number (MRN), Encounter Number (ENC), and Accession Number (ACC#). These master tables hold all unique identifiers for all patients whose data are going to be de-identified. Then, based on these Master tables, Mapper tables are generated by using SHA-2 hashing based algorithm implementations available (SHA-256, SHA-512) as part of stored procedures in the SEA. SHA-2 based algorithms instead of SHA-1 were chosen because of potential mathematical weaknesses of SHA-1 indicated by experts [47]. During our tests, where we have repetitively performed entire Mapper table builds, we

---

**Fig. 3** De-identification process across databases; on the left, operations on the source database system; on the right operations on the destination database system.
have used SHA-256 instead of SHA-512 for faster table generation. Using a secure random number implementation from its stored procedures, the SEA then assigns a unique random number to each hashed value (f(x), Fig. 3). This process results in generation and storage of a hashed and random value pair (e.g. hMRN, rMRN) for each unique identifier (e.g. MRN). The use of a hashing algorithm on all IDs in the source system is a necessary step to ensure that there is no accidental mapping exposure to database users who have access privileges to the source system. We choose not to use the hashed string as reported by Roden et al. [21] for two reasons: 1) hashed string is considered a directly derived ID, thus its use is questionable by HIPAA, and 2) numeric IDs perform better than long strings in database queries [7].

Using the Mapper tables and its stored procedures, SEA takes each record from tables to be de-identified, and first re-runs its hash function (e.g. finds hMRN) on identifiers (e.g. MRN) to be replaced in order to find the random value to be used (e.g. rMRN), then replaces the identifier with the precalculated random value. Then, this record, which now has a DSI that has the form of a real identifier, is written to the corresponding staging table on the destination database's encryption account. At the SEA only the Mapper tables which hold the hashed value and the random number pair are kept. While the hashed numbers scramble the data, the replacement random number to be used in destination system ensures compatibility with the source system. The Master tables are only used as temporary in memory data structures and they are discarded after each data transfer. This is done against accidental exposure of linkage between the random ID and real ID pair, and prevents internal users from manually linking data between IDs used in source and de-identified data.

### 4.2 Operations on the Destination System

As shown on the right side of Fig. 3, operations here take place in a "Destination Encryption Account" (DEA). Evaluating the tables populated by the SEA, the DEA creates its own Mapper Tables by using its own stored procedures analogous to SEA's stored procedures. During the creation of these new Mapper tables each unique identifier gets additional multiple unique random numbers. Hence, each unique random number which was originally created by the SEA (e.g. rMRN, etc.) gets multiple columns of unique random numbers (CURN) at the DEA. For example for an MRN Mapper table we would have the rMRN as the key and if we had n random MRN columns on this table we would have columns such as uMRN1, uMRN2…uMRNn. In addition, DEA creates a random date offset (rDOS) value for each random MRN (uDOS1, uDOS2…uDOSn) and adds them to the Mapper Table as columns. As explained below in creation of Limited Datasets and De-identified Datasets, Mapper tables other than the MRN Mapper (Encounter Mapper, Accession Mapper, etc.) do not have random date offset columns. This process allows production of independent random series to be used against potential attackers who may try brute force attacks (by creating multiple sessions) to expose static random numbers used by the SEA. The number of CURNs are pre-determined prior to setup (we are currently working on a dynamic version as well), and during our tests we used minimum two of them. CURNs are shuffled and refreshed after 40,000 login sessions or within 24 hours (SEA initiates an update), whichever comes first. The additional layer of CURNs adds less than 10% overhead on query execution times (when a query directly involves an identifier related field). In Oracle based systems we eliminate this overhead by utilizing bitmap join indexes [48].

The destination system uses two more database accounts to expose data to its users: 1) a Limited Account (LA) enabling access to Limited Datasets; 2) a De-identified Account (DA) enabling access to De-identified Datasets. Both of these accounts do not hold any physical tables; all database objects in these accounts are views based on tables and views on the DEA. DEA controls and dictates how frequently the views in these two accounts are refreshed.

#### 4.2.1 Creation of Limited Datasets

Using its stored procedures, DEA creates views by joining the tables populated by the SEA and the Mapper tables. In these views each unique identifier (e.g. rMRN) is replaced by another (e.g. uMRN). These views keep the original date values. Using these views DEA creates views for limited dataset in the LA. However view and column names for these views are given based on naming conventions of the source system, which ensures compatibility and simplicity in adaptation for tools and users who are familiar with the source system. DEA gets an additional random number from the session variables for each user login. Session dependent random variables are preferably picked from higher cardinality numbers. For example, if we were to map a 9-digit MRN to a 10-digit MRN, the random session addition would be a value between 1,000,000,000 and 9,000,000,001. This addition is introduced in order to prevent users from comparing results across different studies. It also ensures that a user gets a different identifier for a given patient every time the user logs in to the system (also at system level the database limits how long a session can last).

#### 4.2.2 Creation of De-identified Datasets

Figure 4 provides a walkthrough for the de-identification process. In this example, an MRN, 900000001, is hashed into a long string (H_MRN in H_R_MAPPING Table) using SHA-256. This hashed string is assigned a random number, 6708389166. Patient data for 900000001 is replaced by 6708389166 as a surrogate identifier and saved in the de-identified IW database (LIM_ENC_DX, the blue box). By using the linkage in USER_MRN, we further de-identify the unique surrogate identifier to a value that is changed periodically (U_MRN). This changing U_MRN is further de-identified by a hidden algorithm using a user login session-dependent random key to produce the user viewable identifier in ENC_DX of the de-identified database (the green box). Once a user has obtained a dataset and has logged off, the de-identified MRN, 4760383037, cannot be
linked back to any intermediate or original identifiers (U_MRN, R_MRN, and MRN) – even data analysts do not have the capability of linking the result set back to any identifiable information. Also shown in this example is the date shifting result. Subject dependent date shift values of +4 and –5 days are applied to MR = 900000001 and 900000002, respectively. A session-dependent shift of +2 days is further applied to all date results in a query in this example session. This produces different date shifts among different subjects in the same query, as well as different date shift between same queries run in different user session.

These datasets are created in similar fashion to the Limited Datasets, with access through the DA (based on user privileges a user can only access either LA or DA). The difference is that the date columns are treated as identifiers as well; hence, records are padded by unique random date offset values (for DA) for each MRN, as well as random dynamic session variables. As a result, the time intervals between the dates for any specific patient are kept but the original dates are no longer available. Beginning from the source system, the whole process could be described as a one way one to one mapping ((source) MRN + hashing + (source+destination) random number + (destination) Random number + random session variable).

As one can expect, adding additional views and functions over existing table structures brings performance overhead. However, this overhead can be minimized or eliminated using methods such as placing additional indexes to source tables or pinning frequently used look-up tables to the shared memory, resulting in dramatic improvement on query execution times and overall performance. Being a read only system (end-users are not allowed to alter or write new data); our de-identified instance was fine-tuned for fast query response times rather than a system allowing real-time data updates. Our current configuration is designed for updating once daily.

4.3 System Validation

We performed our system validation by following the trail of the data movement and tested the necessary functionalities along the way. Some of these functionalities are in place to fulfill HIPAA requirements; others are in place to fulfill our internal IRB requirements. While our focus here is still the validation of Limited and De-identified Dataset Generation, we cover validation of all main system functionalities.

1. ZIP code roll-ups: Our current HBP requires that we follow the HIPAA Privacy Rule [3]. Hence, we form ZIP codes with the same 3 initial digits that contain more than 20,000 people. The scripts related to this functionality are executed in the SEA, before the data are passed on to the DEA. We have tested this feature by simply declaring different underpopulated ZIP codes and we have verified the results by simply counting merged population changes and the disappearance of the under-populated ZIP codes in the result set.

2. Handling Patients over Age 89: This feature only applies to de-identified datasets. Within our framework, when patients reach age 90, updates to their records on age related fields are suspended. SEA continues to process these records, and keeps them in a separate table. Updates to DEA for these patients occur only after the patients are deceased.
Until that time their ages are kept at the fixed value of 89. We have verified this feature by querying the datasets to be passed from the SEA and found no patients over the age 89. We also verified that, when suspended dataset (patients 90 and older) were merged back we can retrieve the original dataset.

3. Limited and De-identified Dataset Generation: In theory, each source identifier should be replaced by a new identifier every single time. One to one mapping is a rule that we enforce on all tables and views handling transactions related to identifier de-identification. For completeness, we still performed test queries across the system. For example, one of the queries we executed counted the maximum number of patients and encounters seen among different diagnosis codes, grouped by months and ordered by maximum encounters. We executed similar queries in the source system and compared the results from the limited dataset. Then, we executed similar queries without date constraints, and compared the results from source database to results from the limited and de-identified dataset and we verified that the total counts matched in all cases. Finally, using multiple accounts and sessions running the same query we have verified that during concurrent sessions by multiple users (10 during our tests), or consecutive sessions by the same user (4 during our tests), no identical identifiers are retrieved.

We are aware that having a date obfuscation function, which is function based and parameterized, enables the de-identified dataset to produce similar results to source dataset (which has real dates) when small random shifts are used during the date field generation. The statistical variations and associated re-identification related risk factors introduced by small random time shifts are currently beyond the scope of this paper, and it will be further evaluated and reported later.

4. Query rejection: As an optional and parameterized functionality, query rejection could be turned on to safeguard against at least two scenarios in order to minimize the possibility of subject re-identification: (1) the number of records returned is below a preset threshold; and (2) sufficient number of records returned are from a population (unique subjects) below a preset threshold. Our Honest Broker Status through OSU IRB mandates this parameter to be set at 25 or higher, but it could be set at any number through a user definable parameter. In order to test this feature we simply executed queries which would return results below the preset parameter. It is verified that when queries return less than 25 distinct subjects and/or records, a message is presented to user instead of the result set and the normal queries behave seamlessly to user. Query rejection functionality is implemented as an optional in-database module using Fine Grained Auditing [49].

### 4.4 Test Environment and Setup

For our tests we transferred four tables from our source clinical database to our de-identified instance, and then we generated a limited and de-identified dataset version of each table at destination (Table 1).

Using these tables and mapper tables we created two views for each table; one view for limited dataset access, and another view for de-identified dataset access. Following is an example SQL code snippet for these types of views:

```sql
CREATE OR REPLACE VIEW PATIENT_DEIDENTIFIED
(MRN, GENDER, RACE, DOB ...) AS
/*Procedure call*/
SELECT a.MRN, a.GENDER, a.RACE ...
/*Procedure call*/
FROM PATIENT.PATIENT_LTD a, ENCRYPT.MRN_MAPPER b
WHERE b.MRN_RANDOM = a.MRN;
```

As we have mentioned earlier, users get their random numbers for each new session by calling stored procedures (DEIDVW) of the DEA. Since these views are function based, there are no additional maintenance requirements.

All tests were performed on a Sun Fire V445 server (4× 1.6GHz UltraSPARC IIIi CPUs, 32GB of memory) running Solaris 10 and Oracle Relational Database Management System (RDBMS) 11gR1, with tablespace compression turned on. All necessary indexes were created on all patient identifier fields (rMRN, rENC, etc.), date fields, as well as code look up fields, with table statistics gathered using all available data prior to test query executions. Frequently used code lookup tables (Mapper tables) and their indexes were pinned into memory for fast access. We allowed each connection to utilize only a single CPU.

### Table 1  Tables de-identified during testing

<table>
<thead>
<tr>
<th>Source Table Name</th>
<th>Columns</th>
<th>Unique Identifiers</th>
<th>Date Fields</th>
<th>Records for Testing</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENC_CLIN</td>
<td>45</td>
<td>2</td>
<td>4</td>
<td>5,697,118</td>
<td>OSUMC clinical encounters</td>
</tr>
<tr>
<td>ENC_DX</td>
<td>15</td>
<td>2</td>
<td>2</td>
<td>20,175,845</td>
<td>One or more diagnoses for each encounter</td>
</tr>
<tr>
<td>ENC_ICD9_PROC</td>
<td>14</td>
<td>2</td>
<td>2</td>
<td>1,899,487</td>
<td>One or more procedures for each patient</td>
</tr>
<tr>
<td>Patient</td>
<td>16</td>
<td>1</td>
<td>3</td>
<td>1,005,657</td>
<td>Patient demographics</td>
</tr>
</tbody>
</table>
5. Status Report and Results

5.1 Test Queries and Results

During our query performance evaluation we executed eight representative queries (Table 2 in Appendix for query details). We have executed each query 20 times with four different parameter sets, on two views per source table each representing a limited dataset and a de-identified dataset, and we measured query execution times (1,280 total executions) by CPU time (1.4 – 44 seconds) as reported by Oracle’s SQL Trace Facility and TKPROF utility. Our average CPU time was ~15 seconds. These are considered reasonable execution times for the given types of queries for our environment.

Overall, during our tests, views for limited datasets performed better when compared to their de-identified counterparts (Fig. 5). This is due to the fact that we perform fewer lookups during creation of limited datasets (since original dates are used). In fact, since the de-identified views have more complex functions generating them, slowdowns in query times are expected (e.g. Query 3, Query 5 shown in Appendix) whenever join conditions on date fields are present. However, the whole system being a read only system allows us to optimize for fast reads (rather than read and writes); hence, we can stay within reasonable query response times. Also, through our experiments we have seen that as queries get more complex (e.g. Query 6 is a more complex query when compared to Query 4), query execution times do not get as impacted as one would expect. As this may seem counterintuitive, our query evaluations revealed that additional query complexity enabled this performance upturn by increase in filtering.

5.2 Generalizability

As we have mentioned earlier, our framework is a de-identification framework which in general is a HIPAA and local IRB compliant ETL methodology. Our initial local instances were built using PL/SQL and JAVA stored procedures for data transfers. However, different enterprise level production environments may have their own requirements for software deployment. Therefore, in order to test generalizability we tested all our scripts through two other industry accepted ETL tools: IBM DataStage (version 8.5) [50] and Oracle Warehouse Builder (11g) [51]. Both tools allow execution of local scripts in databases; hence, we were able transfer data using a SEA to a DEA in both cases. In addition we have created an example i2b2 database instance to demonstrate DEAs session based capabilities [52] and another prototype system that demonstrates generation of de-identified image datasets which can also be viewed on mobile devices (e.g. iPad, iPhone or Google Android based devices) [53].

5.3 Reliability

In addition to measuring how queries performed in our environment we further evaluated reliability of our de-identification framework.

5.3.1 Internal Consistency

As a continuation of our development and potential problem discovery efforts, we created artificial datasets with one non-PHI associated field injected with a unique identifier (IUI) marking each record. We created artificial tables (10 tables) holding potential problem discovery efforts, we created artificial datasets with one non-PHI associated field injected with a unique identifier (IUI) marking each record. We created artificial tables (10 tables) holding these records (20 million records). Then we ran these tables through our de-identification framework for 100 consecutive runs and using a third independent database (independent from source and destination databases) we compared resulting de-identified sets from 100 independent sessions. As we expected our IUIs did have 100% matches while fields for PHIs did have 100% mismatches or had no data depending on earlier defined HIPAA requirements (and stricter OSU IRB requirements when necessary based on number of subjects returned).
5.3.2 Random Number Generation during Sessions (Session Security)

As it can be followed from Figures 3 and 4 in our Methods section, final pseudo-identifier generation for a given session relies on generation of random numbers to be used during that session. Therefore, in order to hide the underlying identifier or pointer our random number generation process has to be rigorously tested, because if exposed and not changed, the DSIs can potentially be used as the “new” patient identifiers. For that purpose we have followed National Institute of Standards and Technology (NIST) guidelines for statistically testing random and pseudorandom number generators (PRNG) [54]. During NIST statistical evaluations mainly the algorithmic predictability is measured through multiple tests [55]. A PRNG method passing multiple tests indicates its strength. While NIST’s testing package includes 15 tests, some of these tests supersede others, for example Maurer’s “Universal Statistical” Test [56] supersedes the Monobits Test [55, 57]; however, the former is intended for longer bit sequences. Therefore we did not employ the Maurer’s Test during our evaluation but instead we used the Monobits Test (our test numbers were 32 bits long). We looked at 86400 pseudo random numbers (PRN) generated (representing one PRN generation for each second for a 24-hour period) by using our session login script. These numbers were then converted into 32-bit binary sequences, and given as input to the NIST Package (NIST recommends minimum 1000 PRNs for reliable results, we used 86400). Table 2 shows our test results.

In Table 2 we provide a condensed version of our results. The results report includes the proportion of passing sequences where each pass or fail is assigned based on given internal P-values for each test. During NIST statistical testing a sequence passes a statistical test whenever the $P \geq \alpha$ and fails otherwise, for further details please refer to NIST documentation [55]. During our tests some of the statistical tests were applied multiple times according to default settings on NIST testing package. For example if we look at results from the Non-Overlapping Template Matching Test, which was repeated 149 times; we had 104 results with passing proportion of 32/32 (42 with 31/32, etc.)

Our PRNG methodology did produce satisfactory random numbers based on every applicable NIST statistical test.

6. Discussion

It will be beneficial to the readers to compare the necessity of the steps in our framework to existing systems. Our design objectives were to form a core framework which is flexible enough and can aid operational needs and enable future modifications. Within our framework the source data could be in any form or structure in a database and after de-identification that form and structure can be maintained (with the exception of free-text reports). This approach decouples query tools’ operational security needs by using database security. It takes the maintenance and responsibility of HIPAA compliance from query tools and pushes these responsibilities down to database level. This means most of the open source or commercial tools which are used for looking at the data (e.g. for training and research purposes) can still remain operational with minimal or no changes when pointed to a de-identified instance. This is crucial, especially for open source tools, because with given time and effort their behavior can be altered or modified and potentially any security provided at the tool level can be bypassed. The individual components of our framework are based on well-established and well-known methods.
such as hashing and random number generation which makes it easily adoptable/adaptable [52, 53]. The cryptographic level information security methods used in our framework are empirically measurable and they achieve 100 percent pass rate on well-established national statistical standards [54, 55]. The unique combination with which the individual methods are applied makes this framework enable HIPAA, the Common Rule and the local IRB compliance.

On the source system, the use of a hash function, and then another random number before the data are sent from SEA to DEA prevents hash keys being exposed to potential outside attackers, and the hash keys themselves provide a defense layer against potential internal attackers. If we were to rely on hash keys only, an outside attacker can potentially expose all patient identifiers by knowing only a subset of identifiers (this means attacker has both the input and the output for a given subset, therefore, potentially the method can be hacked). While there are examples in the literature using the one-way-hash method [21, 29], in these examples every patient being included in the datasets has an opt-out signature in file. But, we are trying to avoid the need for collection of opt-out forms (in our source EHR systems we have more than 2.6 million patients with admissions dating back to 1985); and this can be avoided by satisfying the HIPAA and the Common Rule. We would also like to note that under HIPAA, use of a one-way hash as a substitute for de-identification is explicitly forbidden. In practice, this means access to data for research use cannot be granted without an IRB approval when using a one-way-hash only approach.

On the destination system, the use of session based random number generation methods enables implementation of a zero-knowledge protocol for the medical datasets recycled for secondary use. These methods are in place to prevent patient identifiers from being traced back to record identifiers. Here, we would like to note that a simple records-based system cannot be considered exempt from HIPAA. In practice, this means de-identifiers such as static random numbers that remain unchanged across queries cannot be used by themselves in order to receive exemption. The added session dependent identifier generation and query rejection methods minimize the risk of re-identification. Our current (pseudo) random number generator provides mapping by increasing the cardinality of the data, meaning for a given example of 9-digit random identifier when mapped to a 10-digit one, the chance of two users getting the same number on a concurrent session or the same user getting the same number between consecutive sessions is on the order of 1 in $10^{10}$ within our scheme. Considering this is the third place the random switching is taking place, even with a brute-force attack, the attacker can only get back to the previous source number given by the DEA, which is to be replaced within 24 hours. In this scheme, even if the physical hardware is stolen or lost, there would not be an identifier that can be mapped back to source data, since the numbers from SEA are nowhere in destination system.

Unlike k-anonymity and differential privacy based methods, our query restriction methods do not alter query result sets beyond HIPAA requirements. For example, while all patients over 89 may be marked 89 regardless of their age; no female patient would be marked male or vice versa in order to prevent re-identification. There is growing concern that tampering with medical data in order to prevent re-identification may diminish its values and render the data unusable [19, 26], and following the HIPAA and The Common Rule has already been shown to be effective [19, 39]. If and when a query produces a narrow patient population (increase in risk for re-identification), rather than returning an altered dataset, our query restriction methods simply produce a message telling the investigator to seek further IRB guidance in order to access the dataset asked by the query. The number of patients available in our QA system was around 1 million patients, and this number will be exceeding 2.6 million in the production version. The minimum number of distinct records suggested by our IRB for our environment was 25 ($\sim 1 \times 10^{-6}$ of the current population), and we predict as the patient population grows there will be less number of queries which will require specific IRB approval. However, we would like to point out that this is a number each institution should evaluate based on the local environment (level of risk each institution willing to take may differ). Nevertheless, if an institution chooses to employ further restrictive methods our framework does not prohibit such modifications. We acknowledge there are limitations in our de-identification framework. Despite removing all HIPAA required PHI and taking additional preventative measures, there is still a chance for re-identification. As it has been indicated by many researchers in the literature [12, 20, 58], there is always a possibility for re-identification by a primary care giver or relative, etc. who can identify a patient by recognizing his or her unique combination of diagnostic codes, test results, lab values, etc. Currently, we address this issue by utilizing our data use agreements which are part of our HBP.

While our framework fully supports inclusion of scrubbed text documents to be included in the de-identification process, we did not include de-identified text documents in our institutional implementation because of two reasons: 1) Analysis on IW customer data requests indicated that more than 95% of queries were code or value based. Hence, while addressing our immediate needs we did not focus on text de-identification. 2) We believe that 100% removal of the identifiers from free-text reports cannot be guaranteed using automated methods at this point. Even though great progress has been made in medical free-text de-identification over the recent years, the inclusion of text documents still carries a higher risk than we are willing to take at this time. Our current repositories include more than 8 million text reports. The best system we have seen in the literature has 99% precision [38], and even if we had adopted such an implementation with the assumption that it can de-identify any type of text document, we would still carry the risk of exposing more than 80,000 patient identifiers. Currently, for our environment we are evaluating the inclusion of text documents as a query-only-system (no document retrieval allowed) where searches are allowed only through controlled vocabularies [59–61]. Over recent years great progress has been made in infor-

Methods Inf Med 3/2012

© Schattauer 2012
information extraction from textual documents [62–66] and we also believe a combination of these new techniques could potentially speedup the lengthy chart reviews. However, if benefits outweigh the risks for one's purpose, through our modular framework one can enable integration of de-identified text reports. There are mature tools in the literature that can be employed for such purposes [28, 67]. We recommend use of these method before the data are moved (in SEA) into a de-identified instance (DEA).

Currently, our framework is being tested for quality and assurance by developers, business analysts and a select number of expert users who can write and execute their own SQL queries without any assistance. Our analysts report that, when used along with queries where de-identification is a requirement, depending on the complexity of the dataset our framework enables time savings of 30 minutes to an hour compared to routine editing performed by analysts. In addition potential errors which would result in additional laborious work are eliminated. Even though a general end-user savings or usability report is not available at this point, based on the responses from our expert users, we predict users will appreciate the time savings; since it is quite possible for an end-user to wait 10 days to 3 weeks in order to receive manually prepared datasets for their queries from an analyst.

7. Conclusion

The creation of a De-identified Information Warehouse (DIW) is a continuing effort at the OSUMC IW to better support research activities using clinical data. The ultimate goal is to enable a direct connection between a researcher and the data. IW’s HBP has shortened the time it takes for researchers to access the data they need. Compared to 10 data requests in 2003, IW had reached 256 and 311 research data requests in 2009, 2010, respectively.

With the HBP, OSUMC researchers can gain access to limited and de-identified data much faster than before. However, IW analysts can only process so many data requests at any given time. The new bottleneck facing researchers is the lengthening data request queue. As a solution, DIW can easily be coupled with commercial database query tools such as Oracle Answers [68]; in house developed query tools such as uQuery [69]; or open source development efforts such as i2b2 [52, 70, 71], catGrid [72] etc. This can improve the efficiency of data requests by letting researchers perform their own queries on their time, with greater interactivity and increased flexibility.

Our framework successfully de-identifies and removes all HIPAA mandated PHI from structured data elements, and provides conformance with the local IRB, while maintaining data integrity. This framework guarantees that even for each session new identifiers are generated. This makes our framework a suitable tool for aiding core de-identification operations; which needs to take place at medical data warehouses in order to support non-human-subject research using clinical data. Data sharing between multiple institutions, on the other hand, has other challenges associated with data standardization, network management, and many more beyond the scope of this work. Once those challenges are properly addressed, this framework can be used as part of the solutions for multi-institutional data sharing.

References


© Schattauer 2012

Methods Inf Med 3/2012
### Table 1

Queries with different forms and parameters were used during performance evaluations. Each query is repeated in a non-sequential order with different parameters.

<table>
<thead>
<tr>
<th>Query</th>
<th>SQL</th>
<th>Description</th>
</tr>
</thead>
</table>
| Query 1 | ```sql
SELECT count(1) FROM ENC_DX_LTD
WHERE dx_cd like '250%';
``` | Number of diagnoses with diabetes |
| Query 2 | ```sql
SELECT count(distinct a.mrn) FROM ENC_DX_LTD a, ENC_CLIN_LTD b
WHERE a.mrn=b.mrn
AND a.encounter_no=b.encounter_no
AND b.prim_dsch_dx_cd = '707.15'
AND a.dx_cd like '250%'
AND a.dx_rank<>1;
``` | Number of patients who had primary diagnosis of "Ulcer of Other Part of Foot" and a non-primary diagnosis of diabetes. |
| Query 3 | ```sql
SELECT count(MRN) FROM ENC_DX_LTD b
WHERE b.dx_cd like '250%'
AND b.DSCH_DT – b.adm_dt>30;
``` | Number of diagnoses with diabetes where patient stayed more than 30 days. |
| Query 4 | ```sql
SELECT count(1) FROM ENC_CLIN_LTD a
WHERE a.prim_dsch_dx_cd='428.0'
AND a.mrn IN (SELECT mrn FROM enc_dx_ltd b
WHERE b.dx_cd='786.50'
AND b.DSCH_DT – b.adm_dt>30 );
``` | Number of patients who are diagnosed with CHF and who recorded a diagnosis of "Unspecified Chest Pain" where length of stay was greater than 30 days. |
| Query 5 | ```sql
SELECT count(distinct b.MRN)FROM ENC_ICD9_PROC_LTD b, ENC_ICD9_PROC_LTD a
WHERE a.MRN = b.MRN
AND a.ICD9_DT between to_date('1/1/2008','MM/DD/YYYY')and
to_date('12/31/2009', 'MM/DD/YYYY')
AND a.ICD9_CD in ('37.21', '37.22', '37.23')
AND b.ICD9_DT between to_date('1/1/2008','MM/DD/YYYY')and
to_date('12/31/2009', 'MM/DD/YYYY')
AND b.ICD9_CD = '88.72';
``` | Patients who had catheterization and transthoracic echo procedures within given date ranges |
| Query 6 | ```sql
SELECT count(1) FROM ENC_CLIN_LTD a
WHERE a.prim_dsch_dx_cd like'428%'
AND a.mrn IN (SELECT b.MRN FROM ENC_ICD9_PROC_LTD b
WHERE b.ICD9_DT between to_date('1/1/2008','MM/DD/YYYY')
and to_date('12/31/2009', 'MM/DD/YYYY')
AND b.ICD9_CD in ('37.21', '37.22', '37.23', '88.72'));
``` | Patients who had history of CHF, who also had catheterization or transthoracic echo procedures within a given date range |
| Query 7 | ```sql
SELECT count(distinct a.mrn) FROM ENC_DX_LTD a, ENC_CLIN_LTD b
WHERE a.mrn=b.mrn
AND a.encounter_no=b.encounter_no
AND b.prim_dsch_dx_cd = '707.15'
AND a.dx_cd like '250%'
AND a.dx_rank<>1
AND b.DSCH_DT – b.adm_dt>10;
``` | Number of patients who had primary diagnosis of "Ulcer of Other Part of Foot" and a non-primary diagnosis of diabetes, who also had length of stay more than 10 days. |
| Query 8 | ```sql
SELECT count(distinct a.mrn)FROM ENC_DX_LTD a, ENC_CLIN_LTD b
WHERE a.mrn=b.mrn
AND a.encounter_no=b.encounter_no
AND b.prim_dsch_dx_cd = '707.15'
AND a.dx_cd like '250%'
AND a.dx_rank<>1
AND b.DSCH_DT – b.adm_dt>10
AND a.mrn IN (SELECT c.MRNFROM ENC_ICD9_PROC_Ltd c
WHERE c.ICD9_CD
in ('37.21', '37.22', '37.23', '88.72'));
``` | Number of patients who had primary diagnosis of "Ulcer of Other Part of Foot" and a non-primary diagnosis of diabetes, whose length of stay was greater than 10 days and who had any history of catheterization or a transthoracic echo procedure |