asemio

Social impact applications of secure multi-party computation

Abstract

Data sharing between social impact organizations has the potential to amplify social impact by reducing redundant efforts and increasing capacity. Unfortunately, the risks of and restrictions on sharing personal data often limit this potential. New technologies such as secure multi-party computation (MPC) promise to mitigate many of these risks while retaining many of the benefits of traditional data-sharing solutions. In this paper we investigate the use of MPC to solve the record linkage problem of analyzing referral loops between service providers. Initial results indicate that the proposed MPC solution produces correct aggregate results compared to the status quo methods of manual and automatic analysis on plaintext.

Introduction

All communities grapple with complex social problems. All communities aim to improve outcomes for affected populations by streamlining community services. Data sharing offers promising support for these objectives, particularly for collective impact [7], the commitment of a group of important actors from different sectors to a common agenda for solving a specific social problem. Data sharing also supports data-informed decision making [6] by community foundations, state and local governments, and social impact investors, enabling them to rely on empirical data to support strategic decisions about how to address their communities' social issues.

Overview of data sharing

Community data sharing initiatives typically involve expensive governance cycles and complex legal frameworks that are designed to control risk [23]. Sharing of certain kinds of personal data is governed by laws and regulations such as FERPA [21] and HIPAA [22], which necessarily prioritize protecting sensitive data over enabling data sharing for community benefit. Organizations that serve vulnerable populations are particularly wary of privacy breaches, which not only violate federal law but also the ethical mandate of protecting those populations. Leaders are understandably cautious about investing time, energy, money, and trust in data sharing initiatives when the risks are so great.

Plaintext data sharing

Organizations that decide to share personally identifiable information about their constituents typically do so under the governance of a legal agreement, such as a HIPAA Business Associate Agreement (BAA), which extends liability for data breaches from one organization to another [22].

The data itself is usually copied from one organization to the other in plaintext form—e.g., as files of comma-separated values—using secure transport mechanisms such as HTTPS. The data may also be stored in encrypted form. In such a circumstance, both organizations would have the ability to decrypt and read the data in its plaintext form.

This form of data sharing, while common in practice, presents a number of challenges. Organizations often determine that the risks of data sharing outweigh the benefits, and are unwilling to assume the considerable liability imposed by a contract. In such cases, no agreement is established and data sharing does not occur. When a contract is established and a privacy breach happens, it is often difficult to determine who is responsible for the breach, and thus who is legally liable.

Secure multi-party computation

Secure multi-party computation is "a subfield of cryptography with the goal of creating methods for parties to jointly compute a function over their inputs while keeping those inputs private" [5].

The computed function varies between applications. One well-known study used MPC to analyze gender and ethnicity wage gaps among employers within the Greater Boston Area [1]. The researchers collected sensitive compensation data from privately held companies in order to calculate an aggregate statistic (a sum, the computed function) over the data points. Researchers could view the employee earnings totals aggregated across all companies, but the individual company aggregates remained private and were never revealed to any single party.

In the case of social services, the inputs are typically personal information about individuals, and the organization that collects the information is legally liable for its privacy. The computed functions may be statistics about individuals' data, such as average age or total number of children.

MPC is not interchangeable with methods that work on plaintext, as it is not intended to produce record-level results. MPC applications are designed to produce aggregate results in order to preserve individuals' privacy. In contrast, a more general but less private referral tracking system in plaintext would allow two organizations to identify, for example, individuals who were referred from one organization to the other but did not make an appointment. MPC would not allow this record-level output to be produced.

There are numerous implementations of MPC [20]. This paper will focus on the use of a single MPC technology, Cybernetica's Sharemind MPC Application Server [14].

A data sharing application

In order to explore the potential of secure multi-party computation in the Tulsa community, Asemio initiated a pilot project in collaboration with two social impact organizations in Tulsa, henceforth referred to as Data Provider A and Data Provider B. The data providers were interested in using secure multi-party computation to derive insights from shared data without sharing personally identifiable information. Both organizations rely on data-informed decision making, and obstacles to data sharing decrease their effectiveness in providing services to the community. On the other hand, both organizations also work with some of the most vulnerable populations in Tulsa. From both legal and ethical perspectives, there is a need for strong privacy guarantees around any data that is shared by either organization.

Referral loops

Data Provider A frequently refers individuals to Data Provider B for treatment. Both data providers use their own systems to record outgoing and incoming referrals, respectively. There is no existing process to report on these referrals and their outcomes or to distill these referrals from all other referrals in either organization in order to analyze them.

Automatically linking outgoing referrals from Data Provider A to incoming from Data Provider B with software would require the organizations to electronically share personally identifiable information. This is usually accomplished with contractual agreements around plaintext data

sharing, as described above. However, due to the particular sensitivity of the data involved, plaintext data sharing of personally identifiable information was considered too high a risk.

MPC technologies, in contrast, could mitigate much of this risk by allowing data to leave each organization's trusted computing base only in encrypted form and disseminating analysis results only in aggregate.

Our application required counting outgoing referrals recorded by Data Provider A that have a corresponding incoming referral recorded by Data Provider B. The goal was to identify how many referral loops are being "closed" or not, rather than identify specific individuals. More specific aggregate information could be gained by constraining the sets of referrals considered—limiting them to a specific reason type, for example—while still insisting that the aggregates be broad enough to prevent reidentification of individuals [8].

Contributions of this paper

The main contribution of this paper is the procedural and technical description of a software system and a pilot test case for linking records containing highly sensitive personal information by using secure multi-party computation technologies. Our aim was to prove that such a system could work for a real-world application, providing tangible benefits while mitigating risk.

We relied on existing tools and algorithms to accomplish this end, instead of developing new algorithms [1]. This paper will not provide detailed analyses of the security or performance of these tools and algorithms; these analyses can be found in the referenced literature [20].

Solution requirements

Matching outgoing to incoming referrals is a record linkage problem: finding records in two data sets that refer to the same individual [3]. The process can be reduced to three essential steps:

1. **Cleaning and standardizing source records** (aka *data preprocessing*, the *transform* step of Extract-Transform-Load (ETL), or *normalization*) to ensure that the fields of different records are syntactically and semantically comparable
2. **Comparing records**
3. **Classifying record comparisons** as matches or non-matches

The theory and practice of record linkage has been studied extensively, and numerous algorithms exist for every step.

Cleaning and standardizing records

Source records are often presented with nonstandard formats or unparsed text. Dates and phone numbers should be parsed and then reformatted in a consistent way across all records so that they are comparable between records. Freetext address lines should be parsed into their

components (e.g., house number, apartment number, etc.) and separated into different fields. Values in other fields should be controlled: transforming a source gender “Male” into “m,” for example. Most strings should be converted to either entirely upper- or lowercase.

Personal names require special handling, due to cultural variations, frequent misspellings, and other factors [9]. One technique is to convert freetext names into phonemes, using algorithms such as Soundex [10], Double Metaphone [17], and NYSIIS [18].

Comparing records

Once all records are cleaned and standardized, records from one data set are compared to records in the other data set. There are potentially *n x m* record-to-record comparisons, where *n* and *m* are the number of records in each of the respective data sets (i.e., pairwise comparison). In practice, techniques such as blocking can substantially reduce the number of comparisons for large record sets [3].

When comparing two records, the fields of the records are usually considered independently of other fields. First name is compared to first name, phone number is compared to phone number, and so on. The algorithm for comparing a field in two records often depends on the semantics of the field. For example, we would expect that gender, being a controlled value (e.g., male, female, other, unknown), would exactly match or not match at all. Names, on the other hand, may be compared by string edit distance [11], in order to consider fuzzy (nonexact) matches. The functions for comparing the fields of two records are called *comparators*.

In a few cases, comparing two or more fields in conjunction makes more sense than comparing the fields independently. For example, addresses may be pre-geocoded into latitude and longitude fields. These coordinates can then be compared by distance on the Earth’s surface using the Haversine formula[1].

The result of a comparator is a distance in the range [0, 1]. By convention, a distance of 0 is considered an exact match, 1 is considered a non-match, and values in between are considered measures of relative similarity.

The comparison step of the record linkage process applies the comparators to record pairs. The distances are then collected in a *comparison vector* for each pair of records that was compared. The set of all comparison vectors for all compared records is called the *comparison space*.

Classifying record comparisons

The final step is to classify comparison vectors (and, by extension, record pairs) as matches or non-matches. There are two main approaches to this problem: deterministic and probabilistic.

[1] https://en.wikipedia.org/wiki/Haversine_formula

Deterministic algorithms assess whether a comparison vector represents a match or a non-match by examining specific elements of the comparison vector and making a discrete, “all-or-nothing” classification. For example, the National Cancer Institute uses a deterministic algorithm that consists of a sequence of deterministic comparisons in two rounds [12]. In the first round, two records must match on Social Security number and one of the following:

- First and last name, allowing for fuzzy matches, such as nicknames
- Last name, month of birth, and gender
- First name, month of birth, and gender

If the Social Security number is missing or does not match, or two records fail to meet the initial match criteria, they may be declared a match if they agree on the criteria in a second round of deterministic linkages, in which two records must match on last name, first name, month of birth, gender, and one of the following:

- Seven to eight digits of the Social Security number
- Two or more of the following: year of birth, day of birth, middle initial, or date of death

Probabilistic algorithms, in contrast, consider the entire comparison vector as potentially discriminative, and classify vectors as matches, non-matches, or potential matches, according to the likelihood that two records are a true match based on whether they agree or disagree on the various identifiers. (Optionally, potential matches can be further reduced to matches or non-matches according to some likelihood threshold.)

Probabilistic algorithms can be further categorized, such as those that rely on predetermined likelihood ratios, including the well-known Fellegi-Sunter algorithm [13]. Other algorithms rely on machine learning to classify comparison vectors as matches and non-matches, which can be reduced to a conventional binary classification problem[2].

The final result of the three steps is a set of record pairs that have been classified as matches. A multi-party computation would summarize this set to protect the privacy of individual records.

Regulatory requirements

Many schools and social service providers that work with students are governed by the Family Educational Rights and Privacy Act (FERPA) [21], a federal law that protects the privacy of student education records. FERPA allows a school to disclose, without parental consent, information that the school categorizes as *directory information*, such as a student's name, address, telephone number, date and place of birth, honors and awards, and dates of attendance. However, services that work with vulnerable populations frequently choose not to disclose directory information, and thus consider all personal information about a student to be private. The application considered by this paper falls under this category.

[2] https://en.wikipedia.org/wiki/Binary_classification

Other providers are governed by the Health Insurance Portability and Accountability Act of 1996 (HIPAA) and its associated Privacy Rule, which regulates the use and disclosure of protected health information (PHI) held by “covered entities,” such as many medical service providers. HIPAA permits covered entities to disclose PHI to certain parties to facilitate treatment operations without a patient's express written authorization. The application considered by this paper also falls under this category.

Furthermore, HIPAA rules require systems handling PHI to implement “reasonable and appropriate” security measures. This is widely interpreted to mean the use of encrypted data transports such as HTTPS as well as the encryption of data at rest.

Data syntax and semantic requirements

The Data Provider A and Data Provider B referral data sets each consist of single comma-separated value (CSV) files, with each row/record in a file representing a single individual. The Data Provider A individuals had been referred to Data Provider B according to the Data Provider A data system, and the Data Provider B individuals were those who came to Data Provider B for services as a result of a referral from Data Provider A according to the Data Provider B data system.

Each record must provide enough information about an individual to conclude whether two records, one from each data set, refer to the same individual. At a minimum, we required each record to have the first and last names of the individual, date of birth, and address. We made other fields optional, such as gender, race, and phone number. We assumed the semantics of the fields to be comparable (e.g., the same definition of first and last name). This is a reasonable assumption given that all of the individuals reside in the Tulsa area [9]. We also assumed that the providers’ respective record sets contained only a single record per real-world individual, so there was no duplication within a record set.

Deployment requirements

Secure multi-party computation technologies encrypt data before it leaves a data provider’s trusted computing base, which means that the data must be extracted, transformed, and cleaned within that base. With MPC there is additional emphasis on the preprocessing step of record linkage, outlined above, because the comparison step is limited in the types of comparators it can apply to encrypted records. For example, a comparator cannot parse out the components of a field in order to isolate the most discriminatory.

The data providers require a program that will accept their input CSV files, transform and clean records, and upload them to the MPC platform, all with minimal effort on the part of the provider. There can be no plaintext-handling intermediaries between the data provider and the MPC system. We can assume the data providers are technically sophisticated enough to configure the program to map source fields to well-known target fields (e.g., first and last name), with little/no parsing required.

The remainder of the record linkage process (steps 2 and 3) can be driven by a third party who does not (necessarily) provide records itself. Because the process works only on encrypted data, the third party ("analyst") need not be privy to the contents of the records. Asemio played this role in the pilot project.

Proposed solution framework

Our proposed solution consists of a platform for secure multi-party computation; a library and set of command-line programs for transforming, cleaning, and uploading data sets to the platform; and additional command-line programs for linking records from uploaded data sets and reporting aggregates on the linked data sets.

Sharemind

We selected the Sharemind platform as our foundational technology for secure multi-party computation [14]. Sharemind is a distributed database and application server that can enforce privacy controls and efficiently store and process encrypted data. Within Sharemind, no single party can access private values or perform arbitrary processing on them.

Sharemind is built on a form of encryption known as *secret sharing* [15] . Data are uploaded to the Sharemind platform in encrypted form by splitting record field values into random pieces that are distributed among several computation nodes. None of the individual pieces provide any information about the original value.

Data encrypted by secret sharing has homomorphic properties that allow computations on it without the need to decrypt it first. Sharemind computation servers engage in cryptographic protocols to compute on encrypted values. During this process, no values are decrypted. Intermediate computation results also stay secret. Computation results can be published outside the system, provided that all computation nodes agree to do so. The architecture of the Sharemind platform is depicted in Figure 1, below.

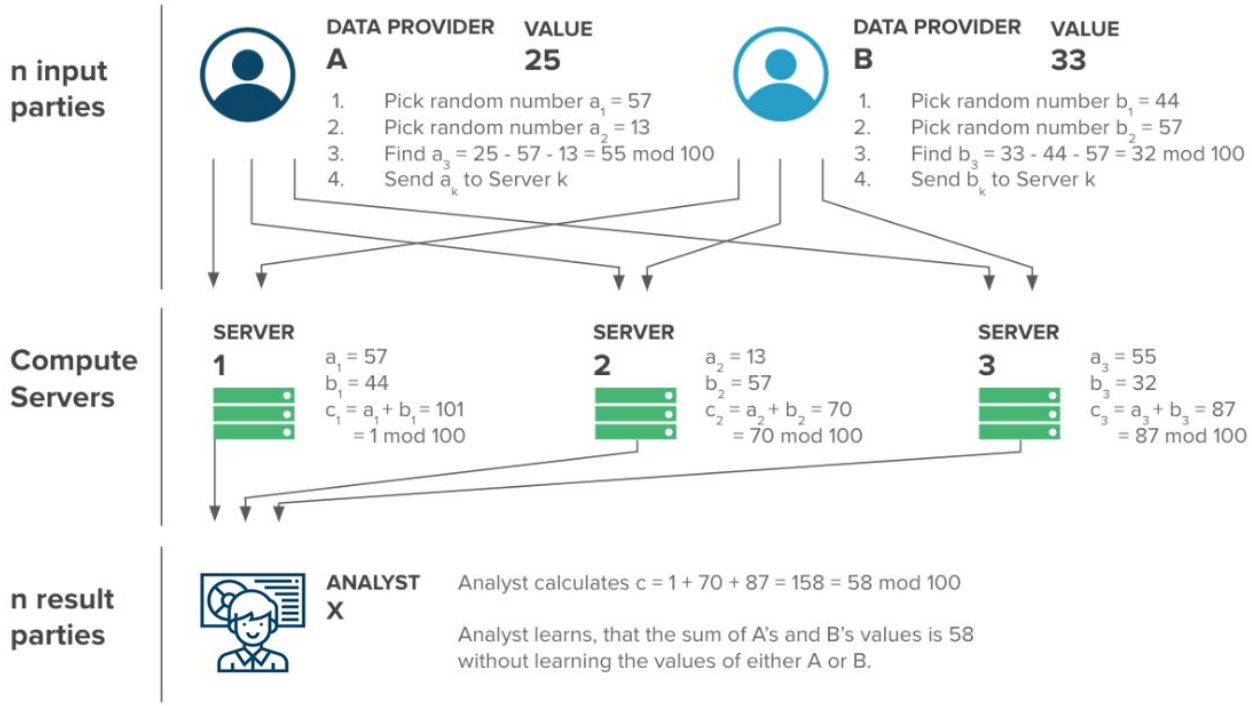

Figure 1: Overview of the Sharemind architecture

The system consists of:

- *n* input parties, in our case the two data providers.
- *n* result parties/analysts, who trigger computations and receive the explicitly published (plaintext) results of a computation.
- Exactly three compute servers (computation parties), which receive secret shares and perform computations on behalf of analysts. Computation parties are usually hosted by three separate legal entities that have incentives (legal or otherwise) not to collude with each other. All computation parties must all agree on which computations may be executed.

Sharemind computations on encrypted data are implemented using the SecreC programming language[3]. SecreC programs clearly tag public and private data and explicitly show locations where private data are made public. Figure 2, below, shows a sample SecreC program.

[3] https://sharemind-sdk.github.io/

```
1  void main ( ) {
2    uint64  arg1 = arguments (“arg1”);  // Arguments can be public
3    pd_shared3p uint64 [ [1] ]  arg2 = arguments(“arg2”);  // Or private
4
5    // Computation results are also private.
6    pd_shared3p bool [ [1] ]  result = arg2 <= arg1;
7
8    // Results are explicitly published to the client.
9    publish(result, “result”);
10 }
```

Figure 2: Sample SecreC program

Privacy-preserving record linking with Sharemind

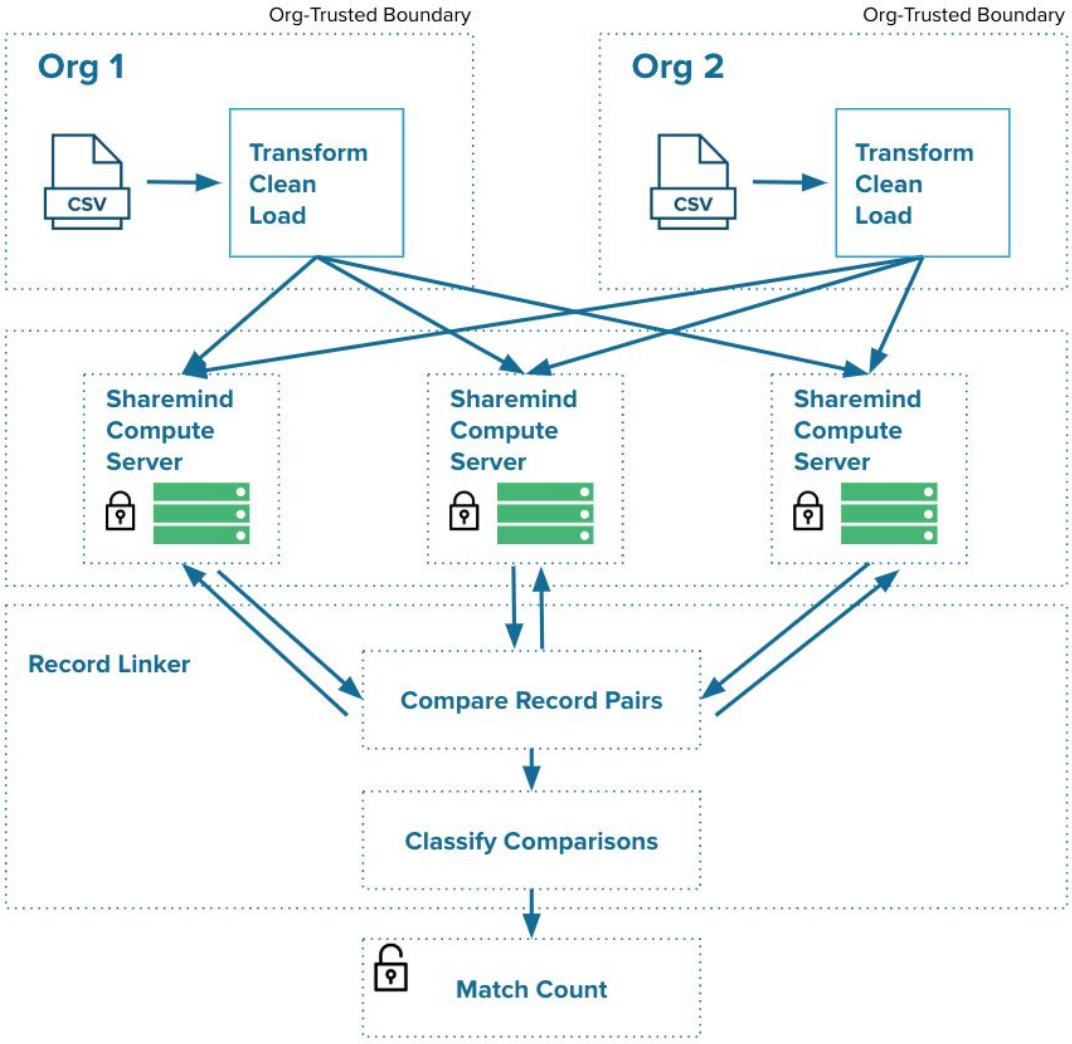

Figure 3: Record linking workflow

The three steps of the record linking workflow are illustrated in Figure 3, above. The following sections will describe the components of the workflow in detail.

Data provider transform-clean-load tools

For this project Asemio developed a set of tools to ease a data provider's access to the Sharemind platform. The tools consist of libraries and command-line programs written in Python, C++, and SecreC. They are designed to run in Linux environments, because of dependencies on Sharemind client libraries.

We use Python to transform and clean records within the data provider's trusted computing base before uploading them in encrypted form to Sharemind. The Python transformers include custom code for enforcing controlled vocabularies (e.g., gender, race, language) as well as third-party libraries and tools such as libphonenumber[4], libpostal[5], python-dateutil[6], and the fuzzy library of string-to-phoneme algorithms[7].

Cleaned records are loaded into Sharemind tables by invoking a Sharemind controller from Python. This is a program written in C++ that utilizes the Sharemind client libraries to communicate with the Sharemind compute servers. Python passes data via subprocess to the C++ controller program, which in turn invokes a SecreC program on the compute servers that accepts the data as arguments and populates Sharemind tables with it. Each record set has its own table in Sharemind, with one record per row.

Record linkage tools

Once the data providers have loaded their record sets into Sharemind, a set of command-line programs developed by Asemio drives the record linking process between two Sharemind tables. The process of uploading and linking records is intentionally designed to be executed infrequently, quarterly or even annually, and to consider entire record sets at once. It does not attempt to accommodate ongoing updates to record sets, which would require overwriting rows in tables, unlinking, and relinking, etc. This functionality, while important in many applications of record linkage, was outside the scope of this pilot.

Comparing records

Records from Data Provider A and Data Provider B consisted of string fields, such as first and last name, and integer fields, such as birthdate month. Fields were compared with one of two simple operations: exact match or Levenshtein string edit distance[8]. The choice of operations depended on the semantics of the field. For example, names were compared by edit distance, while phonemic translations of names were tested for exact matches. These operations were

[4] https://github.com/daviddrysdale/python-phonenumbers
[5] https://github.com/openvenues/libpostal
[6] https://pypi.org/project/python-dateutil/
[7] https://pypi.org/project/Fuzzy/
[8] https://en.wikipedia.org/wiki/Levenshtein_distance

implemented in a SecreC program that compared specific sets of records from the respective tables. (Records were delineated by their row index in the respective tables, which we called a *record ID*.)

The public result of the SecreC program was a set of comparison vectors for the record pairs. We considered that these comparison vectors did not constitute protected information, and thus could be allowed to leave the Sharemind system. The main alternative would have been to do all comparison and match/no-match classification in SecreC, which would have been considerably more complex to implement.

The process of comparing all *n x m* record pairs from the two tables was implemented in a producer/consumer fashion, so that it could be parallelized to any number of consumers and restarted if a process failed. Both properties were important, because each record-to-record comparison required seconds to execute in SecreC on the Sharemind test virtual machine. (As a general rule, computations in Sharemind are 100 to 1,000 times slower than equivalent computations on plaintext. Additionally, the test virtual machine is considerably slower than production servers would be.) The producer process enqueued batches of record ID pairs to a Redis[9] queue as "jobs." The consumer processes dequeued these pairs, executed the comparisons by invoking the SecreC program, and wrote the resulting comparison vectors back to a Redis list. The latter was monitored for completion of all *n x m* comparisons. Upon completion the comparison space was written to a text file to be taken up by the next step, classification.

Classifying comparison vectors

Given the set of comparison vectors from the comparison step, the classification step must decide which record pairs are matches. We experimented with a number of algorithms to do this, from simple deterministic algorithms (match first and last name and date of birth exactly) to probabilistic algorithms that rely on machine learning. We ultimately settled on a machine learning approach, since it proved to be the most robust. In this we were able to take advantage of the dedupe Python library[10], which is designed for record linking. The library uses an L2 regularized logistic regression classifier to classify matches and non-matches with a specific likelihood threshold in lieu of classifying comparisons as potential matches.

The dedupe library requires the match classifier to be trained ahead of time (supervised learning). An interactive training program presents a human supervisor with two sample records and asks the person to decide if the pair is a match or a non-match, or to mark "unsure." This is obviously impossible to do accurately with encrypted records, so we trained the classifier using plaintext record sets that we considered representative of the actual, private record sets from the data providers.

[9] https://redis.io/
[10] https://github.com/dedupeio/dedupe

The result of the classification step is a count of linked records, which corresponds to the number of individuals that were present in both record sets.

Methods

Given the proposed solution framework, we sought to answer the question: does privacy-preserving record linking identify referral loops from one service organization to another as accurately as linking plaintext records?

Record linking methods

In order to answer that question, we compared the result (an overlap count) of the secure multi-party computation record linking process described above to results from other methods, using real-world data sets from Data Provider A and Data Provider B. The methods are summarized in Table 1, below.

Method	Description
Human	Sorting and collating plaintext records and manually inspecting them; performed by multiple people independently, with consensus result
Plaintext	Comparing record fields in plaintext using the limited operations (exact match and string edit distance) supported by the privacy-preserving system and classifying them with the dedupe library's machine learning algorithm
Privacy-preserving	Comparing encrypted record fields in Sharemind and classifying them with the dedupe library's machine learning algorithm, as described above

Table 1: Record linking methods

The labor-intensive human method was employed as a control. The plaintext method was intentionally designed to mimic the limited comparison operations of the privacy-preserving method, rather than take advantage of the full range of dedupe's plaintext comparators (e.g., comparators that parse plaintext addresses on demand).

Data sets

We utilized two data sets, one from Data Provider A and one from Data Provider B. The data sets are summarized in Table 2, below. We have intentionally excluded the number of records in each data set to avoid identifying the data providers.

Organization	Description
Data Provider A	Individuals in the Tulsa area referred to Data Provider B for a specific treatment between August 2015 and May 2018
Data Provider B	Individuals in the Tulsa area referred from Data Provider A for the specific treatment who made an appointment between August 2015 and May 2018

Table 2: Data set summary

For the purposes of this study, Asemio obtained plaintext versions of the private data sets in order to apply the human and plaintext methods of record linking described above, as well as to test the privacy-preserving method without requiring technical effort from Data Provider A or Data Provider B. A real-world deployment of the proposed solution would not require data providers to share plaintext, as described previously.

Prior to conducting the study, Asemio enforced the requirement that an individual would be represented only once within a data set.

Common demographic fields were transformed and cleaned from the two data sets. Fields that could not easily be compared in a privacy-preserving manner were excluded. The resulting fields are summarized in Table 3, below.

Field name	Description	Data type	Comparison type
PersonBirthDateDay	Birthdate day	Integer	Exact
PersonBirthDateMonth	Birthdate month	Integer	Exact
PersonBirthDateYear	Birthdate year	Integer	Exact
PersonContactTelephoneNumberNationalNumber	Parsed and reformatted contact telephone number with country code (if any) excluded	String	Exact
PersonFirstName	First name in lowercase	String	String edit distance
PersonFirstNameDoubleMetaphonePrimary	Double Metaphone phonemic translation of first name	String	Exact
PersonFirstNameDoubleMetaphoneSecondary	Double Metaphone phonemic translation of first name	String	Exact
PersonFirstNameNYSIIS	NYSIIS phonemic translation of first name	String	Exact

PersonFirstNameSoundex	Soundex phonemic translation of first name	String	Exact
PersonGender	Controlled gender values: Female, Male, Other	String	Exact
PersonLanguage	ISO 639 3-letter codes (e.g., “eng”)	String	Exact
PersonLastNameDoubleMetaphonePrimary	Double Metaphone phonemic translation of last name	String	Exact
PersonLastNameDoubleMetaphoneSecondary	Double Metaphone phonemic translation of last name	String	Exact
PersonLastNameNYSIIS	NYSIIS phonemic translation of last name	String	Exact
PersonLastNameSoundex	Soundex phonemic translation of last name	String	Exact
PersonLocationCity	Primary address city	String	String edit distance
PersonLocationPostalCode	Primary address ZIP code	String	String edit distance
PersonLocationState	Primary address state code (e.g., “OK”)	String	Exact
PersonMiddleInitial	Middle initial in lowercase	String	Exact
PersonMiddleName	Middle name in lowercase	String	String edit distance
PersonMiddleNameDoubleMetaphonePrimary	Double Metaphone phonemic translation of middle name	String	Exact
PersonMiddleNameDoubleMetaphoneSecondary	Double Metaphone phonemic translation of middle name	String	Exact
PersonMiddleNameNYSIIS	NYSIIS phonemic translation of middle name	String	Exact
PersonMiddleNameSoundex	Soundex phonemic translation of middle name	String	Exact

Table 3: Demographic fields compared between records

Infrastructure

We used desktop-based spreadsheet applications to apply the human method. The automatic (plaintext and privacy-preserving) methods were executed on a Windows workstation that also hosted a Sharemind test virtual machine provided by Cybernetica. This virtual machine hosted the three compute servers required by Sharemind and executed our Sharemind controllers, acting as both client and server. As mentioned previously, in a real-world deployment the servers would be hosted on three different machines, either self-hosted or in the cloud, and the client(s) would run on still other machines. Emulating a production deployment was not a goal of the pilot project.

Results

All three methods (human, plaintext, privacy-preserving) produced the same count of individuals present in both data sets. We confirmed this by cross-checking the record IDs linked by the privacy-preserving method against the plaintext versions.

Discussion

While a single experiment is not sufficient to prove the correctness of the privacy-preserving method, the results increased our confidence in the viability of the proposed solution for the problem of referral loop feedback, and suggested that further effort in this direction is warranted.

Future work

While the Sharemind platform is commercial ready, the various libraries and command-line programs implemented solely for this project are proof-of-concept rather than production quality. Significant effort would be required to make them more user-friendly, particularly to nontechnical or semitechnical users at data providers, who would be expected to transform data and upload it to Sharemind.

The test data sets were relatively clean compared to real-world data sets [16], with few misspellings and other errors of the kind sophisticated plaintext record linkers are designed to detect and accommodate. This cleanliness can be attributed to good practice on the part of the providers as well as frequent cross-checking of records against other databases.

Due to the limited number and sophistication of record comparison operations in Sharemind, we would expect the privacy-preserving method's results to be less accurate when the data is dirtier. We believe we could at least partially compensate for this through several means:

- Training the classifier on additional and more varied data sets

- Using data dictionaries for common personal names, cities, etc. to provide more potential exact matches while falling back to edit distance
- Incorporating more fields (signals) for the classifier to work with, including:
    - Parsing personal names, especially hyphenated names
    - Providing Hispanic/Non-Hispanic and ethnic origin indicators
    - Geocoding addresses and computing distance between coordinates
- Adding string comparators that are tuned to the semantics of individual fields, particularly names, rather than relying on a single general string edit distance function (Levenshtein distance) [11]
- Indicating potential matches in the results rather than binary match/non-match decisions

Legal questions

As noted previously, the study was conducted within the scope of legally binding data sharing agreements between the data providers and Asemio. These agreements are always required when sensitive data leaves an organization's trusted computing base in plaintext form, even if the transport is encrypted and the data will be encrypted at rest. An open question is whether data residing outside an organization's trusted computing base in inaccessible form still needs to be governed by contractual agreements. From a legal perspective, this largely rests on the question of whether encrypted PHI is still PHI when there are no means for the receiving parties to decrypt the data or otherwise expose it.

In 2014–2015 Cybernetica conducted a study analyzing highly sensitive educational and tax data, which there are significant legal barriers to sharing [2]. The sensitive data was uploaded to the Sharemind application and analyzed in encrypted form, with all parties agreeing on the specific study plan. A subsequent legal analysis [4] suggested that a reasonable interpretation of European Union data protection laws would indicate that the study did not process personal data. However, the analysis concluded that, under current EU law, no general guidance could be given, as each study plan would have to be checked to make sure that the Sharemind platform is not induced to publish personally identifiable information, only summary statistics [2].

Even if systems that use privacy-preserving technology such as MPC still require data sharing contracts under current legal interpretations, the risk mitigation effects of the technology may encourage organizations to share data and allow analysis in encrypted form that they might not have shared in plaintext. In this case, it would be useful to create a "legal toolkit" that explains the platform, lists examples of its successful use (e.g., the aforementioned Cybernetica study), and provides templates for contractual data-sharing agreements with language adapted to the mechanics of the technology.

The standardization of secret sharing/threshold cryptography and its acceptance as a cryptographic primitive would also ease adoption of Sharemind and related technologies under current legal frameworks [19].

Summary and conclusion

In this study, we investigated the use of privacy-preserving, secure multi-party computation technologies as a means of sharing highly sensitive personal data between social service organizations. The specific purpose of sharing the data was to analyze the incidences of successful referrals from one organization to another, with the general goal of improving service delivery.

The results are encouraging if not definitive. The automatic privacy-preserving analysis of encrypted data produced the same result as a manual control analysis on plaintext, indicating that the end result, a count of unique individuals represented in two data sets, was correct. Ensuring correctness in a controlled study was a necessary first step to ascertaining the viability of the technology for larger and more complex applications. We believe that secure multi-party computation has the potential to mitigate many of the risks of sensitive data sharing while preserving many of its benefits, and we plan to pursue this technology further.

References

1. Lapets, A., Volgushev, N., Bestavros, A., Jansen, F., Varia, M. (2016). *Secure Multi-Party Computation for Analytics Deployed as a Lightweight Web Application.* Technical Report BU-CS-TR 2016-008. Boston University: Computer Science Department.

2. Bogdanov, D., Kamm, L., Kubo, B., Rebane, R., Sokk, V., & Talviste, R. (2016). Students and taxes: a privacy-preserving study using secure computation. *PoPETs, 2016*, 117-135.

3. Dusetzina, S. B., Tyree, S., Meyer, A. M. (2014). *Linking data for health services research: A framework and instructional guide*. Rockville, MD: Agency for Healthcare Research and Quality (US).

4. Damiani, E., Bellandi, V., Cimato, S., Gianini, G., Spindler, G., Grenzer, M. (2014). Risk assessment and current legal status on data protection. Retrieved from http://practiceproject.eu/downloads/publications/D31.1-Risk-assessmentlegal-status-PU-M12.pdf

5. Secure Multi-Party Computation. (n.d.). Wikipedia: The Free Encyclopedia. Retrieved November 9, 2018, from en.wikipedia.org/wiki/Secure_multi-party_computation

6. Shen, J. (2016). Data-informed decision making on high-impact strategies: Developing and validating an instrument for principals. *The Journal of Experimental Education, 80*(1), 1-25.

7. Kania, J., & Kramer, M. (winter 2011). Collective impact. *Stanford Social Innovation Review 9*(1), 36-41.

8. Sweeney, L. (2002). K-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10*(5), 557-570.

9. Personal names around the world. (n.d.). World Wide Web Consortium. Retrieved November 9, 2018, from https://www.w3.org/International/questions/qa-personal-names

10. Soundex System. National Archives. Retrieved November 9, 2018, from https://www.archives.gov/research/census/soundex.html

11. Cohen, W. W. (2003). A comparison of string distance metrics for name-matching tasks. *Proceedings of the 2003 International Conference on Information Integration on the Web,* 73-78.

12. Warren, J. L., Klabunde, C. N., Schrag, D. (2002). Overview of the SEER-Medicare data: content, research applications, and generalizability to the United States elderly population. *Med Care, 40*(8 Suppl), IV-3-18.

13. Fellegi, I. P., & Sunter, A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association*, *64*(328), 1183-210.

14. Bogdanov, D., Laur, S., Willemson, J. (2008). Sharemind: A framework for fast privacy-preserving computations. In S. Jajodia & J. Lopez (eds.), *Computer Security - ESORICS 2008. Lecture Notes in Computer Science, (5283)*. Berlin: Springer.

15. Schoenmakers, B. (1999). A simple publicly verifiable secret sharing scheme and its application to electronic voting. In M. Wiener (ed.), *Advances in Cryptology — CRYPTO' 99. Lecture Notes in Computer Science, 1666*. https://doi.org/10.1007/3-540-48405-1_10

16. Christen, P. (2008). Febrl: a freely available record linkage system with a graphical user interface. In J. R. Warren, P. Yu, J. Yearwood, & J. D. Patrick (eds.), *Proceedings of the second Australasian workshop on health data and knowledge management*, *80*, 17-25. Darlinghurst, Australia: Australian Computer Society, Inc.

17. Philips, L. (1990). Hanging on the Metaphone. *Computer Language Magazine 7*(12), 39-44.

18. Taft, R. L. (1970). *Name Search Techniques*. Albany, NY: New York State Identification and Intelligence System.

19. Barker, E. (2016). *Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms.* National Institute of Standards and Technology. Retrieved November 12, 2018, from https://csrc.nist.gov/publications/detail/sp/800-175b/final

20. Archer, D. W., Bogdanov, D., Lindell, Y., Kamm, L., Nielsen, K., Illeborg Pagter, J., Smart, N. P., Wright, R. N. (2018). From keys to databases—real-world applications of secure multi-party computation. *The Computer Journal, 61*(12), 1749-1771.

21. The Family Educational Rights and Privacy Act. 20 U.S.C. § 1232g; 34 CFR Part 99.

22. The Health Insurance Portability and Accountability Act of 1996. Pub. L. 104-191. Stat. 1936.

23. Allen, C. (2014). *Data Governance and Data Sharing Agreements for Community-Wide Health Information Exchange: Lessons from the Beacon Communities*. Washington, DC: EGEMS.