

# On Chameleon Pseudonymisation and Attribute Compartmentation-as-a-Service

Anne V. D. M. Kayem<sup>a</sup>, Nikolai J. Podlesny and Christoph Meinel

*Hasso-Plattner-Institute, University of Potsdam, Prof.-Dr.-Helmert Str. 2-3, 14482 Potsdam, Germany*

**Keywords:** Privacy, Privacy Enhancing Technologies, Pseudonymisation, Data Transformation, Anonymisation, Compartmentation.

**Abstract:** Data privacy legislation and the growing number of security violation incidents in the media, have played a key role in consumer awareness of data protection. Furthermore, the digital trail left by activities such as online purchases, websites browsed, and/or clicked advertisements yield behavioural information that is useful for various data analytics operations. Analysing such information in a privacy-preserving way is useful both in satisfying service level agreements and complying with privacy regulations. Pseudonymisation and anonymisation have been widely advocated as a means of generating privacy-preserving datasets. However, each approach poses drawbacks in terms of composing privacy-preserving datasets from multiple distributed data sources. The issue is made worse when the owners of the datasets co-exist in an untrusted environment. This paper presents a novel method of generating privacy-preserving datasets composed of distributed data in an untrusted scenario. We achieve this by combining cryptographically secure pseudonymisation with data obfuscation and sanitisation. The pseudonymisation and compartmentation are outsourced to a central but fully oblivious entity that can blindly compose datasets based on distributed sources. Controlled non-transitive join operations are used to ensure that the published datasets do not violate the contributing parties' privacy properties. As a further step, the service provider will employ obfuscation and sanitisation to identify and break functional dependencies between attribute values that hold the risk of inferential disclosures. Our empirical model shows that the overhead due to cryptographic pseudonymisation is negligible and can be deployed in large datasets in a scalable manner. Furthermore, we are able to minimise information loss, even in large datasets, without impacting privacy negatively.

## 1 INTRODUCTION


In various domains, such as personalised healthcare, the notion of secure data sharing has become ever more tightly intertwined with privacy. Legislation such as Europe's General Data Protection Regulation and the US Consumer Privacy Act bring the issue of data privacy to the fore, with violation penalties being set to as high as EUR 20 million or 4% of the annual turnover of the preceding financial year depending on which is higher. Yet, as mainstream media stories indicate, providing firm guarantees of data privacy remains a challenging problem.

### 1.1 Context and Motivation

Access to personal data, has and continues to play an important role in the software services industry in

general. For instance, access personal data is useful in supporting data analytics tasks that support service delivery in the marketing and healthcare domains. Data is as such, increasingly crucial to providing dependable services.

Pseudonymisation and anonymisation are typically advocated as methods of transforming data to protect user privacy. Pseudonymisation holds the advantage of generating datasets that can be easily reverted to the original form to crosscheck the validity of results from data analytics processes. However, pseudonymisation has been criticised for resulting in datasets that are vulnerable to re-identification attacks (Sweeney, 2000; Sweeney, 2002a). Furthermore, simply removing attributes that are likely to identify an individual directly does not protect against re-identification (Sweeney, 2002a; Sweeney, 2002b). A plethora of solutions have been proposed in recent years to address this problem (Sweeney, 2002a;

<sup>a</sup>  <https://orcid.org/0000-0002-6587-5313>

Machanavajjhala et al., 2007; Li et al., 2007a; Dwork et al., 2009; Podlesny et al., 2019a). These solutions address various limitations centered around the Sweeney et al. (Sweeney, 2002a) approach to anonymising data. However, as Lehmann (Lehmann, 2019) pointed to these solutions are not well adapted to generating privacy-preserving datasets based on combinations of subsets of data drawn from multiple distributed sources.

Successfully anonymising subsets of data drawn from data sets such as these that contain a high proportion of sensitive data are impractical because any guarantees of privacy would come at the cost of high levels of information loss (Podlesny et al., 2018; Podlesny et al., 2019a). High information loss negatively impacts query result accuracy and veracity, and this can impede research that is dependent on data analytics (e.g. drug trials, personalised prescriptions, ...). A further issue is supporting privacy with cryptographic operations, guaranteeing security, is computationally intensive for large high dimensional datasets (Lehmann, 2019).

## 1.2 Problem Statement

The problem with which we are faced is generating privacy-preserving subsets of data from datasets containing a high degree of sensitive information in an untrusted distributed environment (i.e. the data owners do not trust each other). The central premise here is to do this both a-priori, to create generic data subsets, or on-demand (a-posteriori) - to tailor the generated subset of data to a specific request.

## 1.3 Contributions

We propose a solution for generating low information loss privacy-preserving datasets that involves maintaining as much information from the original dataset as possible without creating vulnerabilities to re-identification. To do this, we combine elements from unlinkable pseudonymisation (Lehmann, 2019) and attribute compartmentation (Podlesny et al., 2019a). We employ three (3) components namely, an *Obfuscator*, a *Converter* and a *Sanitizer*, to ensure that the data remains private.

The *Obfuscator* plays the role of transforming the dataset to prevent re-identifications due to inferences drawn from functional dependencies between attribute values. As a further step, the *Obfuscator* analyses the dataset to discover all quasi-identifiers (functional dependencies) that are likely to result in inferences.

The *Converter* is a central service that handles the

output from the *Obfuscator*. Its job is to derive and convert pseudonyms, and create a *joined* version of attributes based on the request or usage scenario of the data. The *Converter* performs these transforms blindly in that it stores no knowledge of the data sets. To create a subset of data, the *Converter* transform the *Chameleon Pseudonyms*, associated with the attributes that are to place in the requested data subset, to ensure that pseudonyms associated with the same record are mapped to the same value.

The *Sanitizer* serves as an internal validator or verifier to test the data that is to be shared with the data processor to ensure that it is free of elements that can be exploited for inferences.

## 1.4 Outline of the Paper

The rest of the paper is structured as follows. We discuss related work in Section 2, focusing on anonymisation, and pseudonymisation concerning generating privacy-preserving datasets from distributed sensitive data. In Section 3, we briefly present background notions on how unlinkable pseudonymisation and attribute compartmentation work, and then proceed to present our proposed mechanism for generating low information loss privacy-preserving datasets from distributed sensitive data. We discuss results from our empirical model in Section 4 and offer conclusions as well as avenues for future work in Section 5.

## 2 RELATED WORK

A plethora of work exists on the subject of transforming data for privacy. One of the first approaches, namely pseudonymisation, consisted of replacing personally identifying information in shared published datasets with a pseudonym. In Europe for instance, the Data Protection Working Party as an independent advisory body to the European Commission has outlined various data sanitisation methods for protecting individuals' data (Party, 2014). Pseudonymisation and anonymisation have as such been promoted as a means of generating privacy-preserving datasets (McCallister et al., 2010).

In the naive form, pseudonymisation consists of replacing an attribute's real value with a false or pseudonymous value. The idea is to use values that are difficult to link back to the original values. As Sweeney (Sweeney, 2000) pointed out, simply pseudonymising the data or eliminating direct identifying attributes (e.g. name and ID number) is not to prevent re-identification. Furthermore, applying pseudonymisation in the context of dis-

tributed databases is impractical in providing firm assurances of meeting the cooperation parties' privacy service level agreements of the cooperating parties (Camenisch and Lehmann, 2015). Camenisch and Lehmann (Camenisch and Lehmann, 2015) proposed alleviating this issue with an (un)linkable pseudonym system that overcomes these limitations. Additionally, it enables controlled privacy-friendly exchange of distributed data in untrusted environments yet. However, as mentioned before, the issue that remains is how to generate and provision the pseudonyms securely.

In recent work, Lehmann (Lehmann, 2019) addresses the issue of generating and provisioning pseudonyms securely. To do so, Lehmann (Lehmann, 2019) proposed the notion of unlinkable pseudonymisation (ScrambleDB) as a method of providing pseudonymisation-as-a-service in distributed settings, where multiple players contribute data to a shared pool. Generated pseudonyms protect the data from the attribute values and only combined through controlled as well as non-transitive JOIN operations to form data subsets on a per-request basis. This approach provides a secure method for pseudonymising data, but does not provide a mechanism for obfuscating the data. In settings involving highly sensitive data, this can become an issue since there is no way of preventing re-identifications due to inferences drawn from attribute values.

Obfuscating data can be achieved by techniques such as anonymisation. One of the first works in this direction was proposed by Sweeney (Sweeney, 2000; Sweeney, 2002a; Sweeney, 2002b). In order to enable anonymisation, a variety of data transformation methods such as *generalisation* (Sweeney, 2002a; Samarati and Sweeney, 1998; Ciglic et al., 2014), *suppression* (Samarati, 2001), and *perturbation* (Rubinstein and Hartzog, 2015), have been studied as a mean of transforming data for privacy, with low information loss. Sweeney's  $k$ -anonymisation approach, for instance, is basically a method of transforming data for privacy. In  $k$ -anonymity, a dataset is transformed to ensure that records are grouped into equivalence classes of size  $k$  such that every record in the given equivalence class is similar (indistinguishable) from the other  $k - 1$  records. Privacy in this case, rests on the principle of record similarity. Subsequent works such as  $l$ -diversity (Machanavajjhala et al., 2007; Meyerson and Williams, 2004; Bayardo and Agrawal, 2005),  $t$ -closeness (Li et al., 2007b; Fredj et al., 2015), dealing with minimality attacks (Wong et al., 2006; Wong et al., 2007a; Wong et al., 2007b; Wong et al., 2009; Wong et al., 2011), and differential privacy (Dwork et al., 2009; Dwork et al., 2006; Dwork,

2008; Dwork, 2011; Islam and Brankovic, 2011; McSherry and Talwar, 2007; Koufogiannis et al., 2015; Kifer and Machanavajjhala, 2011) have been studied quite extensively in a bid to answer the question of preventing disclosures of sensitive personal data in anonymised datasets (Sakpere and Kayem, 2014; Ghosh et al., 2012).

Data transformation algorithms offer the advantage of modifying attributes values in ways that make it hard to discover the original value. This is useful when the attribute-values are unique or present some property (e.g. distribution similarity, etc.) that might make it easier to draw inferences that result in discovering the original value and eventually the record (tuple).

However, as Aggrawal et al. (Aggarwal, 2005) have shown, applying generalisation and suppression to high dimensional data results in high information loss, thereby rendering the data useless for data analytics. This is made worse when the data must be formed from multiple distributed high-dimensional datasets.

High information loss can be addressed, to some degree, by perturbation. Perturbation modifies the original value to the closest similar findable value. This includes using an aggregated value or a close-by (approximate) value that is employed in a way that requires modifying only one value (or a very few values) instead of multiple ones to build clusters (equivalence classes). One drawback of perturbation is efficiently finding such values since finding such a value can take longer due to the effect of having to recheck the newly created value(s) iteratively. This affects performance negatively and so further work has sought to find alternatives that optimise the search time (Fienberg and Jin, 2012; Vaidya and Clifton, 2003; Vaidya et al., 2008).

Podlesny et al. (Podlesny et al., 2018; Podlesny et al., 2019c; Podlesny et al., 2019a; Podlesny et al., 2019b) proposed attribute compartmentation as a method of transforming high-dimensional datasets to offer privacy but at the same time minimise information loss. Attribute compartmentation ensures that the data is obfuscated, but assumes a trusted environment and so, is not designed to handle compositions of data from multiple distributed belonging to parties that co-exist in an untrusted environment. Furthermore, as is the case with all anonymisation approaches, a large proportion of information loss is incurred due to the necessity of having to eliminate personally identifying information. In datasets such as genome data which contain several unique identifiers having to eliminate these values reduces (negatively impacts on) the usefulness of the data for analytics

operations.

Therefore, having a method of retaining such attribute values and at the same time being able to prevent linkages that can be exploited for inference and re-identification, can make an important contribution to enabling privacy-preserving data sharing in untrusted environments. In the following section, we describe our approach to address this problem.

### 3 ENABLING OBFUSCATION AND SANITISATION

Addressing the issue of re-identification can be handled by obfuscation (Lehmann, 2019), in which case some form of transformation (e.g. generalisation, and/or suppression) is applied to the subset of data that is to be shared. In this regard, attribute compartmentation (Podlesny et al., 2019a) can be used to complement ScrambleDB. Attribute compartmentation provides data anonymity by identifying all personal identifiers and eliminating these from the dataset. The remainder of the dataset is then analysed to identify and eliminate or transform quasi-identifiers to prevent re-identification. Attribute compartmentation however, results in high information loss for datasets with high-level sensitive information (such datasets contain a high level of personal information like genomic mutations). Since high information loss impacts query result accuracy, having a solution that minimises information loss is beneficial.

Combining the ScrambleDB and Compartmentation, can serve as a mechanism for addressing the issue of preserving data utility without negatively impacting on data privacy. In this case, we need a method of ensuring that the different components of the data can be combined securely and in a way that ensures unlinkability. This is because while compartmentation eliminates the risk of re-identification attacks by breaking inter-attribute value functional dependencies, combining compartmented data with pseudonymous data JOIN operations could become transitive. So, in certain cases, the joined data could enable or result in re-identification. Furthermore, the data needs to be joined to preserve the utility of the data by minimising information loss.

#### 3.1 Preliminaries

We now describe the privacy setting and database structure to serve as building blocks for ScrambleDB and Compartmentation.

**Privacy Setting.** The privacy scenario in which ScrambleDB and Compartmentation operate is one in

which a central service exists to either anonymise or pseudonymise data. Multiple distributed data owners provide data and it is the job of the central service to render a pseudonymised or anonymised version of the data. At this stage, we assume that all the data owners trust the central service. The data owners are assumed not to trust each other.

**Database Specification.** In addition, we assume that the data is structured in the using a relational model, where  $T^{m \times n}$  denotes a table in the database that consists of  $n$  rows (records /tuples), and  $m$  columns (attributes). Each record in the table is uniquely identifiable via a primary key ( $uid$ ) and a table is uniquely identifiable by a table identifier ( $tid$ ). An attribute value falls in the cell at the intersection between an attribute (column) and the associated record. For instance,  $T^{m \times n}[i, j]$  contains a value  $val_{i,j}$  which denotes the value attribute  $attr_j$  holds for record  $rec_i$ . We note that  $1 \leq i \leq n$  and  $1 \leq j \leq m$ .

#### 3.2 Shared Data Eco-System

The architecture of our privacy-preserving shared data ecosystem is captured in Figure 1. As shown in Figure 1, our architecture is comprised of a set of  $N$  distributed data owners  $D = D_1, \dots, D_N$ . Each data owner owns and/or controls  $S = S_1, \dots, S_M$  a set of data sources. Hypothetically, both  $N$  and  $M$  are infinitely large; however, we will assume that both variables are bounded for practical reasons. So,  $N \geq 1$  and  $M \geq 1$ . In addition, a data owner can be both an owner and a processor  $P$ . Data processor, is an entity that requests a dataset, typically a concatenation of subsets of data emanating from various disjoint data sources. As is the case with  $D$  and  $S$ ,  $P \geq 1$  and we assume that  $|P|$  is finite and bounded. We make the following assumptions with respect to the interactions between  $D, S$ , and  $P$ :

- **Assumption #1.** The data sources  $S$  are independent. That is a data owner,  $N(x)$  who owns or controls dataset,  $S(x)$  has no access to  $S(x')$  that owned by  $N(x')$ .
- **Assumption #2.** A data processor,  $P$ , can be either a data owner  $N$  (i.e.  $P \equiv N$ ) or can be an external (i.e. outside of the data management / architectural ecosystem) requester ( $P_{ext}$ ). So  $P(D)$  accesses a joined dataset  $D$  composed from subsets of data such that:

$$D \leftarrow S(1), \dots, S(i), \dots, S(n)$$

where  $i$  denotes the  $i$ th data owner that contributes to the formation of  $D$  and  $n$  is the total number of data owners involved.

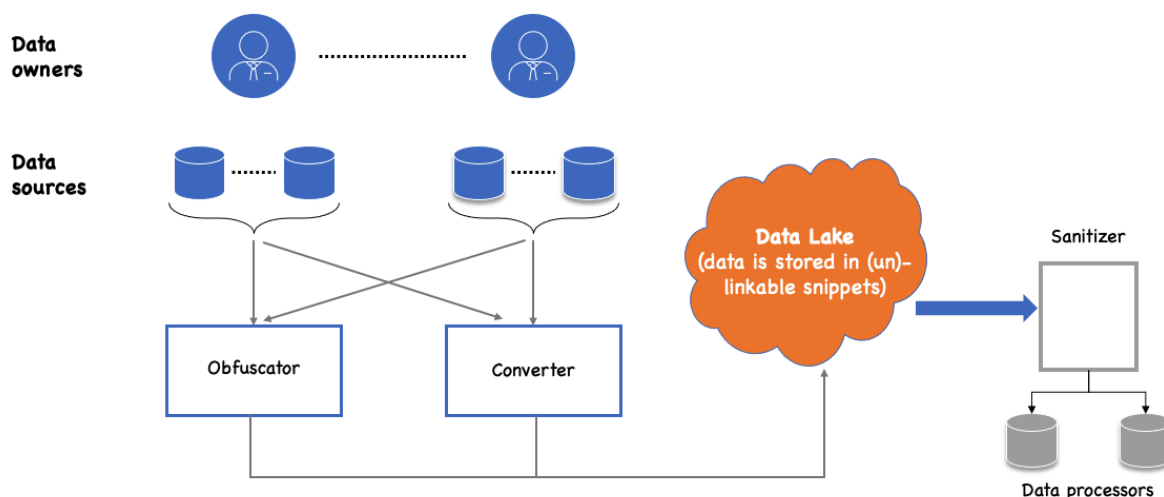


Figure 1: Data Sharing Architecture: Overview.

- **Assumption #3.** Data processors are not allowed to share datasets. That is, a collusion prevention strategy must be implemented to prevent this.

### 3.3 Privacy and Trust Setting

**Trust Set-up.** In the trust setup for our data sharing ecosystem, we consider that the data owners do not trust each other, but trust the central authority (CA). Furthermore, the central authority and data processors are honest-but-curious in that they have a vested interest in ensuring that the ecosystem works well, but would like to gain some advantage by learning or inferring hidden information. In the next section therefore, we formalise the security and privacy properties required to ensure data consistency and non-transitivity of joined datasets against both the CA and  $P$ . We now consider the structure of the data, in  $S$  as this is important is describing the method in which  $L$  is formed.

### 3.4 Data Source Structure

We employ a relational database model in which a database  $S$  is comprised of a set of tables represented by  $T^{m \times n}$  where  $n$  denotes the number of rows (records /tuples), and  $m$  columns (attributes). The set of attributes  $A \in S$  is such that  $A = a_1, \dots, a_i, \dots, a_n$  where  $a_i$  is the  $i$ th attribute. Likewise the set of tuples  $R \in S$  is such that  $R = r_1, \dots, r_j, \dots, r_m$  where  $r_j$  is the  $j$ th tuple. Typically,  $n \geq m$ , however we note that in high-dimensional datasets  $m$  can be quite high ( $m \geq 100$ ).

- **Assumption #4.** All data owners have the same data structure for the datasets they own. Likewise, all the datasets in the data-sharing ecosys-

tem, have the same relational structure or a means of obtaining a relational version from the data available.

The data lake can be composed and stored in anticipation of data processor requests. The management of join requests is handled by a central authority (CA) that is comprised of three components namely the *Obfuscator*, *Converter*, and *Sanitizer*.

### 3.5 Pseudonymisation and Compartmentation-as-a-Service

Essentially we have a data lake  $L$  that is created by contributions of data from distributed sources. In a manner similar to that in ScrambleDB (Lehmann, 2019), the *Obfuscator* transforms a dataset for pseudonymity, by binding each attribute value to a pseudonym. That is for every table  $T^{m \times n}$  in the dataset, where  $m$  is the number of attributes and  $n$  the number of records (rows),  $m$  pseudonymous tables are created, each containing  $n$  pseudonymous values. The pseudonym is generated securely to prevent linkability.

The *Obfuscator* plays the role of transforming the dataset to prevent re-identifications due to inferences drawn from functional dependencies between attribute values. As a further step, the *Obfuscator* analyses the dataset to discover all quasi-identifiers (functional dependencies) that are likely to result in inferences.

For flexibility, the *Obfuscator* employs *Chameleon pseudonyms* to handle the data when it is collected. The *Chameleon pseudonyms* are useful in that they allow for the data collected and the associated attribute values to be stored in a manner

that is *fully unlinkable*. In the data lake, the data is stored as a set of unlinkable attribute tables. This guarantees that the data is secure at rest and offers the flexibility of allowing the *Converter* to bind unlinkable secure-pseudonyms to data at runtime when a secure and private subset of data is required.

As a further step, the *Obfuscator* analyses the dataset to discover all quasi-identifiers (functional dependencies) that are likely to result in re-identification. To do this, the *Obfuscator* classifies attributes as either 1st-class or 2nd-class identifiers. Attributes classified as 1st-class identifiers explicitly identify an individual, while 2nd-class identifiers identify an individual if a “correct” attribution combination is found. The *Obfuscator* evaluates different combinations of attributes based on the risk of re-identification when placed together in the same subset of data. We use a uniqueness constraint to determine the entropy level required to classify an attribute or attribute value as prone to provoking re-identification. Additionally, at this stage, data transformation operations (such as generalisation and suppression) can be applied to guarantee privacy further.

Anonymising data via attribute compartmentation requires that we proceed in two steps. The first step involves identifying attributes that qualify as 1st class-identifiers (explicit identifiers). Since these attributes are eliminated from the dataset to prevent personal information disclosure when the data is shared, this can result in high information loss in datasets that contain a high proportion of sensitive data. We address this issue by using unlinkable pseudonymisation to replace the attribute values with an equivalent pseudonym. In this way, the levels of information loss due to eliminations of 1st class-identifiers is reduced significantly.

The *Converter* is a central service that handles the output from the *Obfuscator*. Its job is to derive and convert pseudonyms, and create a *joined* version of attributes based on the request or usage scenario of the data.

The *Converter* performs these transforms blindly in that it stores no knowledge of the data sets. To create a subset of data, the *Converter* transform the *Chameleon Pseudonyms*, associated with the attributes that are to be placed in the requested data subset, to ensure that pseudonyms associated with the same record are mapped to the same value.

We suppose that we have several distributed data sources  $S$ , that contribute data to a shared pool (data lake)  $L$ . A set of users (data processors)  $P$  periodically request access to the data  $L$ . All such accesses must be handled in a privacy-preserving manner, so that the snapshot of  $L$  that is shared with  $P$  is secure and un-

linkable. The snapshot of  $L$  is created by joining a subset of the table’s output from the pseudonymisation and anonymisation operations. Snapshots of data are created when the data lake receives a request for a given dataset. For simplicity, we assume that the snapshot is created in anticipation of a set of queries or a series of data analytics operations rather than on-demand.

In order to create a data snapshot (*JOIN* the tables), the data lake interacts with two entities namely: the *Obfuscator* and the *converter*. The *Obfuscator* uses attribute compartmentation to support or validate permissible join operations (non-transitive joins) and handles the data transformations (e.g. generalisation and suppression) needed to create privacy-preserving data. The *converter* transforms the data output from the *Obfuscator* by mapping attribute values to unlinkable pseudonyms but in a consistent manner. That is, pseudonyms belonging to the same record are mapped to the same unlinkable pseudonym.

If a data processor  $P$  requires a snapshot of the data containing certain attributes, or if a generic data set containing certain attributes need to be created, a *joined* copy of the data (snapshot) is created and shared following coordination between the *Obfuscator* and the *Converter*. The *Converter* takes care of blindly transforming the unlinkable pseudonyms into a consistent form and preserves privacy by preventing linkability. The *JOIN* operation is strictly non-transitive in that every transformation cannot be correlated with data received in previous or other snapshots requested.

The *Sanitiser* serves as an internal validator or verifier to test the data that is to be shared with the data processor to ensure that it is free of elements that can be exploited for inferences.

To support the *Sanitiser*, we employ the compartmentation procedure that essentially checks each dataset to be shared to ensure that no quasi-identifiers remain therein. If quasi-identifiers are discovered in the dataset, the *Sanitiser* transforms the data using standard procedures such as generalisation and suppression (Sweeney, 2002a). Since data transformation loss procedures, by necessity, imply information loss, the *Sanitiser* employs an optimisation model to evaluate the cost-benefit ratio of information loss to privacy disclosure risk and usability.

In so doing, we have an internal verification model to ensure that *joined* attributes in the dataset to be shared do not raise re-identification vulnerabilities due to quasi-identifiers in the dataset; and we control information loss on the discovered quasi-identifiers to ensure a high-degree of usability of the shared dataset.

### 3.6 Formation of the Data Lake

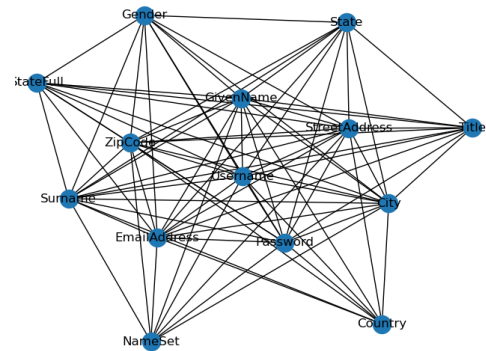
For simplicity, we will consider for now that the joined data is formed according to Assumption #2. The data processors wishing to access a subset of the data shared by  $N$  interact with the data lake  $L$ . For instance, when  $P$  contacts  $L$  to request a subset of data containing attributes  $a_1, \dots, a_l$ ,  $L$  produces a joined version of data from a set of tables  $T_1^{1 \times m}, \dots, T_l^{1 \times m}$ .

To ensure that the joined dataset is free of inference, vulnerable attribute values the *Sanitiser* supports  $L$ , by analysing the attribute combinations. For each combination of attributes, the *Sanitiser* maps the attributes into an attribute-combination graph  $G : (V, E)$  where  $V$  denotes the attributes (nodes) and  $E$  denotes the attribute-combination (edge). Each edge is weighted by the cardinality (or entropy) associated with the attribute-value combination. For example,  $A \longleftrightarrow B$  represents the attribute combination  $AB$ , and  $A \longleftrightarrow (10)B$  indicates the degree of uniqueness (entropy) in attribute-combination values is 10.

To obtain ensure that the combined (joined) dataset is privacy-preserving, standard transformation measures such as *local suppression*, *generalisation* and *perturbation* are first applied to reduce the cardinality (uniqueness) of the attribute-value combinations (Sweeney, 2002a). This ensures that we reduce the degree of inference due to the attribute-values associated with the attribute-combination. This is similar to how standard anonymisation approaches work to generate privacy, preserving datasets.

As a further measure, and to break the functional dependencies that enable inferences, we compute maximal cliques of  $G$  to determine which attributes can be placed together within a compartment without posing a risk of inference. These cliques guide the formation of feature sets (compartments) that can be published together safely (joined) without the risk of inference. Finally, as a post-processing measure, all feature sets are tested to ensure no inference causing attribute-combination values to existing therein.

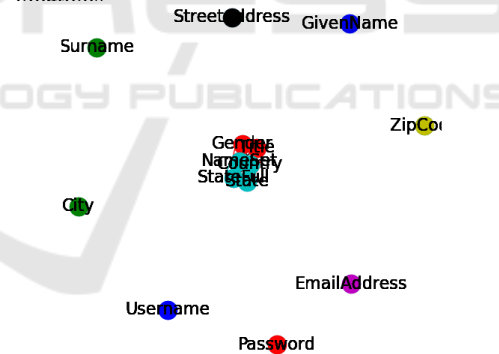
Figure 2 provides a real-world example of how compartmentation is applied to datasets. Figure 2(a.) has a tightly connected graph representing the attribute combinations that can be exploited for disclosure. Based on this graph, we can compute several cliques, as shown in Figure 2(b.). These cliques can then be combined in different ways to form feature sets of attribute combinations that can be published together safely without the risk of disclosure (see Figure 2(c.)). In the following section, we present results from our empirical model.



(a) Attribute tuples forming QIDs to be compartmentalized



(b) Inversed tuple edges as allowed attribute combinations



(c) Maximal cliques within inverse graphs

Figure 2: Attribute Compartmentation as max cliques in real-world.

## 4 EVALUATION

To validate our hypothesis, we have conducted a series of experiments along with key metrics. To assess the practicability, we delineate the runtime as an indicator of the time complexity, degree of information-loss and risk of privacy exposure through remaining quasi-identifiers.

## 4.1 Experimental Setup

For transparency and repeatability, we briefly outline the used experimental hardware and the composition of the dataset used. **Dataset.** A larger dataset based on semi-synthetic PII data<sup>1</sup> was compiled for the experiments. The information from these profiles has been combined with information from the genome (Clarke et al., 2012; Project, ). As a result, the dataset resembles a traditional multi-party setup in the Digital Health market, with its existing difficulties in data privacy-preserving publishing and sharing (Schadt and Chilukuri, 2015; Aue et al., 2015). **Hardware.** Our analysis uses a GPU-accelerated high-performance computing cluster with 160 CPU cores (E5-2698 v4), 760GB RAM, and 10 dedicated Tesla V100 units. Each GPU instance has 5120 CUDA cores and 1120 TFlops of combined Tensor performance. For GPU-related experiments, the execution area will be limited to ten CPU core and a single dedicated Tesla V100 GPU<sup>2</sup>. The compute cluster's runtime area is limited to ten dedicated cores for CPU-related experiments.

We briefly implemented a proof-of-concept model of a scrambleDB setup with the support of attribute compartmentation. Further, experiments have been conducted to evaluate the degree of information loss on per row and column and per-table basis as well as the risk of privacy exposure through the remaining quasi-identifier tuples. Finally, we also look at practical hashrates that can be achieved with state-of-the-art hardware and runtime constraints for large-scale datasets.

## 4.2 Practical Hashrates for Real-time Obfuscation

Our initial concern originated around the possibilities of the generated large amount of collision-free hashes in a reasonable time window. Especially in the context of stream data a proper throughput must be ensured. To assess this, we have run several benchmarks on state-of-the-art hardware, where Figure 3 delineates the results as hash-rates per second. Depending on the hashing method, these GPU-accelerated frameworks enable up to 39.9 GH/s, so 39 billion NTLM hashes per second. Figure 4a depicts a breakdown of the runtime composition, comparing a tuple length of 2 attributes against 80 attributes. With a growing number of attributes, QID discovery's effort significantly increases while the spin-up time re-

<sup>1</sup><https://www.fakenamegenerator.com/>

<sup>2</sup><https://images.nvidia.com/content/technologies/volta/pdf/volta-v100-datasheet-update-us-1165301-r5.pdf>

mains and the time required for hashing only fractional increases. This confirms the feasibility of creating collision-free hashes for large-scale datasets.

## 4.3 Degree of Information-loss

To measure the degree of the information-loss suffered from the data anonymisation processing, we start from the untreated dataset's diversity where each unique attribute value receives points and penalises alternations in the sanitized dataset. For the complete suppression, none points remain. For falsified numeric values, negative points are given and for alternated attribute values, we subtract points accordingly to the generalisation steps. Figure 6 delineates the evolution over growing dataset dimensions by columns and rows to indicate the trend of each anonymisation technique. Ultimately, the more describing attributes are available, the more information is present, yet more attribute values are being sanitized. Figure 5 depicts the information-loss over the same evolution of increasing describing attributes broken down by anonymisation approach. Here, it becomes transparent that the novel technique of combined scrambleDb and compartmentation behaves similarly to compartmentation while generalisation achieves the highest scores with the remark on significant – almost impractical – runtime costs.

## 4.4 Risk of Privacy Exposure

To evaluate the remaining risk of private data exposure, we re-run the quasi-identifier discovery to validate that given each anonymisation technique no unique attribute value tuples remain that serve auxiliary data attackers to draw conclusions and derive private information by linking external datasets. Following  $k$ -anonymity constraints, for all approach no quasi-identifiers have been found. So, in effect, the *Sanitizer* module is able to offer strong guarantees of data privacy in the combined data that is released or shared publicly.

## 4.5 Runtime

The execution time is an important metric to evaluate the practicality of each data anonymisation technique. As we know, generalisation is a really strong method that promises outstanding anonymisation results on numeric data for a known hierarchy, yet for high-dimensional data non-heuristic generalisation quickly becomes impractical due to its NP-hard runtime. For this purpose, Figure 7 compares the established syntactic data anonymisation techniques



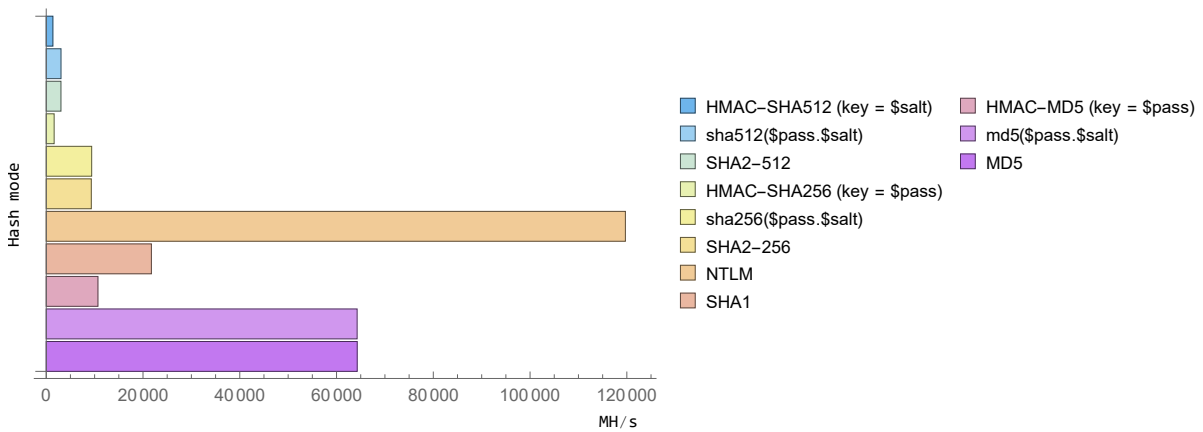
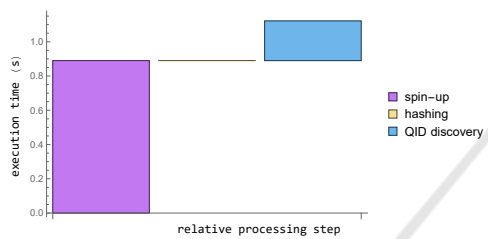
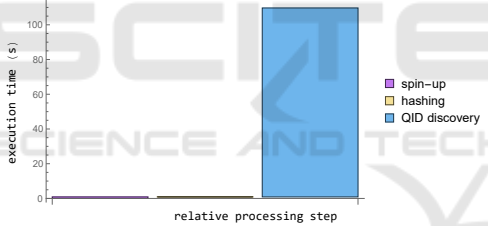


Figure 3: GPU-accelerated hashrates for obfuscation.



(a) Breakdown of execution time in seconds for L=2.



(b) Breakdown of execution time in seconds for L=20.

Figure 4: Runtime breakdown for scrambleDB combined with compartmentation.

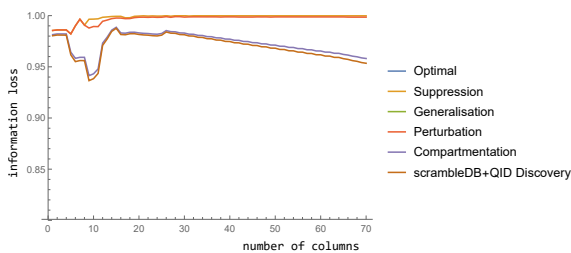


Figure 5: Information-loss comparison.

against the novel scrambleDB combined with compartmentation approach. All methods suffer significant runtime growth, yet given the introduced nature of scrambleDB a lot of workloads can be shifted from a-priori to adhoc at runtime combined with the soundness of  $k$ -anonymity requirements. They become ap-

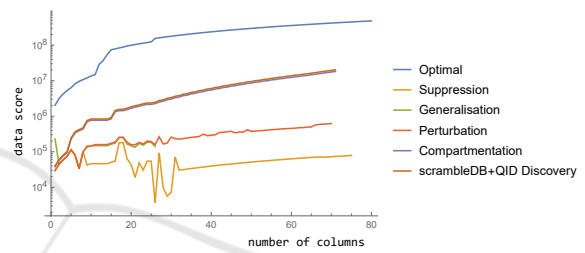


Figure 6: Data score comparison.

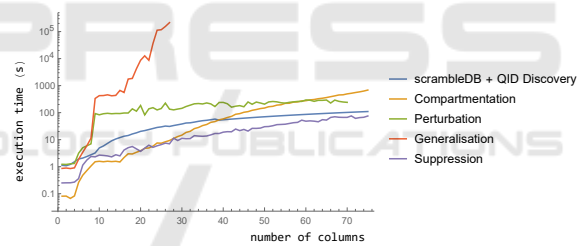


Figure 7: Run time comparison of selected synthetic data anonymisation techniques.

parent particularly for large attribute amounts where the curve flattens quicker than alternative syntactic data anonymisation methods.

## 4.6 Discussion

ScrambleDB combined with attribute compartmentation offers interesting insights for the security perspective, as it upgrades the original Pseudonymisation-as-a-Service to ensure the absence of any quasi-identifier in its results set. Establish evaluation metrics evince no significant improvements over existing syntactic data anonymisation techniques regarding runtime or information-loss for traditional settings. Yet, the novel approach does enable a new angle, especially for distributed data environments. These distributed environments are often present in medical settings acting as data ecosystems where

multiple players have subsets of data repositories and want to share information privacy-conform with private data exposure risk.

## 5 CONCLUSIONS

This paper has shown that by combining cryptographically secure pseudonymisation and compartmentation, we can generate privacy-preserving high-dimensional datasets. Furthermore, this can be done safely (in that the participating parties' privacy service level agreements can be adhered to) in an untrusted distributed environment. In such environments, each data owner holds a dataset and the composed data must be formed by joining subsets of data belonging to parties who co-exist in an untrusted environment. The pseudonymisation and compartmentation are outsourced to a central but fully oblivious entity that can blindly compose datasets based on distributed sources. Controlled non-transitive join operations are used to ensure that the published datasets do not violate the contributing parties' privacy properties. As a further step, the service provider will employ obfuscation and sanitisation to identify and break functional dependencies between attribute values that hold the risk of inferential disclosures. Results from our empirical model indicated that the overhead due to cryptographic pseudonymisation is negligible and can be deployed in large datasets in a scalable manner. Furthermore, we succeed in minimising information loss, even in large datasets, without impacting privacy negatively.

As future work, it would be interesting to consider how joined datasets can be formed even in the presence of active adversaries amongst the data owners. It would also make sense to consider cases in which data owners wish to contribute data to the data lake and control how the data is merged or shared. For instance, we could consider cases in which the data owners are able to upload or influence the join policies and, by extension, what impact this will have on the privacy of the joined data, particularly in the presence of active adversaries.

## REFERENCES

- Aggarwal, C. C. (2005). On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st International Conference on Very Large Data Bases, VLDB '05*, pages 901–909. VLDB Endowment.
- Aue, G., Biesdorf, S., and Henke, N. (2015). ehealth 2.0: How health systems can gain a leadership role in digital health.
- Bayardo, R. J. and Agrawal, R. (2005). Data privacy through optimal k-anonymization. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 217–228. IEEE.
- Camenisch, J. and Lehmann, A. (2015). (un)linkable pseudonyms for governmental databases. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15*, page 1467–1479, New York, NY, USA. Association for Computing Machinery.
- Ciglic, M., Eder, J., and Koncilia, C. (2014). k-anonymity of microdata with null values. In Decker, H., Lhotská, L., Link, S., Spies, M., and Wagner, R. R., editors, *Database and Expert Systems Applications*, pages 328–342, Cham. Springer International Publishing.
- Clarke, L., Zheng-Bradley, X., Smith, R., Kulesha, E., Xiao, C., Toneva, I., Vaughan, B., Preuss, D., Leinonen, R., Shumway, M., et al. (2012). The 1000 genomes project: data management and community access. *Nature methods*, 9(5):459–462.
- Dwork, C. (2008). Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer.
- Dwork, C. (2011). Differential privacy. In *Encyclopedia of Cryptography and Security*, pages 338–340. Springer.
- Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*. Springer.
- Dwork, C., Naor, M., Reingold, O., Rothblum, G. N., and Vadhan, S. (2009). On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, STOC '09*, pages 381–390, New York, NY, USA. ACM.
- Fienberg, S. E. and Jin, J. (2012). Privacy-preserving data sharing in high dimensional regression and classification settings. *Journal of Privacy and Confidentiality*.
- Fredj, F. B., Lammari, N., and Comyn-Wattiau, I. (2015). Abstracting anonymization techniques: A prerequisite for selecting a generalization algorithm. *Procedia Computer Science*, 60:206–215.
- Ghosh, A., Roughgarden, T., and Sundararajan, M. (2012). Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693.
- Islam, M. Z. and Brankovic, L. (2011). Privacy preserving data mining: A noise addition framework using a novel clustering technique. *Knowledge-Based Systems*, 24(8).
- Kifer, D. and Machanavajjhala, A. (2011). No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, SIGMOD '11*, pages 193–204, New York, NY, USA. ACM.
- Koufogiannis, F., Han, S., and Pappas, G. J. (2015). Optimality of the laplace mechanism in differential privacy. *arXiv preprint arXiv:1504.00065*.
- Lehmann, A. (2019). Scrambledb: Oblivious (chameleon) pseudonymization-as-a-service. *PoPETS*, 2019(3):289–309.

- Li, N., Li, T., and Venkatasubramanian, S. (2007a). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115.
- Li, N., Li, T., and Venkatasubramanian, S. (2007b). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd ICDE*, pages 106–115.
- Machanavajjhala, A., Kifer, D., Gehrke, J., and Venkatasubramanian, M. (2007). l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3.
- McCallister, E., Grance, T., and Scarfone, K. (2010). Guide to protecting the confidentiality of personally identifiable information (pii). *NIST Special Publication 800-122, National Institute of Standards and Technology*, 2010.
- McSherry, F. and Talwar, K. (2007). Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 94–103. IEEE.
- Meyerson, A. and Williams, R. (2004). On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228. ACM.
- Party, T. A. . D. P. W. (2014). Opinion 05/2014 on anonymisation techniques.
- Podlesny, N. J., Kayem, A. V., and Meinel, C. (2019a). Attribute compartmentation and greedy ucc discovery for high-dimensional data anonymization. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, pages 109–119. ACM.
- Podlesny, N. J., Kayem, A. V., and Meinel, C. (2019b). Towards identifying de-anonymisation risks in distributed health data silos. In *International Conference on Database and Expert Systems Applications*. Springer.
- Podlesny, N. J., Kayem, A. V., von Schorlemer, S., and Uflacker, M. (2018). Minimising information loss on anonymised high dimensional data with greedy in-memory processing. In *International Conference on Database and Expert Systems Applications*, pages 85–100. Springer.
- Podlesny, N. J., Kayem, A. V. D. M., and Meinel, C. (2019c). Identifying data exposure across distributed high-dimensional health data silos through bayesian networks optimised by multigrid and manifold. In *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress, DASC/PiCom/CBDCCom/CyberSciTech 2019, Fukuoka, Japan, August 5-8, 2019*, pages 556–563.
- Project, . G. 1000 genomes project.
- Rubinstein, I. and Hartzog, W. (2015). Anonymization and risk.
- Sakpere, A. and Kayem, A. (2014). A state-of-the-art review of data stream anonymization schemes. In *Information Security in Diverse Computing Environments*, pages 24–50. IGI Global.
- Samarati, P. (2001). Protecting respondents identities in microdata release. *IEEE transactions on Knowledge and Data Engineering*, 13(6):1010–1027.
- Samarati, P. and Sweeney, L. (1998). Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, Technical report, SRI International.
- Schadt, E. and Chilukuri, S. (2015). The role of big data in medicine.
- Sweeney, L. (2000). Simple demographics often identify people uniquely. *Health (San Francisco)*, 671:1–34.
- Sweeney, L. (2002a). Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):571–588.
- Sweeney, L. (2002b). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570.
- Vaidya, J. and Clifton, C. (2003). Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215. ACM.
- Vaidya, J., Kantarcıoğlu, M., and Clifton, C. (2008). Privacy-preserving naive bayes classification. *The VLDB Journal?The International Journal on Very Large Data Bases*, 17(4):879–898.
- Wong, R. C.-W., Fu, A. W.-C., Wang, K., and Pei, J. (2007a). Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases, VLDB '07*, page 543?554. VLDB Endowment.
- Wong, R. C.-W., Fu, A. W.-C., Wang, K., and Pei, J. (2007b). Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases, VLDB '07*, page 543?554. VLDB Endowment.
- Wong, R. C.-W., Fu, A. W.-C., Wang, K., and Pei, J. (2009). Anonymization-based attacks in privacy-preserving data publishing. *ACM Trans. Database Syst.*, 34(2).
- Wong, R. C.-W., Fu, A. W.-C., Wang, K., Yu, P. S., and Pei, J. (2011). Can the utility of anonymized data be used for privacy breaches? *ACM Trans. Knowl. Discov. Data*, 5(3).
- Wong, R. C.-W., Li, J., Fu, A. W.-C., and Wang, K. (2006). (? , k)-anonymity: An enhanced k-anonymity model for privacy preserving data publishing. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06*, page 754?759, New York, NY, USA. Association for Computing Machinery.