Journal of Privacy and Confidentiality
Vol. 12 (1) 2022

Submitted Nov 2020
Published Jul 2022

A TAXONOMY OF ATTACKS ON PRIVACY-PRESERVING RECORD LINKAGE

ANUSHKA VIDANAGE, THILINA RANBADUGE, PETER CHRISTEN, AND RAINER SCHNELL

The Australian National University, Canberra, Australia
e-mail address: anushka.vidanage@anu.edu.au

Data61, CSIRO, Canberra, Australia
e-mail address: thilina.ranbaduge@csiro.au

The Australian National University, Canberra, Australia
e-mail address: peter.christen@anu.edu.au

University Duisburg-Essen, Duisburg, Germany
e-mail address: rainer.schnell@uni-due.de

Abstract. Record linkage is the process of identifying records that correspond to the same real-world entities across different databases. Due to the absence of unique entity identifiers, record linkage is often based on quasi-identifying values of entities (individuals) such as their names and addresses. However, regulatory ethical and legal obligations can limit the use of such personal information in the linkage process in order to protect the privacy and confidentiality of entities. Privacy-preserving record linkage (PPRL) aims to develop techniques that enable the linkage of records without revealing any sensitive or confidential information about the entities that are represented by these records. Over the past two decades, various PPRL techniques have been proposed to securely link records between different databases by encrypting and/or encoding sensitive values. However, some PPRL techniques, such as popular Bloom filter encoding, have shown to be susceptible to privacy attacks. These attacks exploit the weaknesses of PPRL techniques by trying to reidentify encrypted and/or encoded sensitive values. In this paper, we propose a taxonomy for analysing such attacks on PPRL where we categorise attacks across twelve dimensions, including different types of adversaries, different attack types, assumed knowledge of the adversary, the vulnerabilities of encoded and/or encrypted values exploited by an attack, and assessing the success of attacks. Our taxonomy can be used by data custodians to analyse the privacy risks associated with different PPRL techniques in terms of existing as well as potential future attacks on PPRL.

Key words and phrases: Reidentification, privacy attacks, encoding, encryption, vulnerability analysis.

1. Introduction

Data privacy has become a growing concern in application domains including information sharing, population informatics, financial systems, and cyber-security, as increasingly large amounts of personal or otherwise sensitive data are being collected from numerous organisations everyday (Vaidya et al., 2006). The linking of records across databases, known as record linkage (Herzog et al., 2007), is no exception to such data privacy concerns (Christen et al., 2020). Because often no common record identifiers are available across the databases to be linked, record linkage is often based on personal identifying information of people (known as quasi-identifiers) such as their names, addresses, and dates of birth (Christen, 2012). However, the use of such personal information can raise privacy and confidentiality concerns, and organisations are often not willing or allowed to share such sensitive information with other organisations (Vatsalan et al., 2013). Laws and policies, such as the European Union General Data Protection Regulation (GDPR)[1], are in place in many countries to regulate how and when sensitive personal data can be used (Christen et al., 2020). The demand for privacy protection of sensitive data has led to research into *privacy-preserving record linkage* (PPRL) (Hall and Fienberg, 2010). PPRL focuses on developing techniques that enable the linking of records across databases while preserving the privacy of the entities that are represented by these records (Gkoulalas-Divanis et al., 2021).

Over the last two decades, a variety of PPRL methods have been proposed (Vatsalan et al., 2013). These methods can be divided into two main categories: (a) secure multi-party computation (SMC) and (b) perturbation based methods. SMC based methods enable several parties to be involved in a computation with their sensitive input values, where at the end of the computation no party learns anything about any other party's sensitive input values but all parties learn the final result of the computation (Lindell and Pinkas, 2009). While provably secure, SMC based methods generally incur high computation and communication costs. Perturbation based methods, on the other hand, transform and encode sensitive values such that no sensitive information of entities can be obtained from such encoded data (Vatsalan et al., 2013). Perturbation based methods generally have a trade-off between linkage quality, scalability to linking large databases, and privacy protection.

One perturbation based PPRL method that has gained popularity is Bloom filter (BF) encoding. This is due to its ability to efficiently calculate approximate similarities between records, and the ease of implementation (Schnell et al., 2009). BF encoding is now being used in several real-world PPRL applications (Boyd et al., 2015; Antoni and Schnell, 2017; Pita et al., 2018). However, recent research has shown that BF encoding based PPRL methods are vulnerable to privacy attacks (Kuzu et al., 2011; Niedermeyer et al., 2014; Christen et al., 2017; Mitchell et al., 2017; Vidanage et al., 2019). To overcome the privacy weaknesses of BF encoding, alternative perturbation based encoding techniques have recently been developed (Smith, 2017; Randall et al., 2019; Ranbaduge et al., 2020a). However, some of these techniques have also shown to be susceptible to certain privacy attacks (Culnane et al., 2017a; Vidanage et al., 2020b,a).

Different privacy attacks have been discussed not only in PPRL but also in the contexts of privacy-preserving data mining, statistical disclosure control, and secure data publishing (Domingo-Ferrer et al., 2015). These attacks aim to reidentify encoded and/or anonymised values in a sensitive database in order to disclose the identities of entities (individuals) represented by those values. Furthermore, real-world data breaches (Narayanan

[1]See: `https://gdpr-info.eu/`

and Shmatikov, 2008; Culnane et al., 2017b) have highlighted that any technique aimed at preserving the privacy of sensitive data needs to be carefully analysed before being used in real-world applications.

Given there are different attack methods that can potentially be applied on PPRL techniques (Kuzu et al., 2011; Kroll and Steinmetzer, 2015; Christen et al., 2017; Mitchell et al., 2017; Vidanage et al., 2019), it is important to explore the characteristics of such attacks in order to understand their suitability in real-world situations. In this paper, we propose a novel taxonomy of attacks on PPRL, where we categorise attacks using twelve dimensions. This taxonomy can be used to analyse different aspects of privacy attacks and will help data custodians to understand the different reasons behind potential privacy breaches, the level of accessibility to information that an adversary requires in order to be able to successfully conduct a privacy attack, and to determine the privacy guarantees of PPRL methods with respect to existing attacks.

The remainder of this paper is structured as follows. In Section 2 we describe the different types of parties involved in a PPRL protocol and give a brief overview of the PPRL process. In Section 3 we provide an overview of a general attack on PPRL and describe the main steps of such an attack. In Section 4 we then propose a taxonomy of attacks on PPRL based on twelve dimensions that can be used to categorise attacks. We divide these twelve dimensions into adversarial, technical, and practical aspects. In Section 5 we provide recommendations that should be considered when conducting real-world PPRL projects.

For the interested reader, we provide an overview of the different adversary models that can be considered in a PPRL protocol in Appendix A, extend our discussion on an adversary's assumed knowledge of the data to be linked in Appendix B, describe the five types of vulnerabilities that can exist in encoded and plaintext databases in Appendix C, and discuss existing attacks proposed for different PPRL techniques in Appendix D.

2. Overview of Privacy-Preserving Record Linkage

PPRL (Hall and Fienberg, 2010) addresses the problem of linking databases that contain sensitive information while preserving the privacy and confidentiality of the entities that are represented in these databases from parties both internal as well as external to the linkage process. Generally, the sensitive data that are being linked refer to people (Christen et al., 2020). For a PPRL technique to be used in real-world linkage applications, it needs to be able to link the true matching record pairs across different databases correctly and efficiently while ensuring the privacy of the entities that are represented by the sensitive values in the databases being linked.

We first discuss the types of parties involved in a PPRL protocol because it is important to understand the accessibility to data that the different parties in a linkage protocol have. We then provide a brief overview of the PPRL process of applying an encoding or encryption method to protect sensitive information, and conducting the linkage using the generated encodings or encryptions.

2.1. Roles of Parties.

In a PPRL protocol, different parties can participate in the linkage protocol. The roles of these parties can be categorised as follows.

- **Database owner (DO)**: A person or organisation who holds the authority over a database that is to be linked is named a database owner (Christen et al., 2020). DOs are also referred to as data owners or data custodians. Depending on the PPRL protocol used, DOs do undertake certain operations in the linkage process, such as pre-processing and encoding of their own databases. Healthcare service providers, business owners, and government departments are some examples of DOs.

- **Linkage unit (LU)**: A linkage unit is a party that participates in the linkage process to conduct the linkage using the encoded attribute values it receives from the DOs. A LU can either be one of the DOs themselves or a party external to the DOs (Vatsalan et al., 2013; Christen et al., 2020). As we discuss below, in general the LU does not have access to any of the sensitive data held by the DOs, and it also does not know any of the parameter settings and the secret key(s) (if any) used in the encoding process.
- **Data consumer (DC)**: A person or organisation who utilises the linked data at the end of the PPRL process to conduct data analysis is called a data consumer (Bizer et al., 2011; Christen et al., 2020). Depending on the requirements of (or agreements made prior to) a PPRL process, a DC will have access to the microdata from the sensitive database and a set of unique identifiers assigned to matched records. Any sensitive microdata need to be properly anonymised before they are being made available to a DC to prevent any form of reidentification (Domingo-Ferrer and Torra, 2003; Taylor et al., 2018). DCs can either be the DOs whose databases have been linked, or an external party such as a researcher.

Other parties that might also be involved in a linkage protocol can include a global authority (a party that plays the role of a central, public, and trusted regulatory agency that creates a global summary using local summaries), a facilitator (a party that helps the DOs to negotiate and plan their common objectives of a linkage process), and a data controller (a party that incorporates relevant laws and regulations for a linkage protocol and that provides permissions to the parties involved in the protocol) (Christen et al., 2020).

2.2. **General Encoding and Linkage Process.** The general PPRL process consists of five main steps (Vatsalan et al., 2013): data pre-processing and encoding, private blocking, private comparison, private classification, and evaluation. Data pre-processing includes cleaning, normalisation, and standardisation of data before the linkage. This step generally does not require any privacy technique since it can be conducted independently by each database owner (DO) (Christen et al., 2020). However, if the employed data pre-processing methods require the DOs to share metadata (such as attribute values and their frequencies), potential privacy concerns can occur due to the characteristics of such metadata, for example the uniqueness of values, that are being shared.

After the databases to be linked have been cleaned, the DOs encode their own databases using an agreed encoding or an encryption technique. In the process of encoding the sensitive values in their databases, the DOs first need to agree upon a set of quasi-identifying (QID) attributes common across all the databases to be used to link records, and an encoding function $enc()$ with a common set of encoding parameters $\mathbf{p}$, as well as a secret key s to be used for the encoding. For instance, the list of QIDs can be [*FirstName, LastName, StreetAddress, City, Zipcode*].

We represent a sensitive database as $\mathbf{D}^s = (\mathbf{Q}^s, \mathbf{M})$, where $\mathbf{Q}^s$ is the list of actual attribute values of records in the selected QIDs to be used, and $\mathbf{M}$ are the microdata in $\mathbf{D}^s$ (such as medical or financial details of individuals) that can be used in the analysis of the linked database. Each DO$_i$ then encodes the values in $\mathbf{Q}^s$ of its own sensitive database $\mathbf{D}^s_i$ using the encoding function $enc()$, where $1 \leq i \leq d$ and d is the number of databases to be linked. We denote the encoding of a database $\mathbf{D}_i^s$ using its list of QID values $\mathbf{Q}_i^s$ as:

$$\mathbf{E}_i = enc(\mathbf{Q}_i^s, \mathbf{p}, s), \tag{2.1}$$

where $\mathbf{p}$ is the set of encoding parameter values, s is the secret key used, and $\mathbf{E}_i$ is the resulting list of encoded QID values that have been generated.

We represent an encoded database as $\mathbf{D}^e = (\mathbf{E}, \mathbf{M})$ where $|\mathbf{E}| = |\mathbf{Q}^s|$. The microdata, $\mathbf{M}$, are not encoded in $\mathbf{D}^e$ because these are the values of interest to a data analyst (data consumer) after the linkage process. However, $\mathbf{M}$ can include certain QID values such as gender, year of birth, and zipcode, that might be useful in an analysis task. In such a scenario these QID values are often generalised to prevent reidentification before being published or used for an analysis (Taylor et al., 2018).

After the encoding process, the linkage of encoded QID values $\mathbf{E}_i$ of each database $\mathbf{D}_i^e$ from DO$_i$ is conducted. This process includes the steps private blocking, private comparison, and private classification (Vatsalan et al., 2013). In private blocking encoded records that likely correspond to matches are grouped into blocks. Private blocking aims to reduce the quadratic comparison space of all pair-wise comparisons (Ranbaduge, 2017). Private comparison focuses on calculating the similarities between encoded record pairs that are in the same block (Karakasidis and Verykios, 2011; Durham, 2012). Next, in private classification, the compared record pairs are classified into two classes, *matches* and *non-matches*, based on the calculated similarities between them (Christen, 2012). The class of *matches* are record pairs that are assumed to refer to the same real-world entity while the class of *non-matches* are pairs where their records are assumed not to correspond to the same entity. Finally, the linkage results (pairs of record identifiers from the class *matches*) are shared between the corresponding DOs to conduct further analysis. Alternatively, linkage results can also be shared with a data consumer (DC) as we discuss below.

Certain linkage protocols employ a third party, commonly known as the linkage unit (LU), to conduct the linkage of records. When such a LU is employed, the encoded QID values $\mathbf{E}_i$ of each DO$_i$ are sent to the LU to conduct the linkage (Gkoulalas-Divanis et al., 2021; Vatsalan et al., 2013). The LU conducts the private blocking, private comparison, and private classification steps, and returns the linkage results back to the DOs.

Alternatively, the DOs can collaboratively conduct the linkage process without the involvement of a LU by communicating directly with each other. Protocols without a LU are generally more secure compared to protocols that do involve a LU because, by eliminating the need for a LU, they prevent any type of collusion that might occur between the DOs and the LU, and also they do not require the disclose of data to an external party.

At the end of the PPRL process, the DOs will only learn which records in their own sensitive database $\mathbf{D}_i^s$ have been matched with records in the other sensitive databases, while the LU (if a third party is employed) will learn nothing about the sensitive values in any of the $\mathbf{D}_i^e$. The final step is evaluating the linkage process in terms of the quality of the linkage results, scalability of the linkage to large databases, and the privacy protection offered by a PPRL protocol (Vatsalan et al., 2014; Ranbaduge, 2017).

We now formally define PPRL (Vatsalan et al., 2013; Verykios and Christen, 2013) as:

Definition 2.1. Privacy-preserving record linkage (PPRL)
Let us assume d database owners DO$_1$, DO$_2$, ..., DO$_d$ each with a sensitive database, $\mathbf{D}_1^s$, $\mathbf{D}_2^s$, ..., $\mathbf{D}_d^s$, respectively, where $d > 1$, that need to be linked. Privacy-preserving record linkage across all these databases determines which record pairs (r_a, r_b), where $r_a \in \mathbf{D}_i^s$,

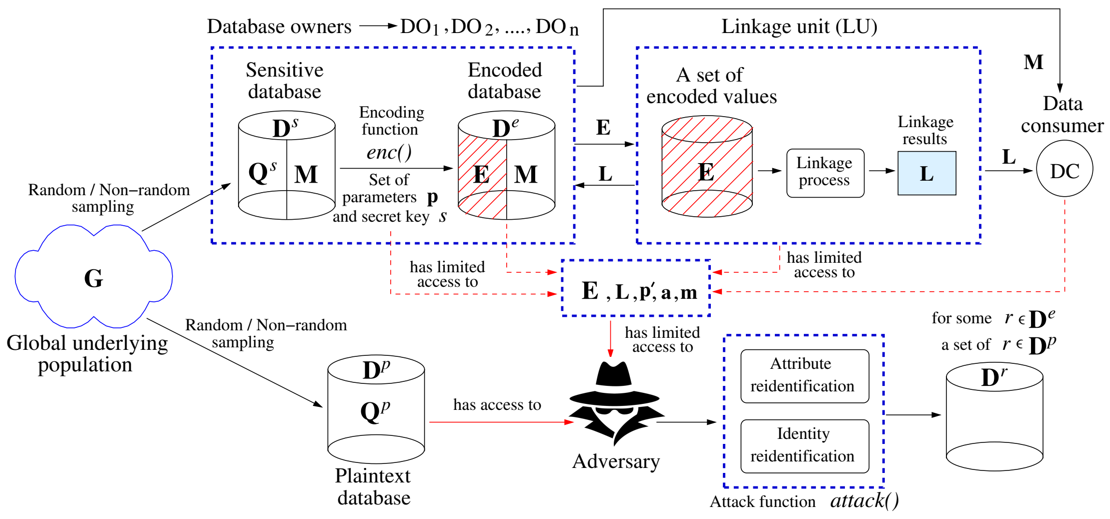

Figure 1: General overview of a privacy attack on PPRL, as described in detail in Section 3. We use the dashed line box in the top left to illustrate the steps conducted by the database owners (DOs) in the linkage protocol. The adversary runs an attack function *attack*() using the information she has access to, with the aim to reidentify plaintext QID values for a set of encoded records in $\mathbf{E}$.

$r_b \in \mathbf{D}_j^s$, $1 \leq a \leq |\mathbf{D}_i^s|$, $1 \leq b \leq |\mathbf{D}_j^s|$, $1 \leq i, j \leq d$, and $i \neq j$, represent the same real-world entity according to a decision function without revealing any sensitive information about these entities. At the end of the PPRL process each DO only learns which records in its sensitive database have been classified as matches with records in any of the other databases without learning the actual values of any records in the other databases.

3. Overview of Privacy-Preserving Record Linkage Attacks

In this section, we discuss the basic building blocks of an attack on PPRL. We describe the underlying assumptions that an attack makes in order to successfully reidentify encoded sensitive values, and then provide an overview of the main steps of a privacy attack. We discuss actual attack techniques that have been proposed in the context of PPRL following these steps in Appendix D.

3.1. **Underlying Assumptions.** In Figure 1 we outline the fundamental steps of a privacy attack on PPRL. As illustrated, we assume a global underlying population $\mathbf{G}$ which is not seen by an adversary. We also assume the sensitive database $\mathbf{D}^s = (\mathbf{Q}^s, \mathbf{M})$ and the publicly available plaintext database $\mathbf{D}^p = (\mathbf{Q}^p)$ have been sampled from $\mathbf{G}$. The sensitive database $\mathbf{D}^s$ can be owned by an organisation such as a hospital, a financial institution, a government department, or a security or military agency. $\mathbf{D}^s$ contains a list of QID values $\mathbf{Q}^s$ for a set of real-world entities $\mathcal{E}^s$. The sensitive database $\mathbf{D}^s$ also contains microdata $\mathbf{M}$ of the entities $\mathcal{E}^s$. We assume that the sensitive database $\mathbf{D}^s$ is deduplicated such that a real-world entity is represented by a maximum of one record (Christen, 2012).

Similar to the sensitive database $\mathbf{D}^s$, the plaintext database $\mathbf{D}^p$ also contains a list of QID values $\mathbf{Q}^p$ for a set of real-world entities $\mathcal{E}^p$. We assume that an adversary has access to this plaintext database $\mathbf{D}^p$. Such a plaintext database can be sourced externally (such as a public telephone directory or voter database) or internally (if the attack is conducted by an insider, as we discuss in Section 4.1.1, who has access to a plaintext database that has some overlap with the sensitive database $\mathbf{D}^s$). The plaintext database $\mathbf{D}^p$ may or may not contain microdata. If $\mathbf{D}^p$ contains any microdata we assume that those are anonymised and cannot be used in the attack (Domingo-Ferrer et al., 2015; Elliot et al., 2016).

3.2. **Main Steps of an Attack on PPRL.** Based on the functionalities of existing privacy attacks on PPRL, we now describe the main steps of a privacy attack. We assume that an adversary has access to the plaintext database $\mathbf{D}^p$ and the encoded database $\mathbf{D}^e$. The adversary does not have access to the sensitive database $\mathbf{D}^s$, and therefore she does not know which set of encoded QID values $\mathbf{e}_i \in \mathbf{E}$ belongs to which set of plaintext QID values $\mathbf{q}_i \in \mathbf{Q}^p$, and which record $r \in \mathbf{D}^e$ represents which entity $\epsilon \in \mathcal{E}^p$.

We define an attack function *attack*(), where this function consists of two steps. The first step is *attribute reidentification*, where the adversary runs a set of algorithms on the encoded data she has access to and aims to identify which plaintext QID values are encoded in each encoding $e \in \mathbf{E}$. For instance, in BF encoding (Schnell et al., 2009), bit patterns in a set of BFs can be used to identify q-grams that have been encoded in each BF in an encoded database (Kuzu et al., 2011). Privacy attacks on PPRL aim to identify a possible relationship between encoded and plaintext values in $\mathbf{D}^e$ and $\mathbf{D}^p$, respectively, and then use this relationship to reidentify encoded QID values.

The second step of the function *attack*() is *identity reidentification*. Based on the QID values identified in the first step, a set of encodings in $\mathbf{E}$ is assigned to a set of records in $\mathbf{D}^p$. Essentially identity reidentification focuses on assigning encodings $e \in \mathbf{E}$ to real-world entities $\epsilon \in \mathcal{E}^p$. We can formally define the attack function *attack*() as follows:

$$\mathbf{D}^r = attack(\mathbf{E}, \mathbf{D}^p, \mathbf{L}, \mathbf{p}', \mathbf{a}, \mathbf{m}), \tag{3.1}$$

where $\mathbf{L}$ are the linkage results (the identifiers of the matched records), $\mathbf{p}'$ is a set of parameter settings, $\mathbf{a}$ is a set of encoding and attack algorithms, and $\mathbf{m}$ are the metadata of encoded sensitive data that the adversary has access to. It is important to note that it might not be necessary nor possible for an adversary to have access to all the information mentioned above in a given attack scenario. Different attacks have been proposed (as we discuss in Appendix D) which require either all or some of the above information. The set of algorithms $\mathbf{a}$ includes encoding, hashing, frequency alignment, pattern recognition, or graph similarity matching (Christen et al., 2020). The metadata $\mathbf{m}$ can contain information about data quality such as completeness and validity of the data, the date when the data were recorded, the database size, ownership details of the data, and so on. $\mathbf{D}^r$ is the reidentified database of records that is a subset of records from $\mathbf{D}^e$. For each encoded record $r \in \mathbf{D}^r$, one or more plaintext records $r \in \mathbf{D}^p$ can potentially be assigned by the attack.

4. A Taxonomy of Privacy-Preserving Record Linkage Attacks

In this section, we propose a taxonomy of attacks on PPRL, which will help to obtain a better understanding of existing attacks, identify the limitations and weaknesses of attack methods, and identify possible future research directions towards privacy attacks on PPRL,

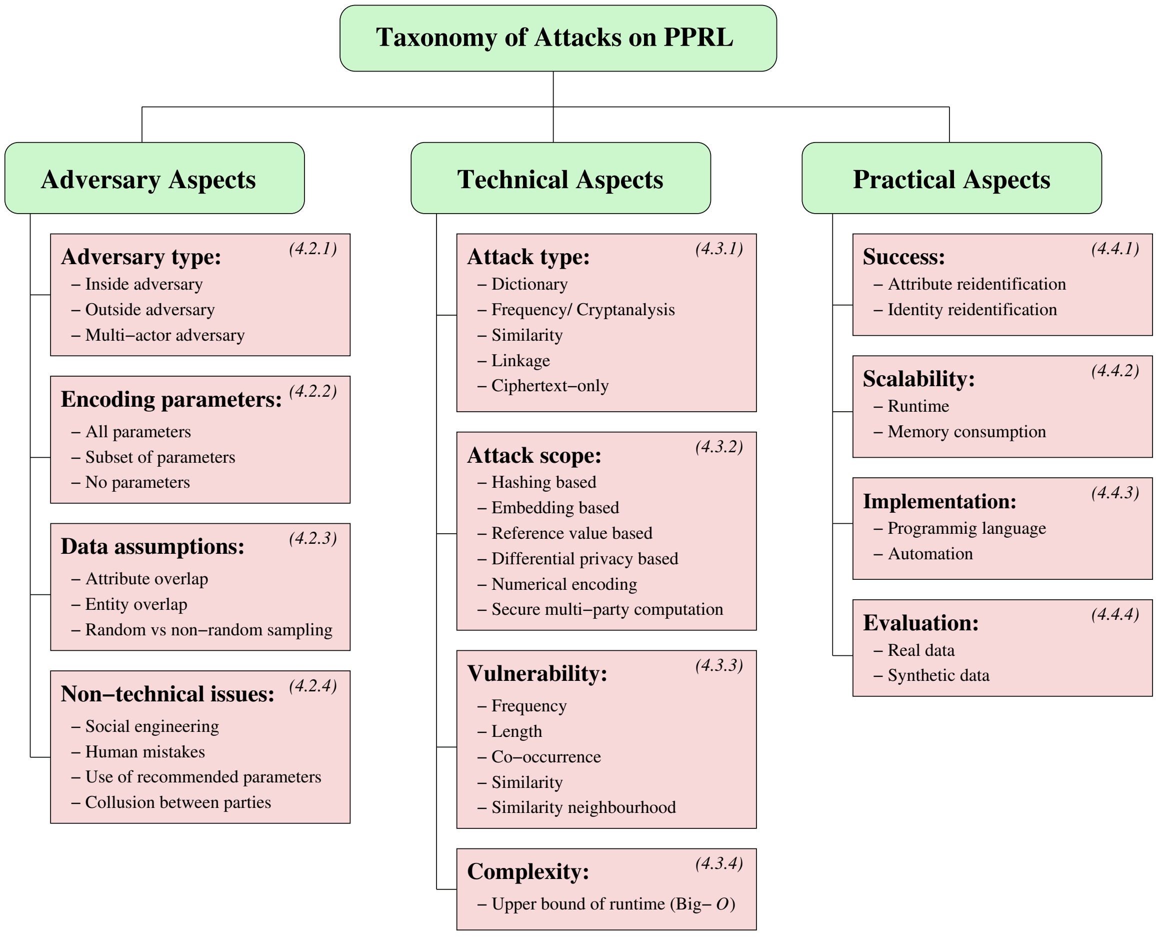

Figure 2: A taxonomy of attacks on PPRL described under three main aspects, where twelve dimensions are considered when characterising an attack. For each dimension we show the section number where it is being discussed (Vidanage, 2022)

as well as more secure encoding and encryption techniques for PPRL. We first provide an overview of the taxonomy and then discuss each aspect of the taxonomy in detail in the following sections.

In Figure 2 we show the three main aspects that can be considered when characterising an attack on PPRL. As can be seen, twelve dimensions are identified under these three aspects (Vidanage, 2022). We first provide a brief overview of each.

- **Adversarial Aspects** focus on characterising the adversary, assumptions about the prior knowledge of the adversary, and non-technical issues related to the adversary. We characterise three different adversary types depending on the information that an adversary has access to, and the collusion between different parties. The assumptions about an adversary's knowledge can be divided into two main dimensions: knowledge about the encoding parameters and assumptions about the database being attacked. We also characterise four different non-technical issues that the adversary can utilise, including social engineering and human mistakes.

- **Technical Aspects** cover the dimensions attack type, attack scope, vulnerability, and complexity. We characterise an attack on PPRL in terms of the actual techniques used to reidentify encoded sensitive values, and identify the attack scope that defines the encoding methods upon which an attack can be applied to. We then classify different vulnerabilities of encoded values that are being exploited by an attack to reidentify encoded sensitive values, and we describe how the complexity of an attack can be assessed using the big-O notation.
- **Practical Aspects** are important in understanding how a privacy attack could be applied in a real-world situation. We discuss how the success of an attack can be quantified based on the numbers of correct attribute and identity reidentifications achieved, and how the scalability of an attack can be measured using runtime and memory consumption. We also cover different implementation techniques used to develop existing attacks on PPRL, and the databases used to experimentally evaluate attacks on PPRL, respectively.

In the following, we discuss the twelve dimensions in these three aspects in more detail. We also provide a characterisation of privacy attacks on PPRL using our taxonomy in Table 1.

4.1. **Adversarial Aspects.**

We first discuss the different types of adversaries who can conduct an attack and then describe an adversary's accessibility to encoding function and parameter settings, and data and domain related information. Finally, we describe the possible impacts of non-technical issues for an attack.

4.1.1. *Different Types of Adversaries.*

Based on the parties that are internal or external to a linkage protocol, as we discussed in Section 2.1, there can be three types of adversaries involved in an attack: an inside adversary, an outside adversary, and a multi-actor adversary.

Inside Adversary (IA): An IA can be a person inside the organisation or a participant in a linkage protocol who tries to learn information about the sensitive data that are being encoded. An insider might have access to the encoded database, linkage results, encoding algorithms, and potentially the encoding parameters and even the secret key(s) used (Mitchell et al., 2017). All parties discussed in Section 2.1 (DO, LU, and DC) can be an IA (Vidanage, 2022). An IA can also be someone who is not among the main parties in a linkage protocol. For instance, an ex-employee who used to work in linkage projects will potentially know details about the used encoding function and previously used parameter settings. Furthermore, an employee inside the organisation might gain access to encoded values using company databases or another employee's login credentials.

Outside Adversary (OA): An external party that does not participate in a linkage protocol and does not have access to the encoding algorithms and/or encoding parameters used, yet that aims to learn about the sensitive information encoded in the databases that are being linked, can be considered as an OA. An OA can gain access to encoded data, **E**, or microdata, **M**, in different ways. For instance, if **M** is made public after the encoding or anonymisation of QID values under the assumption that no information could be linked to unique entities, these microdata can be accessed by possibly anyone. Such microdata can contain aggregated QID values such as age groups, gender, and/or zipcode. In such a scenario an OA can potentially conduct a linkage attack using another known plaintext database with similar QID values to reidentify records (Sweeney, 2001). We discuss how a

Table 1: A characterisation of existing privacy attacks using the twelve dimensions described in Figure 2. We use the following abbreviations: SE - Social engineering, HM - Human mistakes, RP - Use of recommended parameters; BF - Bloom filters, TMH - Tabulation min-hashing, 2SH - Two-step hashing, MMK - Multiple match-keys, HMAC - Keyed-hash message authentication code, ST - Similarity tables, NCVR - North Carolina voter registration database, GN - German names, GNA - German names and addresses, ALN - Australian last names, MVR - Michigan voter registration database, UKCD - Rawtenstall (England) census database, TITANIC - Titanic passenger database, and EURO - European census database (synthetic). For complexity analysis we assumed $n = |\mathbf{D}^e| = |\mathbf{D}^p|$, Q = the set of all q-grams from the QID values in $\mathbf{D}^p$, and $m = |A^p| = |A^s|$.

Attack	**Adversarial aspects**				**Technical aspects**				**Practical aspects**			
method	**Adversary type**	**Non-tech. issues**	**Encoding param.**	**Data assumptions**	**Attack type**	**Attack scope**	**Exploited vulnerability**	**Complexity**	**Success** (# reident.)	**Scalability** (runtime)	**Implement.** (language)	**Eval.** (data)
Kuzu et al. (2011)	IA, OA	SE/HM	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \supset \mathcal{E}^s$	Cryptanalysis	BF	Frequency, Length, Co-occurrence	$O(2^n)$	400 out of 3,500 values	$>$ 20 min	Java	NCVR
Niedermeyer et al. (2014)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \supset \mathcal{E}^s$	Cryptanalysis	BF	Frequency	$O(n)$	934 out of 10,000 values	multiple days	R	GN
Kroll and Steinmetzer (2015)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \supset \mathcal{E}^s$	Cryptanalysis	BF	Frequency, Co-occurrence	$O(n + \|Q\|!)$	44,000 out of 100,000 records	$>$ 402 min	C++/ Python	GNA
Christen et al. (2017)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$	Cryptanalysis	BF	Frequency	$O(n \cdot log\ n)$	6 to 10 out of top 100 values	$<$ 2 sec	Python	NCVR UKCD
Mitchell et al. (2017)	IA	SE/HM/RP	$\mathbf{p}' \equiv \mathbf{p}$, s	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \equiv \mathcal{E}^s$	Dictionary	BF	Co-occurrence	$O(n)$	364,450 out of 474,319 values	395 min	C++	NCVR
Culnane et al. (2017a)	IA, OA	HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \subset \mathcal{E}^s$	Similarity	HMAC and ST	Similarity neighbourhood	$O(n^2)$	93% out of 390,523 values	$<$ 1 day	Java	ALN
Christen et al. (2018a)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$	Cryptanalysis	BF	Frequency, Length	$O(n^2)$	41 to 27,665 out of 222,251 values	$<$ 22 min	Python	NCVR MVR, UKCD
Christen et al. (2018b)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$	Cryptanalysis	BF	Frequency, Co-occurrence	$O(n^2)$	10 to 5,000 out of 222,251 records	$<$ 150 min	Python	NCVR
Vidanage et al. (2019)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$	Cryptanalysis	BF	Frequency, Co-occurrence	$O(n^2)$	163 to 36,796 out of 222,251 records	$>$ 1 day	Python	NCVR
Vidanage et al. (2020b)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \supset \mathbf{E}$ $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$	Frequency	MMK	Frequency	$O(2^m \cdot n)$	14,415 out of 8,114,702 records	$<$ 210 min	Python	NCVR MVR
Vidanage et al. (2020a)	IA, OA	SE/HM/RP	$\mathbf{p}' \subset \mathbf{p}$	$\mathbf{Q}^p \equiv \mathbf{E}$ $\mathcal{E}^p \equiv \mathcal{E}^s$	Similarity	BF, TMH, 2SH	Similarity neighbourhood	$O(n^2)$	55.1% out of 25,343 records	$>$ 1 day	Python	NCVR, TITANIC, EURO

linkage attack works in Section 4.2. Alternatively, an OA can gain access to the encoded and/or anonymised data as well as encoding parameter values illegally such as from stolen hard drives and using social engineering attacks (Vidanage, 2022).

Multi-actor Adversary (MA): An MA is a subset of participating parties in a linkage process (as we discussed in Section 2.1) that can collude with each other to learn information about another party's encoded sensitive values. In such a collusion, each party provides different information for an attack, such as details about the encoding process (the encoding parameter settings, algorithms, and secret keys used), details about the sensitive databases, technical expertise, and sometimes even financial support (Christen et al., 2020). There can be different types of collusion scenarios depending on the linkage protocol used and the number of parties involved in the linkage process (Ranbaduge et al., 2020b).

An IA can also seek additional information from a party outside a linkage protocol or an organisation, such as how certain algorithms work and how to attack them, and/or aim to gain access to plaintext databases. However, we assume an outside party cannot provide any additional knowledge about the encoding process that an inside party does not already know. Therefore, in such a scenario, the outside party becomes an inside party and we categorise it as an IA.

4.1.2. *Knowledge about the Encoding.* Knowledge about the encoding includes identifying the actual encoding method and the used encoding parameters $\mathbf{p}$. Based on the encoded values in $\mathbf{E}$ an adversary might be able to guess which encoding method was used. For instance, recent PPRL encoding techniques such as two-step hash encoding (Ranbaduge et al., 2020a) generate sets of integers for each record in a database, while the multiple match-key encoding method (Randall et al., 2019) generates a list of hash values per record, obtained by hashing subsets of QID values. Since there are currently no other encoding methods that generate similar outputs, the adversary can guess the encoding method used by a simple examination of encoded values. On the other hand, both BF encoding (Schnell et al., 2009) and tabulation min-hash encoding (Smith, 2017) result in a list of bit vectors as encodings of QID values. Therefore, it will be difficult for an adversary to guess which encoding method was used if she only had access to an encoded database of bit vectors.

The encoding parameters $\mathbf{p}$ consist of different settings depending on the actual encoding method used. For instance, parameters used for BF encoding are the BF length, the q-gram length, the number of hash functions, the hashing method, and the encoding method (Schnell et al., 2009). Some of these parameters, such as BF length, can be obtained by an adversary based on the length of the encoded values. However, it is difficult to guess other parameter values using a simple analysis of encoded values.

Given $\mathbf{p}$ and s, the set of parameter values and the secret key used for the encoding, respectively, and $\mathbf{p}'$, the set of parameter values that the adversary has access to (or can guess), we define three scenarios:

(1) $\mathbf{p}' \equiv \mathbf{p}$ and s: The adversary knows all the encoding parameters including any secret keys used in the encoding process. It is important to note that for certain encoding methods (such as BF encoding), having access to all the parameter settings does not necessarily mean that the adversary can simply reverse-engineer the encoding process to easily reidentify all values in an encoded database. Such an attack was proposed by Mitchell et al. (2017) on BF encoding where it was assumed that the adversary was one of the DOs that knows all encoding parameters and secret key(s) used.

(2) $\mathbf{p}' \subset \mathbf{p}$: The adversary knows a subset of the encoding parameters. This does not include any secret keys used. Most of the existing attacks on PPRL are proposed under this assumption. Different attacks make different assumptions about the adversary's actual knowledge of the parameter settings, as we discuss in Appendix D. Such prior knowledge of parameter settings allows the adversary to efficiently conduct a privacy attack without having to try all possible parameter settings.
(3) $\mathbf{p}' = \emptyset$: The adversary does not know any of the encoding parameters. Without having access to at least some of the parameter settings used for the encoding most existing attacks on PPRL will be impossible to perform. In such scenarios, the adversary might need to consider all possible combinations of values for different parameter settings, therefore making the attack not practical. However, even if the adversary does not have access to any of the parameter settings used, with certain encodings, she can potentially guess the encoding function, *enc*(), as we described above. Furthermore, real-world incidents based on linkage attacks (Sweeney, 2001; Narayanan and Shmatikov, 2008; Culnane et al., 2017b), as we discuss in Section 4.2.1, show that certain methods such as generalisation and randomisation techniques can be susceptible to privacy attacks even without the knowledge of the parameter settings used.

4.1.3. *Availability and Accessibility to Data.* All existing privacy attacks on PPRL assume that the adversary has access to a plaintext database $\mathbf{D}^p$ that has some degree of overlap with the sensitive database $\mathbf{D}^s$, and therefore by extension with the encoded database $\mathbf{D}^e$. Using the plaintext database the adversary can aim to reidentify values in the encoded database by performing for example frequency alignments between plaintext and encoded values (Christen et al., 2018a; Vidanage et al., 2019). However, depending upon the actual overlap of the two databases $\mathbf{D}^p$ and $\mathbf{D}^s$, the difficulty of conducting an attack can vary.

Conceptually a database can be considered as a matrix with rows corresponding to the records of individuals in the database while columns corresponding to the different attributes (including QIDs), where each record (row) has values for those attributes. A privacy attack can be more successful if the plaintext database $\mathbf{D}^p$ overlaps in both rows and columns with the sensitive database $\mathbf{D}^s$ because as the overlap between the value distributions in these two databases increases, there is a higher possibility that more encoded values in $\mathbf{D}^e$ can be reidentified. In Figure 3, we show 25 different scenarios of an adversary's accessibility to a plaintext database. Each scenario represents a different overlap of QIDs and records between the plaintext database $\mathbf{D}^p$ and the sensitive database $\mathbf{D}^s$. In the following we briefly discuss how different types of overlap can increase the difficulty of an attack, while in Appendix B we discuss the specific scenarios of overlaps in more detail.

Column-wise Overlap: The overlap between columns in $\mathbf{D}^p$ and $\mathbf{D}^s$ determines how much information about QIDs, such as *FirstName*, *LastName*, and *StreetAddress*, are common in both databases. Assuming A^s and A^p are the sets of QID names in $\mathbf{D}^s$ and $\mathbf{D}^p$, respectively, we define the five different scenarios of availability of QID information to an adversary as shown in the top row in Figure 3. The amount of information that the adversary has access to will reduce with less overlap between QID names. Therefore, when going from the first to the fifth scenario the difficulty of an attack will increase.

Row-wise Overlap: The individual records represented by the rows in the databases allow the adversary to perform different operations such as frequency or similarity distribution analysis of QID values or q-grams. Such operations are usually the fundamental building

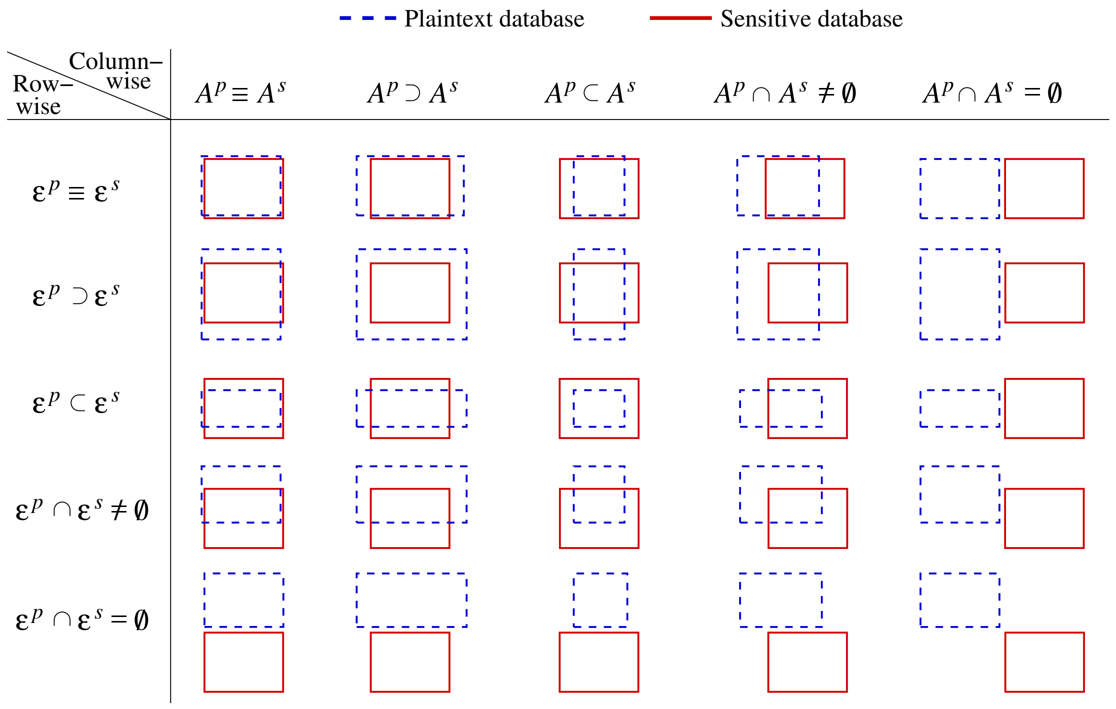
Plaintext database
Sensitive database
Column–wise
Row–wise
$A^p \equiv A^s$
$A^p \supset A^s$
$A^p \subset A^s$
$A^p \cap A^s \neq \emptyset$
$A^p \cap A^s = \emptyset$
$\varepsilon^p \equiv \varepsilon^s$
$\varepsilon^p \supset \varepsilon^s$
$\varepsilon^p \subset \varepsilon^s$
$\varepsilon^p \cap \varepsilon^s \neq \emptyset$
$\varepsilon^p \cap \varepsilon^s = \emptyset$

Figure 3: All 25 combinations of column-wise and row-wise overlap between the sensitive and plaintext databases $\mathbf{D}^s$ and $\mathbf{D}^p$. We consider the overlaps between the two sets of QID names A^s and A^p, and the two sets of entities $\mathcal{E}^s$ and $\mathcal{E}^p$ in column-wise and row-wise overlaps, respectively. Each combination illustrates a possible scenario of an adversary's accessibility to plaintext values. As can be seen these range from having access to a database with the same QID values and/or the same entities, to having access to a database with non-equivalent QID values and different entities compared to those in the sensitive database.

blocks of a privacy attack (Christen et al., 2020). If there is no overlap in the rows of the two databases $\mathbf{D}^s$ and $\mathbf{D}^p$ then conducting a privacy attack will be difficult. Here we assume that the rows in both databases $\mathbf{D}^s$ and $\mathbf{D}^p$ have been recorded at the same time with perfect data quality, and therefore there are no differences between the QID values of the same real-world entity. However, in practice, there can be significant temporal differences between $\mathbf{D}^s$ and $\mathbf{D}^p$ (such as two census snapshots recorded with a large time gap between them), which can lead to differences in QID values such as *LastName* and *StreetAddress* for the same entity. There can also be various types of errors and missing data in the QID values of real-world databases (Christen et al., 2020). Assuming each row in the sensitive and plaintext databases are from two sets of real-world entities $\mathcal{E}^s$ and $\mathcal{E}^p$, respectively, there can be five scenarios of accessibility as shown in Figure 3. Similar to column-wise overlap, the lower the overlap between the sets of entities the less information the adversary will have access to and therefore the difficulty of an attack will likely increase.

Random versus non-random sampling: In Figure 1 we show that both the sensitive database $\mathbf{D}^s$ and the plaintext database $\mathbf{D}^p$ are sampled from the underlying population $\mathbf{G}$ either randomly or non-randomly. While random sampling might be considered (even preferred) in a theoretical analysis, this is likely not the case in a practical real-world situation Christen and Schnell (2021). For instance, assume $\mathbf{D}^s$ is a health database from a hospital in a specific state and $\mathbf{D}^p$ is a publicly available voter registration database from the same state. While the health database will include records about children, non-citizen residents, and travellers, the voter registration database will only contain records about adult citizens. These aspects will influence the randomness of the samples in both of these databases.

If both $\mathbf{D}^s$ and $\mathbf{D}^p$ are sampled randomly, every entity in $\mathbf{G}$ has a non-zero probability of being selected for inclusion into $\mathbf{D}^s$ and $\mathbf{D}^p$. Therefore, the characteristics, such as frequency distribution of QID values, of these databases will likely be more similar to each other, and the adversary will have access to those similar characteristics which can be used in an attack. On the other hand, if the databases are sampled non-randomly then different subsets of $\mathbf{G}$ will likely be selected for $\mathbf{D}^s$ and $\mathbf{D}^p$ and therefore the difficulty of an attack will increase because of potential biases that can be introduced, resulting in different characteristics of $\mathbf{D}^s$ and $\mathbf{D}^p$ such as different frequency distributions of QID values.

4.1.4. *Non-Technical Issues.* Non-technical issues focus on how human behaviour can affect both privacy and security[2] of a PPRL system. The disclosure of information via non-technical means can sometimes bypass all technical security measures that have been implemented to protect the privacy of entities (Winkler, 1996). Furthermore, certain non-technical issues we discuss below can lead to an OA to become an IA by obtaining confidential information about a PPRL protocol. Therefore, recognising such non-technical issues is as important as technical issues in order to control and minimise the privacy risks associated with a PPRL scenario. In the following, we discuss four such non-technical issues that potentially can be employed by an adversary in a privacy attack.

Social Engineering: Social engineering is the malicious manipulation of people through human interaction to gather confidential and/or harming information (Hadnagy, 2010). Techniques used in social engineering attacks include phishing, vishing, smishing, pretexting, and so on (Yeboah-Boateng and Amanor, 2014; Vidanage, 2022). Almost all existing privacy attacks proposed on PPRL make certain assumptions about an adversary's prior knowledge about the used encoding methods (Niedermeyer et al., 2014; Kroll and Steinmetzer, 2015; Christen et al., 2017; Mitchell et al., 2017). Through social engineering attacks, an adversary can potentially gain access to such required information, as we discussed in Section 3.2.

Human Mistakes: Human mistakes or human errors are a fact that should be taken into consideration in PPRL, yet they are often disregarded as trivial. Human mistakes refer to an action of a person that is not intended, not expected by a set of rules or an external observer, and that move the task or system outside its acceptable limits[3]. Human mistakes can occur due to bad decision making, stressful environments, human-computer interface issues, and poor use of available resources (Christen and Schnell, 2021). An empirical study by Liginlal et al. (2009) showed that human mistakes in information processing constitute for most cases of privacy breaches. In PPRL, human mistakes can occur in three main ways.

(1) Use of highly predictable secret keys/passwords: Despite warnings not to, people tend to use passwords or secret keys that can be easily remembered, such as dates of births,

[2]While privacy is concerned with protecting the identities of individuals when using their personal information, security is focused on safeguarding personal or otherwise sensitive data as well as the systems that those data are stored in (Bansal, 2017)

[3]See: `https://en.wikipedia.org/wiki/Human_error`

or the names of partners or family members. An adversary can potentially predict such passwords.

(2) Use of PPRL methods and parameter settings that have been shown to be vulnerable to privacy attacks: Although certain PPRL methods, such as one-way hash encoding, can be successfully attacked (Vidanage et al., 2020b), some organisations are using these methods. This might be due to the ease of implementation, or the organisation has built their infrastructure specific to that method.

(3) Allowing an outside adversary to become an inside adversary due to social engineering: For instance, an ex-employee of an organisation can gain access to encoded values and/or encoding parameter settings because the responsible authorities have failed to change the system passwords or remove that ex-employee from the lists of authorised users. Such incidents can occur due to irresponsible behaviour of people (Liginlal et al., 2009).

Use of Recommended Parameter Settings: The use of recommended, commonly known, or published parameter settings is another non-technical issue that can potentially help an adversary. Often in publications where PPRL methods are proposed, a set of parameter settings are recommended. In a real-world application of that PPRL method, if the same set of parameter settings are employed, then an adversary can guess those parameter settings just by referring to the relevant publication. Even if the PPRL method did not use the exact same set of recommended parameters, the published parameter settings will provide the adversary with a likely starting point to estimate the actual parameter values.

Collusion between Parties: In a collusion scenario, two or more parties in a linkage protocol (from multiple actors as we discussed in Section 4.1.1) work together to learn information about another party's sensitive data by sharing their own information such as the parameter settings used in the encoding process (Vatsalan et al., 2014; Ranbaduge et al., 2020b). As shown in Table 1, most existing privacy attacks on PPRL make different assumptions about an adversary's prior knowledge of certain parameter settings used in the encoding. One possibility of attaining encoding details is via collusion between parties. Collusion might happen for various reasons, which include financial gain, intellectual gain, and personal vendetta (Vidanage, 2022). For instance, a collusion between a DO and a researcher (DC) can be inspired by the additional information that the researcher can gain access to in order to conduct more sophisticated data analytics, while the DO can learn more about the linked data of another DO.

4.2. **Technical Aspects.**

In this section, we describe the main technical aspects related to attacks on PPRL, which are attack types and their scope, the different vulnerabilities exploited by these attacks, and measuring the complexity of an attack. We also discuss the possibility of linkage attacks and ciphertext-only attacks in a PPRL context.

4.2.1. *Attack Types.*

Based on the different techniques used, privacy attacks can be divided into four categories. Note that some existing attacks can be placed into more than one category because they use a combination of different techniques.

Dictionary Attacks: A dictionary attack is a method of reidentifying an encoded sensitive value by encoding a large number of likely possible plaintext values in a given list of values until a matching encoding is found. In the context of PPRL, a dictionary attack is possible when the adversary has access to the encoding function $enc()$ and the used set of parameters

p. In such a scenario the adversary can use *enc*() and **p** to encode a large number of QID values in a plaintext database and see which encoding matches with any encoded values. However, a dictionary attack becomes difficult if the adversary does not have access to **p**, or if the encoding method uses a secret key s (Niedermeyer et al., 2014) to generate these encodings. In such a scenario a brute-force attack needs to be conducted using all combinations of possible values for s and/or **p**.

Frequency Analysis based Attacks: This type of attack analyses the frequency distributions of values in **E** and compares these distributions with the distributions of values in $\mathbf{D}^p$. Frequency analysis of combinations of characters in different languages is commonly used to break classical ciphers such as substitution and Vigener ciphers (Singh, 2000). A frequency analysis can be conducted without requiring any or using only limited knowledge about the encoding function *enc*() and its parameters **p**. As we discussed in Section 4.1.3, having similar frequency distributions in $\mathbf{D}^p$ and **E** will depend on the row-wise overlap and the random/non-random sampling of the databases. Furthermore, when several QID values in a record are encoded together into a single encoding (such as record-level BFs (Durham, 2012)) just the alignment of frequent values will not be enough to accurately reidentify encoded values because only a limited amount of frequency information will be available that can be extracted (Kroll and Steinmetzer, 2015).

A variation of a frequency attack that utilises additional knowledge about an encoding method and other techniques along with frequency alignments to reidentify encoded values in a sensitive database is known as a cryptanalysis attack (Christen et al., 2020). In general, a cryptanalysis attack starts with a basic frequency alignment of plaintext and encoded values. The results of such an alignment are then used with other methods to successfully reidentify encoded sensitive values. These methods can include constraint satisfaction models (Kuzu et al., 2011), language models (Kroll and Steinmetzer, 2015), correlation analysis (Christen et al., 2018a), and pattern mining (Vidanage et al., 2019). Most existing privacy attacks proposed on BF encoding for PPRL can be categorised as frequency based cryptanalysis attacks, as we describe in Appendix D.

Similarity Attacks: Most PPRL encoding methods calculate approximate similarities between encoded records. Approximate similarities can also be calculated for plaintext QID values (for example using corresponding sets of q-grams) in a plaintext database. Often these similarities calculated using the plaintext and the corresponding encoded values have an approximate linear relationship (Vidanage et al., 2020a). A similarity attack can exploit such relationships between similarity distributions to match an encoded value with a plaintext QID value. One way of identifying such relationships is using similarity graphs to perform a matching of nodes across graphs (Culnane et al., 2017a; Vidanage et al., 2020a).

There are a number of other types of attacks that are not directly related to PPRL, but that have been used in the contexts of privacy-preserving data publishing and cryptography. In the following, we describe two such attack types and briefly discuss why these attack types have not been used in the context of PPRL.

Linkage Attacks: This type of attack attempts to reidentify entities in an anonymised database by linking anonymised records with available external information (such as a public database) based on the uniqueness of the values in these records. Unlike the other attack types, a linkage attack is not conducted on encoded values in the PPRL context, rather it can be conducted on generalised and/or randomised QID values or microdata (Domingo-Ferrer et al., 2015). Such QID values might include zip code, year of birth, gender, and age group. A DC can conduct a linkage attack, or a collusion between a DO and a DC can also facilitate a linkage attack as we discussed in Section 4.1.1, because a DC will have access to the microdata $\mathbf{M}$ from all DOs. No linkage attacks have been proposed in the context of PPRL because PPRL techniques are focused on encoding sensitive information, not on generalising and publishing them. However, several real-world linkage attacks have been proposed in the context of privacy-preserving data publishing (Sweeney, 2001; Narayanan and Shmatikov, 2008; Culnane et al., 2017b).

Ciphertext-only Attacks: In cryptography, ciphertext-only attacks are a basic type of attack where an adversary analyses ciphertext (or encoded values) with the aim to recover the plaintext values and/or the secret key using only the ciphertext (Katz and Lindell, 2007). In such an attack, the adversary exploits common characteristics of a certain language such as redundancies in words and common occurrences of characters. This type of attack is different from a known plaintext attack because in a ciphertext-only attack the adversary does not have access to any plaintext data that she can use in the attack.

In the context of PPRL, to the best of our knowledge, no attack has been proposed so far to reidentify encoded values in $\mathbf{D}^e$ without using a plaintext database $\mathbf{D}^p$. This is because PPRL methods encode QID values of people such as their names and addresses and the characteristics of such values can differ from population to population and can also change over time. Therefore, a successful attack cannot be conducted using only encoded data without having access to a plaintext database that contains such QID values.

4.2.2. *Attack Scope.* The scope of an attack on PPRL defines what PPRL techniques the attack can be applied on. As we discussed in Section 1, PPRL techniques can be categorised as SMC based and perturbation based techniques. All existing attacks on PPRL have been proposed for perturbation based techniques because these techniques do not have provable privacy guarantees and different privacy weaknesses exist in perturbation based techniques that can be exploited. However, if a SMC based technique calculates approximate similarities of records, that technique can potentially be attacked using a similarity attack as we discussed in the previous section. We divide perturbation based techniques into five categories:

- Hashing based techniques use one-way hash functions such as secure hash algorithms (SHA) (Katz and Lindell, 2007) to either convert sensitive plaintext values into hash codes or to map plaintext values into bit or numerical vectors (Dusserre et al., 1995; Churches and Christen, 2004; Schnell et al., 2009; Randall et al., 2019).
- Embedding based techniques map plaintext values into multi-dimensional vectors (Scannapieco et al., 2007; Yakout et al., 2009), where these vectors are then used to calculate the similarities between records.
- Reference value based techniques use publicly available lists of reference values to calculate similarities between sensitive QID values and reference values (Pang et al., 2009; Vaiwsri et al., 2018). These similarities are then shared between the DOs or sent to a LU to calculate the similarities between records.
- Differential privacy based techniques focus on anonymising QID values in a sensitive database by blocking values such that each block satisfies certain differential privacy requirements (He et al., 2017; Rao et al., 2019). The generated blocks are anonymised by adding noise, most commonly from the Laplace distribution (Dwork, 2006).

- Numerical encoding techniques focus on encoding numerical values in such a way that the distances between numerical values are preserved (Vatsalan and Christen, 2016; Karapiperis et al., 2017).

We refer the interested reader to the book by Christen et al. (2020) and the survey by Gkoulalas-Divanis et al. (2021) for further details about different PPRL techniques proposed under these categories. Almost all existing privacy attacks on PPRL exploit the weaknesses in hashing based encoding techniques and their different parameter settings (Kuzu et al., 2011; Kroll and Steinmetzer, 2015; Christen et al., 2017; Vidanage et al., 2020b,a), as we describe in Appendix D.

4.2.3. *Vulnerability.* In a privacy attack, an adversary can exploit one or more vulnerabilities of values in both the plaintext ($\mathbf{D}^p$) and the encoded ($\mathbf{D}^e$) databases in order to reidentify encoded sensitive values. A value in either $\mathbf{D}^p$ or $\mathbf{D}^e$ becomes vulnerable based on how unique or rare it is in its corresponding database. Following the concept of k-anonymity (Samarati, 2001), two parameters, ε and k, are used to measure the vulnerability of a value. In the following, we use $\mathbf{D}$ to represent both the plaintext and encoded databases, $\mathbf{D}^p$ and $\mathbf{D}^e$, and v_i to represent a plaintext or an encoded value in one of these databases, where $1 \leq i \leq |\mathbf{D}|$.

Vulnerability of a single value: A value $v_i \in \mathbf{D}$ becomes vulnerable if it is distinguishable from all other values in $\mathbf{D}$ according to some characteristic, such as frequency or length, as we discuss below. For a value $v_i \in \mathbf{D}$ and the set $\mathbf{v}_i = \{v_j : func(v_i, v_j) \leq \varepsilon, v_i \neq v_j\}$ of other values $v_j \in \mathbf{D}$ with a tolerance $\varepsilon \geq 0$, v_i becomes (ε, k)-vulnerable in the database $\mathbf{D}$ with regard to the function $func()$ if $0 \leq |\mathbf{v}_i| < k$.

Three types of vulnerabilities, which correspond to the function $func()$, of a single value can be exploited by an attack on PPRL: frequency, length, and similarity neighbourhood. We discuss each of these vulnerabilities in detail in Appendix C.

Vulnerability of a pair of values: A pair of values (v_i, v_j) becomes vulnerable if it is distinguishable from other pairs of values in $\mathbf{D}$ according to some characteristic, such as co-occurrence or similarity, as we discuss below. For a pair of values (v_i, v_j) where $v_i \in \mathbf{D}$ and $v_j \in \mathbf{D}$, and the set $\mathbf{v}_i = \{(v_a, v_b) : func((v_i, v_j), (v_a, v_b)) \leq \varepsilon, (v_i, v_j) \neq (v_a, v_b)\}$ of other pairs of values (v_a, v_b) where $v_a \in \mathbf{D}$ and $v_b \in \mathbf{D}$ with a tolerance $\varepsilon \geq 0$, the pair of values (v_i, v_j) becomes (ε, k)-vulnerable in $\mathbf{D}$ with regard to the function $func()$ if $0 \leq |\mathbf{v}_i| < k$. Note that for certain vulnerabilities this definition can be generalised to a set of three or more values.

Two types of vulnerabilities, which corresponds to the function $func()$, of a pair of values can be exploited by an attack on PPRL: co-occurrence and similarity. We discuss these two vulnerabilities in more detail in Appendix C.

It is worth noting that for a reidentification of an encoded value to be successful it not only needs to be vulnerable within the encoded database, but it also has to be assignable to a vulnerable plaintext value (Vidanage, 2022). For instance, if an encoded value has a unique frequency in the encoded database, there also needs to be a vulnerable value in the plaintext database with a similar frequency (based on ε) in order for that encoded value to be assignable to that plaintext value. A successful reidentification can occur only based on such assigned pairs or encoded to plaintext values.

4.2.4. *Complexity.* A theoretical analysis of the computational complexity of an attack can be performed using the big-O notation. The big-O notation describes the asymptotic upper bound for the complexity requirements of an algorithm in terms of the size of the input. Given n records in the encoded list of values, $n = |\mathbf{E}|$, the big-O notations of $O(log\ n)$, $O(n)$, $O(n\ log\ n)$, $O(n^c)$, $O(c^n)$, and $O(n!)$ represents logarithmic, linear, log-linear, polynomial, exponential, and factorial complexities, respectively. Only some attacks on PPRL have been analysed with regard to their computational complexity in terms of the big-O notation.

4.3. **Practical Aspects.** In this section, we describe the practical aspects related to privacy attacks on PPRL. We discuss how such an attack can be evaluated based on how accurate it is in reidentifying encoded QID values, and how scalable the attack is when applied to (large) real-world databases. We then discuss the implementation aspects of an attack and the databases that have been used to evaluate attacks on PPRL.

4.3.1. *Success.* Assessing the success of a privacy attack can be used to quantify the privacy protection provided by the encoding techniques that are being used in a linkage protocol. However, neither a unified set of measures nor a standard framework to measure the accuracy of an attack has been proposed so far. The accuracy measures utilised in existing attack methods are usually ad-hoc. In the following, we define a set of measures based on disclosure risks and the evaluations of existing attacks on PPRL to quantify the success of a privacy attack on PPRL (Christen et al., 2018a; Vidanage et al., 2019).

Attribute Reidentification Accuracy: This is based on attribute disclosure risk and measures the accuracy of reidentified attribute values in an encoded sensitive database. In attribute reidentification, the adversary is able to reidentify some of the actual QID values encoded in $\mathbf{D}^e$. For instance, the *FirstName*, *StreetAddress*, or *DateOfBirth* values of a record $r \in \mathbf{D}^s$ that are encoded in $\mathbf{D}^e$ have been correctly reidentified. However, attribute reidentification does not necessarily imply that the adversary is able to reidentify the actual real-world entity or entities represented by these records (Andreou et al., 2017). For instance, if the adversary was able to reidentify 100 records in $\mathbf{D}^s$ with the *FirstName* 'John' and the *LastName* 'Smith', then the adversary cannot distinguish with which record belongs to which actual John Smith in the real-world.

The construction principles of all existing attacks on PPRL are focused on identifying QID values encoded in $\mathbf{D}^e$ for individual records. Based on that aspect and the evaluation methods employed in these attacks, a set of metrics can be defined to measure the attribute reidentification accuracy of an attack. As used by earlier attacks on PPRL (Christen et al., 2017, 2018b; Vidanage et al., 2019), for a given total number of reidentifications, the accuracy of a privacy attack can then be assessed using six measures:

(1) Correct 1-to-1 matches: The number of correct alignments between encoded and plaintext values (reidentifications) where one encoded value is aligned with one plaintext value.
(2) Correct 1-to-*many* matches: The number of correct alignments between encoded and plaintext values where one encoded value is aligned with several plaintext values and the correct plaintext value is among those alignments.
(3) Correct *many*-to-1 matches: The number of correct alignments between encoded and plaintext values where multiple encoded values are aligned with one plaintext value, where that plaintext value is the correct one for all the encoded values. Note that *many*-to-1 matches, where multiple encoded values are aligned with one plaintext value,

can also be considered as multiple 1-to-1 matches where each encoded value is aligned with the plaintext value, because the amount of information the adversary was able to gain in both of these situations is the same.

(4) Correct *many*-to-*many* matches: The number of correct alignments between encoded and plaintext values where multiple encoded values are aligned with multiple plaintext values, and the correct plaintext values are among those alignments.
(5) Wrong matches: The number of wrong reidentifications in the total number of reidentifications.
(6) None-matches: The number of encoded values that had no reidentifications, which means the attack was not able to align any of the plaintext values to any encoded values.

Note that all six measures are calculated for a single QID at a time and can be based on the top t or all reidentifications that the attack was able to identify.

Identity Reidentification Accuracy: This is based on identity disclosure risk and measures the accuracy of reidentifying actual real-world entities using the attribute reidentification results of a privacy attack. If an adversary is able to reidentify individual entities (people) in an encoded database using a privacy attack then their identities are being compromised. It is important to note that in certain situations even with low attribute reidentification accuracy, there can be high identity reidentification risk for some entities due to the uniqueness of the combination of their QID values (Andreou et al., 2017). For instance, if an attack was only successful in reidentifying the *LastName* value of a certain record correctly, and if there is only one real-world entity that has this *LastName* value, then the identity of that person is compromised (Sweeney, 2000).

Despite the fact that certain QID values, such as year of birth, city, gender, and zipcode, can be more common among a large number of people, the combination of these QID values can make certain records unique. For instance, consider a scenario where a database has 100 records with the *YearOfBirth* value '1990' and 500 records with the *City* value 'Queanbeyan', but when combined, there is only one record with *YearOfBirth* '1990' and *City* 'Queanbeyan'. Then the identity of that particular person is disclosed. Sweeney (2000) was able to uniquely identify over 85% of the population in the United States using only the combination of values in the QIDs *Zipcode*, *Gender*, and *DateOfBirth*. Therefore, it is important to understand the disclosure risks of identities associated with attacks on PPRL.

Since identity reidentification is directly related to the uniqueness of a record, this uniqueness can be used to measure the accuracy of reidentification of identities in an attack. For a given encoded record $r \in \mathbf{D}^e$, assume that $\mathbf{c}$ is the set of attribute values reidentified correctly for that record. The probability of suspicion disclosure risk measure proposed by Vatsalan et al. (2014) uses the uniqueness of attribute values in a plaintext database to assess the risk of reidentification. This measure can be adapted to calculate the probability of suspicion, PS, for a record based on the set of reidentified attribute values $\mathbf{c}$ for that record. For the record r the normalised $PS()$ would be:

$$PS(r) = \frac{1/n_c - 1/n}{1 - 1/n}, \tag{4.1}$$

where n_c is the number of records in the plaintext database that have the exact same set of reidentified attribute values $\mathbf{c}$, and $n = |\mathbf{D}^p|$ is the total number of records in the plaintext database $\mathbf{D}^p$. Using the normalised $PS()$ values for all records in $\mathbf{D}^e$ that each has at least one reidentified plaintext value assigned to it, four aggregated privacy measures can be defined (Vatsalan et al., 2014):

(1) Maximum risk: Maximum $PS()$ value of any records in $\mathbf{D}^e$.
(2) Mean risk: Average $PS()$ value of all the records in $\mathbf{D}^e$.
(3) Median risk: Median $PS()$ value of all the records in $\mathbf{D}^e$.
(4) Marketer risk: The ratio of the numer of records in $\mathbf{D}^e$ that have a probability of suspicion of $PS() = 1$.

These four aggregated measures can be used to evaluate the overall accuracy of the identity reidentification of an attack.

4.3.2. *Scalability.* With the increasing amount of data being collected every day, it is important to assess the scalability of an attack in order to analyse how practical the attack can be when conducted on large encoded databases. The scalability of an attack can be evaluated based on its runtime and memory consumption. Certain existing attacks on PPRL (as we discuss in Appendix D) that are based on graph matching and maximal frequent pattern mining will likely not be scalable to very large databases because of their high memory requirement and long runtimes. However, some attacks that are based on frequency alignments have shown to be scalable to encoded databases with millions of records, as can be seen in Table 1.

4.3.3. *Implementation.* Different programming languages have been used to implement existing attacks on PPRL. For instance, as we show in Table 1, most existing attacks on PPRL are implemented using Python. It is important to know whether an attack has been implemented and evaluated empirically to understand its real-world applicability. Some attacks proposed in the literature provide source code to facilitate replication of experiments.

It is also important to understand whether an attack is fully automated or whether it includes manual processes which require human involvement. If an attack consists of manual processes it will likely reduce the practicality of that attack in certain real-world situations, especially when applied to large encoded databases. In the attack proposed by Niedermeyer et al. (2014), several steps have to be conducted manually. All the other attacks on PPRL we mention in Table 1 are fully automated.

4.3.4. *Databases.* An attack on PPRL can be evaluated using real or synthetic databases. In order to conduct a critical evaluation of an attack ideally multiple databases with different real-world data characteristics, such as different QID combinations, records with missing values, and attributes with errors, need to be used. However, because it is difficult to always obtain real databases with sensitive personal information, synthetic databases with similar characteristics can also be used in the evaluation process (Christen and Vatsalan, 2013). A comprehensive evaluation of an attack on a set of different databases will provide a better understanding of how well an attack will perform in different real-world situations. This will allow DOs to take necessary precautions within a PPRL protocol to prevent possible reidentification of values from such attacks.

5. Recommendations for Privacy-Preserving Record Linkage

We now provide a list of recommendations that should be followed when executing a real-world PPRL project in order to strengthen the privacy of the entities whose records are being linked. Following these recommendations will make the PPRL process more secure by making it more resilient to the attacks we discussed in Section 4.2.

5.1. **Technical Recommendations.**

Technical recommendations focus on providing directions for using suitable encoding techniques and parameters, and testing a sensitive database both before and after the encoding process for potential vulnerabilities.

(1) Use encoding methods and appropriate parameter settings that are not known to be vulnerable to any of the existing attack methods. Attribute-level Bloom filters (ABF), for example, are vulnerable to frequency based attacks (Kuzu et al., 2011; Christen et al., 2017). Furthermore, the double hashing method and cryptographic long-term key encoding method for BFs have also shown to be susceptible to certain privacy attacks. Therefore, when using BF encoding (Schnell et al., 2009), we recommend to (a) use the random hashing method (Schnell and Borgs, 2016), (b) encode values from several QIDs into one BF, such as record-level BF encoding as proposed by Durham (2012), and (c) apply BF hardening techniques that have shown to be resilient to existing attacks on PPRL (Ranbaduge and Schnell, 2020).

(2) Use salting to generate record specific encodings. Stable and reliable QID values such as year of birth and place of birth can be used as salts when hashing/encoding other QID values in records. This is an effective method to reduce the amount of frequency information that an adversary can gain from an encoded database. As we discussed in Sections 4.2.1 and 4.2.3, the frequencies of values is one of the main vulnerabilities that can be exploited by an adversary. However, salting requires QID values that are complete, clean, and that do not change over time.

(3) Use different encoding settings for different blocks (for example by using salting) if the used blocking method generates non-overlapping blocks (Vaiwsri et al., 2022). This can be a strong defence mechanism against existing attacks on PPRL because the generated encodings in distinct blocks will be different from one another and therefore the frequency information that can be extracted using encoded values will be reduced. This will potentially also limit the adversary's knowledge of the encoding settings as we discussed in Section 4.1.2. However, this method would not work if the blocks are overlapping since the same record will have different encodings depending on the blocks that record have been inserted into.

(4) Assess the trade-off between privacy, linkage quality, and scalability based on the specific requirements of a linkage project. If the linkage project requires minimal scalability with improved privacy we recommend to use SMC based techniques (Christen et al., 2020; Vidanage, 2022) for PPRL because these techniques are provably secure and so far no attacks have been proposed for SMC based techniques.

(5) Test for the vulnerabilities discussed in Section 4.2.3 in both the plaintext and encoded sensitive databases. Understanding how vulnerable certain QID values in sensitive databases are will allow the DOs to make necessary decisions with regard to selecting a suitable encoding technique, only encoding not vulnerable QID values, and potentially removing or perturbing certain vulnerable values.

(6) Test for the overlap between the sensitive database and databases that are publicly available. As we discussed in Section 4.1.3 and illustrated in Figure 3, both row-wise and column-wise overlaps can be measured. The column-wise overlap can be restricted by selecting QID values that are less likely to be publicly available yet which provide good linkage quality. Row-wise overlap, on the other hand, cannot be directly eliminated without removing records which is unlikely a feasible option for PPRL projects.

(7) Run existing attacks on PPRL on the encoded database using publicly available databases to test if it can be successfully attacked. A DO can employ measures described in Section 4.3.1 to assess the success of an attack. This is a practical approach to investigate whether there is enough information available in publicly available databases to attack a sensitive database that is encoded using a certain encoding method and a set of parameters. Furthermore, such an assessment will also help in deciding if an attack is feasible on the encoded database in terms of time and memory complexities. If an attack is possible then the DOs can change the used encoding technique accordingly.

(8) Perform parameter tuning based on the databases and their selected QID values for a linkage in such a way that the selected parameter values will provide good linkage quality as well as privacy protection. Not using any recommended parameter settings (as we discussed in Section 4.1.4) will potentially limit the adversary's ability to guess certain parameter values used for the linkage. This is because unlike the recommended parameter settings, the tuned parameter values for individual databases are not publicly available and the adversary might not be able to gain access to those parameter values. However, it is worth noting that some parameter settings used in certain PPRL methods, such as q-gram length and the hashing method for BF encoding, do only have a small number of suitable options.

(9) Accrediting of computer systems, software, and algorithms used for PPRL projects will ensure the security of these systems by evaluating them with regard to different security standards. Note that accreditation will not guarantee the security or privacy of the sensitive data themselves rather it will only provide a certification for the computer system, software, and (possibly) algorithms used.

5.2. **Non-Technical Recommendations.** These focus on non-technical factors that will influence the security and privacy of the sensitive data used in a PPRL project.

(1) Provide proper training for the parties and their employees involved in a PPRL project with suitable programs that instruct employees to effectively recognise and prevent social engineering attacks (as we discussed in Section 4.1.4). Such training should not be a one-time undertaking, but a continuous process where employees are regularly trained on how and when to protect the sensitive data that they are trusted with.

(2) Encourage all the parties involved in a PPRL project to follow best practice approaches at all times in every step of the process (as discussed in Section 2.2). These best practices include implementing organisational structures to protect sensitive information, such as incident response plans and following the Five Safes framework for risk assessment in data access (Desai et al., 2016), implementing data access privileges to different parties, developing data privacy and security policies within the organisation, always tracking and monitoring the usage of sensitive data, using strong passwords and secret keys that cannot be easily guessed, and so on.

(3) Follow ethical and legal standards such as the European Union General Data Protection Regulation (GDPR), the US Health Insurance Portability and Accountability Act (HIPAA)[4], or the Australian Privacy Principles (APPs)[5], as put forth by governments and other organisations to protect sensitive information and ensure the privacy of entities.

While the combination of these technical and non-technical recommendations will make a PPRL project more secure, given the lack of provable security of many existing PPRL methods, further research into more secure PPRL methods is required.

6. Conclusion and Future Work

In this paper, we have presented a novel taxonomy of attacks on privacy-preserving record linkage (PPRL). We have identified twelve dimensions that can be used to characterise privacy attacks on PPRL, and have provided a comprehensive discussion of different aspects related to attacks on PPRL. These include adversary types, assumed knowledge of the adversary, different attack types, the vulnerabilities exploited by attacks, and measuring the success of attacks on PPRL. The owners of sensitive databases can evaluate and compare different attacks on PPRL using our proposed taxonomy in order to identify both strengths and limitations associated with attacks and if they are successful on their sensitive databases. To the best of our knowledge, no such taxonomy currently exists in the literature.

Given that most existing attacks on PPRL can only be applied to certain PPRL techniques, it would be difficult to conduct a comprehensive evaluation of all attacks with regard to their applicability to different PPRL encoding and encryption techniques. However, as future work, we aim to perform an experimental evaluation on some of the existing privacy attacks on PPRL, and compare their success and performance with each other.

Acknowledgment

This work was partially funded by the Australian Research Council grant DP160101934 and Universities Australia and the German Academic Exchange Service (DAAD). We would like to thank Murat Kantarcioglu and Chris Culnane for providing technical details of their attacks on PPRL (Kuzu et al., 2011; Culnane et al., 2017a).

References

A. Andreou, O. Goga, and P. Loiseau. Identity vs. attribute disclosure risks for users with multiple social profiles. In *ASONAM*, pages 163–170, 2017. https://doi.org/10.1145/3110025.3110046.

M. Antoni and R. Schnell. The past, present and future of the German record linkage center. *Journal of Economics and Statistics*, 2017. https://doi.org/10.1515/jbnst-2017-1004.

Y. Aumann and Y. Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. In *TCC*, pages 137–156, Amsterdam, 2007. https://doi.org/10.1007/978-3-540-70936-7_8.

[4]See: https://www.hhs.gov/hipaa/index.html

[5]See: https://www.oaic.gov.au/privacy/australian-privacy-principles

G. Bansal. Distinguishing between privacy and security concerns: An empirical examination and scale validation. *Journal of Computer Information Systems*, 57(4):330–343, 2017. https://doi.org/10.1080/08874417.2016.1232981.

C. Bizer, T. Heath, and T. Berners-Lee. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227. IGI Global, 2011. https://doi.org/10.4018/jswis.2009081901.

J. H. Boyd, S. M. Randall, and A. M. Ferrante. Application of privacy-preserving techniques in operational record linkage centres. In *Med Data Privacy Handbook*, pages 267–287. Springer, 2015. https://doi.org/10.1007/978-3-319-23633-9_11.

P. Christen. *Data Matching – Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Data-Centric Systems and Appl. Springer, 2012. https://www.springer.com/de/book/9783642311635.

P. Christen and R. Schnell. Common misconceptions about population data. *arXiv preprint*, 2021. https://arxiv.org/abs/2112.10912v2.

P. Christen and D. Vatsalan. Flexible and extensible generation and corruption of personal data. In *ACM CIKM*, pages 1165–1168, San Francisco, 2013. https://doi.org/10.1145/2505515.2507815.

P. Christen, R. Schnell, D. Vatsalan, and T. Ranbaduge. Efficient cryptanalysis of Bloom filters for privacy-preserving record linkage. In *PAKDD*, pages 628–640, Jeju, Korea, 2017. https://doi.org/10.1007/978-3-319-57454-7_49.

P. Christen, T. Ranbaduge, D. Vatsalan, and R. Schnell. Precise and fast cryptanalysis for Bloom filter based privacy-preserving record linkage. *IEEE TKDE*, 2018a. https://doi.org/10.1109/TKDE.2018.2874004.

P. Christen, A. Vidanage, T. Ranbaduge, and R. Schnell. Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In *PAKDD*, pages 530–542, Melbourne, 2018b. https://doi.org/10.1007/978-3-319-93040-4_42.

P. Christen, T. Ranbaduge, and R. Schnell. *Linking Sensitive Data – Methods and Techniques for Practical Privacy-Preserving Information Sharing*. Springer, 2020. https://www.springer.com/gp/book/9783030597054.

T. Churches and P. Christen. Some methods for blindfolded record linkage. *BMC Medical Informatics and Decision Making*, 4(9), 2004. https://doi.org/10.1186/1472-6947-4-9.

C. Culnane, B. Rubinstein, and V. Teague. Vulnerabilities in the use of similarity tables in combination with pseudonymisation to preserve data privacy in the UK Office for National Statistics' privacy-preserving record linkage. *arXiv Preprint*, 2017a. https://arxiv.org/abs/1712.00871.

C. Culnane, B. I. Rubinstein, and V. Teague. Health data in an open world. *arXiv Preprint*, 2017b. https://arxiv.org/abs/1712.05627.

T. Desai, F. Ritchie, and R. Welpton. Five safes: Designing data access for research. Technical report, Department of Accounting, Economics and Finance, Bristol Business School, University of the West of England, 2016. https://uwe-repository.worktribe.com/output/914745/five-safes-designing-data-access-for-research.

J. Domingo-Ferrer and V. Torra. Disclosure risk assessment in statistical microdata protection via advanced record linkage. *Statistics and Computing*, 13(4):343–354, 2003. https://doi.org/10.1023/A:1025666923033.

J. Domingo-Ferrer, S. Ricci, and J. Soria-Comas. Disclosure risk assessment via record linkage by a maximum-knowledge attacker. In *PST*, pages 28–35, Los Alamitos, USA, 2015. https://doi.org/10.1109/PST.2015.7232951.

E. A. Durham. *A Framework for Accurate, Efficient Private Record Linkage.* PhD thesis, Faculty of the Graduate School of Vanderbilt University, Nashville, TN, 2012. http://hdl.handle.net/1803/11417.

L. Dusserre, C. Quantin, and H. Bouzelat. A one way public key cryptosystem for the linkage of nominal files in epidemiological studies. *Medinfo*, 8:644–647, 1995. https://pubmed.ncbi.nlm.nih.gov/8591288/.

C. Dwork. Differential privacy. *Automata, languages and programming*, pages 1–12, 2006. https://doi.org/10.1007/11787006_1.

M. Elliot, E. Mackey, K. O'Hara, and C. Tudor. *The Anonymisation Decision-making Framework.* UKAN Manchester, 2016. https://eprints.soton.ac.uk/399692/.

A. Gkoulalas-Divanis, D. Vatsalan, D. Karapiperis, and M. Kantarcioglu. Modern privacy-preserving record linkage techniques: An overview. *IEEE TIFS*, pages 4966–4987, 2021. https://doi.org/10.1109/TIFS.2021.3114026.

C. Hadnagy. *Social engineering: The art of human hacking.* John Wiley & Sons, 2010. https://www.wiley.com/en-au/Social+Engineering:+The+Art+of+Human+Hacking-p-9780470639535.

R. Hall and S. Fienberg. Privacy-preserving record linkage. In *PSD, Springer LNCS 6344*, pages 269–283, Corfu, Greece, 2010. https://doi.org/10.1007/978-3-642-15838-4_24.

X. He, A. Machanavajjhala, C. Flynn, and D. Srivastava. Composing differential privacy and secure computation: A case study on scaling private record linkage. In *ACM CCS*, pages 1389–1406, Dallas, 2017. https://dl.acm.org/doi/10.1145/3133956.3134030.

T. N. Herzog, F. J. Scheuren, and W. E. Winkler. *Data Quality and Record Linkage Techniques.* Springer, 2007. https://www.springer.com/gp/book/9780387695020.

A. Karakasidis and V. Verykios. Secure blocking + secure matching = secure record linkage. *JCSE*, 5(3), 2011. https://doi.org/10.5626/JCSE.2011.5.3.223.

D. Karapiperis, A. Gkoulalas-Divanis, and V. S. Verykios. Distance-aware encoding of numerical values for privacy-preserving record linkage. In *IEEE ICDE*, pages 135–138, San Diego, 2017. https://doi.org/10.1109/ICDE.2017.58.

J. Katz and Y. Lindell. *Introduction to modern cryptography.* CRC press, 2007.

H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. In *Internet RFCs*, 1997. https://dl.acm.org/doi/book/10.17487/RFC2104.

M. Kroll and S. Steinmetzer. Who is 1011011111...1110110010? automated cryptanalysis of Bloom filter encryptions of databases with several personal identifiers. In *BIOSTEC*, pages 341–356, Lisbon, 2015. https://doi.org/10.1007/978-3-319-27707-3_21.

M. Kuzu, M. Kantarcioglu, E. Durham, and B. Malin. A constraint satisfaction cryptanalysis of Bloom filters in private record linkage. In *PET*, pages 226–245, Waterloo, 2011. https://doi.org/10.1007/978-3-642-22263-4_13.

D. Liginlal, I. Sim, and L. Khansa. How significant is human error as a cause of privacy breaches? an empirical study and a framework for error management. *Computers and Security*, 28:215–228, 2009. https://doi.org/10.1016/j.cose.2008.11.003.

Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. *JPC*, 1(1):5, 2009. https://doi.org/10.29012/jpc.v1i1.566.

W. Mitchell, R. Dewri, R. Thurimella, and M. Roschke. A graph traversal attack on Bloom filter-based medical data aggregation. *IJBDI*, 4(4):217–226, 2017. https://doi.org/10.1504/IJBDI.2017.10006842.

N. Mohammed, B. C. Fung, and M. Debbabi. Anonymity meets game theory: secure data integration with malicious participants. *VLDB Journal*, 20(4):567–588, 2011. https://doi.org/10.1007/s00778-010-0214-6.

A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE SP*, pages 111–125, 2008. https://doi.org/10.1109/SP.2008.33.

F. Niedermeyer, S. Steinmetzer, M. Kroll, and R. Schnell. Cryptanalysis of basic Bloom filters used for privacy preserving record linkage. *JPC*, 6(2), 2014. https://doi.org/10.29012/jpc.v6i2.640.

C. Pang, L. Gu, D. Hansen, and A. Maeder. Privacy-preserving fuzzy matching using a public reference table. *Intelligent Patient Management*, pages 71–89, 2009. https://doi.org/10.1007/978-3-642-00179-6_5.

R. Pita, C. Pinto, S. Sena, R. Fiaccone, L. Amorim, S. Reis, M. L. Barreto, S. Denaxas, and M. E. Barreto. On the accuracy and scalability of probabilistic data linkage over the Brazilian 114 million cohort. *IEEE JBHI*, 22(2):346–353, 2018. https://doi.org/10.1109/JBHI.2018.2796941.

T. Ranbaduge. *A Scalable Blocking Framework for Multidatabase Privacy-preserving Record Linkage*. PhD thesis, The Australian National University, Canberra, Australia, 2017. http://hdl.handle.net/1885/140918.

T. Ranbaduge and R. Schnell. Securing Bloom filters for privacy-preserving record linkage. In *ACM CIKM*, 2020. https://dl.acm.org/doi/abs/10.1145/3340531.3412105.

T. Ranbaduge, P. Christen, and R. Schnell. Secure and accurate two-sep hash encoding for privacy-preserving record linkage. In *PAKDD*, pages 139–151, Singapore, 2020a. https://doi.org/10.1007/978-3-030-47436-2_11.

T. Ranbaduge, D. Vatsalan, and P. Christen. Secure multi-party summation protocols: Are they secure enough under collusion? *TDP*, 13:25–60, 2020b.

S. Randall, A. Brown, A. Ferrante, and J. Boyd. Privacy preserving linkage using multiple dynamic match keys. *International Journal of Population Data Science*, 4, 2019. https://doi.org/10.23889/ijpds.v4i1.1094.

F.-Y. Rao, J. Cao, E. Bertino, and M. Kantarcioglu. Hybrid private record linkage: Separating differentially private synopses from matching records. *ACM TOPS*, 22(3):1–36, 2019. https://dl.acm.org/doi/10.1145/3318462.

P. Samarati. Protecting respondents' identities in microdata release. *IEEE TKDE*, 13(6): 1010–1027, 2001. https://doi.org/10.1109/69.971193.

M. Scannapieco, I. Figotin, E. Bertino, and A. K. Elmagarmid. Privacy preserving schema and data matching. In *ACM SIGMOD*, pages 653–664, Beijing, 2007. https://dl.acm.org/doi/10.1145/1247480.1247553.

R. L. Scheaffer, W. Mendenhall III, R. L. Ott, and K. G. Gerow. *Elementary survey sampling*. Cengage Learning, 2011.

R. Schnell and C. Borgs. Randomized response and balanced Bloom filters for privacy preserving record linkage. In *ICDMW DINA*, Barcelona, 2016. https://doi.org/10.1109/ICDMW.2016.0038.

R. Schnell, T. Bachteler, and J. Reiher. Privacy-preserving record linkage using Bloom filters. *BMC Medical Informatics and Decision Making*, 9(1), 2009. https://doi.org/10.1186/1472-6947-9-41.

S. Singh. *The code book: the secret history of codes and code-breaking*. Fourth Estate, 2000.

D. L. Smith. Secure pseudonymisation for privacy-preserving probabilistic record linkage. *JISA*, 34:271–279, 2017. https://doi.org/10.1016/j.jisa.2017.01.002.

L. Sweeney. Simple demographics often identify people uniquely. *Health (San Francisco)*, 671:1–34, 2000. https://doi.org/10.1184/R1/6625769.v1.

L. Sweeney. *Computational disclosure control: A primer on data privacy protection.* PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2001. https://dspace.mit.edu/handle/1721.1/8589.

L. Taylor, X.-H. Zhou, and P. Rise. A tutorial in assessing disclosure risk in microdata. *Statistics in Medicine*, 37(25):3693–3706, 2018. https://doi.org/10.1002/sim.7667.

UK Office for National Statistics. Beyond 2011 Safeguarding data for research: Our policy, 2013a. Methods and Policies Report M10.

UK Office for National Statistics. Beyond 2011 Matching anonymous data, 2013b. Methods and Policies Report M9.

J. Vaidya, C. Clifton, and M. Zhu. *Privacy Preserving Data Mining.* Springer, 2006. https://www.springer.com/gp/book/9780387258867.

S. Vaiwsri, T. Ranbaduge, and P. Christen. Reference values based hardening for Bloom filters based privacy-preserving record linkage. In *AusDM, CRPIT*, pages 189–202, Bathurst, 2018. https://doi.org/10.1007/978-981-13-6661-1_15.

S. Vaiwsri, T. Ranbaduge, P. Christen, and R. Schnell. Accurate privacy-preserving record linkage for databases with missing values. *Elsevier IS*, 106:101959, 2022.

D. Vatsalan and P. Christen. Privacy-preserving matching of similar patients. *JBI*, 59: 285–298, 2016. https://doi.org/10.1016/j.jbi.2015.12.004.

D. Vatsalan, P. Christen, and V. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Elsevier IS*, 38(6):946–969, 2013. https://doi.org/10.1016/j.is.2012.11.005.

D. Vatsalan, P. Christen, C. M. O'Keefe, and V. Verykios. An evaluation framework for privacy-preserving record linkage. *JPC*, 6(1), 2014. https://doi.org/10.29012/jpc.v6i1.636.

V. S. Verykios and P. Christen. Privacy-preserving record linkage. *WIREs Data Mining and Knowledge Discovery*, 3(5):321–332, 2013. https://doi.org/10.1002/widm.1101.

A. Vidanage. *Efficient Cryptanalysis Techniques for Privacy-Preserving Record Linkage.* PhD thesis, The Australian National University, Canberra, Australia, 2022. https://doi.org/10.25911/VSBZ-A727.

A. Vidanage, T. Ranbaduge, P. Christen, and R. Schnell. Efficient pattern mining based cryptanalysis for privacy-preserving record linkage. In *IEEE ICDE*, pages 1698–1701, Macau, 2019. https://doi.org/10.1109/ICDE.2019.00176.

A. Vidanage, P. Christen, T. Ranbaduge, and R. Schnell. A graph matching attack on privacy-preserving record linkage. In *ACM CIKM*, CIKM '20, page 1485–1494, 2020a. https://dl.acm.org/doi/abs/10.1145/3340531.3411931.

A. Vidanage, T. Ranbaduge, P. Christen, and S. Randall. A privacy attack on multiple dynamic match-key based privacy-preserving record linkage. *IJPDS*, 5(1), 2020b. https://doi.org/10.23889/ijpds.v5i1.1345.

I. S. Winkler. The non-technical threat to computing systems. *Computing Systems*, 9(1): 3–14, 1996.

M. Yakout, M. J. Atallah, and A. K. Elmagarmid. Efficient private record linkage. In *IEEE ICDE*, pages 1283–1286, Shanghai, 2009. https://doi.org/10.1109/ICDE.2009.221.

A. C.-C. Yao. How to generate and exchange secrets. In *IEEE SFCS*, pages 162–167, Toronto, 1986. https://doi.org/10.1109/SFCS.1986.25.

E. O. Yeboah-Boateng and P. M. Amanor. Phishing, smishing & vishing: an assessment of threats against mobile devices. *Journal of Emerging Trends in Computing and Information Sciences*, 5(4):297–307, 2014.

Appendix A. Adversary Models

In the context of privacy-preserving record linkage (PPRL), four different types of adversary models can be considered: fully trusted, honest-but-curious, malicious, and covert.

- **Fully Trusted**: In a fully trusted adversary model all parties involved in a linkage protocol follow all protocol steps without attempting to learn any information about any other party's sensitive data (Lindell and Pinkas, 2009). Furthermore, a fully trusted party does not try to alter the linkage process by sharing invalid or fake data (instead of valid and correct data) to compromise the integrity of the protocol. Therefore, in practice, a fully trusted model can be regarded as a model without a potential internal adversary. However, in real-world scenarios, not every party involved in a linkage situation can be fully trusted, where some parties might try to learn sensitive information from the data and linkage results they receive from the other parties.
- **Honest-But-Curious (HBC)**: In the HBC adversary model all parties follow the defined steps of a linkage protocol without deviating from them, while at the same time attempting to infer as much information as possible about the sensitive data of other DOs (Lindell and Pinkas, 2009; Christen et al., 2020). For instance, either the database owners (DOs) or a linkage unit (LU) who participates in a linkage protocol can use the data they receive from the other parties (including the linkage results) to conduct certain explorations, such as frequency alignments or cryptanalysis attacks. Therefore, for a linkage protocol to be considered as secure under the HBC adversary model, no participant of the protocol must be able to learn any sensitive information of any other parties' data. However, as we discussed in Section 4.1.1, collusion is also possible under the HBC model where more than one party that participate in the protocol can work together to infer sensitive information about the data of another party.
- **Malicious**: Under the malicious adversary model, the parties can behave maliciously by not following the necessary steps of the protocol (Lindell and Pinkas, 2009; Hall and Fienberg, 2010; Mohammed et al., 2011). A malicious party can send invalid or falsified data to other parties, change agreed parameter values, and even refuse to participate in the protocol. Therefore, in the malicious adversary model, the PPRL techniques used in a linkage protocol must guarantee that a malicious party will not be able to learn anything from the information it receives from any other party. Compared to the HBC model, achieving privacy under the malicious adversary model is more difficult since there are numerous ways for a party to deviate from the defined steps of a protocol.

  PPRL methods developed under the malicious adversary model generally use secure multi-party computation (SMC) based techniques (Yao, 1986; Lindell and Pinkas, 2009) to provide strong privacy guarantees to ensure that a malicious party will not be able to learn any information about the sensitive data of any other party. These PPRL methods also try to make sure that even if a participating party becomes malicious the protocol will still be able to compute the correct linkage results. However, SMC based PPRL methods generally incur high computation and communication costs that make them impractical for use in real-world linkage applications when large databases have to be linked.

- **Covert**: In practice, HBC based linkage protocols generally do not provide enough security, whereas the malicious model provides improved security at the cost of high computational complexity. The covert adversary model was proposed as a model that lies between the HBC and the malicious models, where it is suitable to be used in real-world linkage scenarios (Aumann and Lindell, 2007; Lindell and Pinkas, 2009). In the covert adversary model, it is assumed that the parties can behave maliciously and attempt to learn information about the sensitive data of other parties until they are being caught. This model assumes that an honest party in a linkage protocol can identify a cheating party with a defined probability (however, this probability will not be close to 1) (Christen et al., 2020). This adversary model can be used in real-world linkage situations where the participating parties cannot be fully trusted, yet they cannot afford to be identified as a malicious party due to the proceeding consequences such as loss of reputation, facing legal charges, or financial loss.

Appendix B. Overlap between Databases

In this appendix, we expand the discussion from Section 4.1.3 about the adversary's assumed availability and accessibility to plaintext data. Below we describe the overlap between the sensitive database $\mathbf{D}^s$ and the plaintext database $\mathbf{D}^p$ based on two overlap types, column-wise and row-wise, as illustrated in Figure 3.

Column-wise Overlap: Five scenarios of column-wise overlap can occur between the set of QIDs A^s from $\mathbf{D}^s$ and the set of QIDs A^p from $\mathbf{D}^p$.

(1) $A^p \equiv A^s$: The adversary has access to a set of QIDs from $\mathbf{D}^p$ that is equivalent to the set of QIDs in the sensitive database $\mathbf{D}^s$. For instance, if $\mathbf{D}^s$ contains the set of QIDs $A^s = \{$*FirstName, LastName, StreetAddress*$\}$, then $\mathbf{D}^p$ has the same set of QIDs. Depending on the number of common records in the two databases, as we discuss below in row-wise overlap, in this scenario there will likely to be a high overlap between QID values across both databases. Therefore, compared to the following scenarios, in this situation the adversary will likely have more information, such as more similar frequency distributions of QID values, that can be used in a privacy attack.

(2) $A^p \supset A^s$: The adversary has access to a set of QIDs from $\mathbf{D}^p$ that is a superset of the set of QIDs in $\mathbf{D}^s$. Compared to the above scenario, the adversary has access to the same set of QIDs used in $\mathbf{D}^s$ along with some additional QIDs. This scenario will be different from the above scenario when the adversary does not have specific knowledge about the QIDs used for the encoding. The adversary needs to perform an attack using different subsets of QIDs in A^p.

(3) $A^p \subset A^s$: The adversary has access to a set of QIDs from $\mathbf{D}^p$ that is a subset of the QIDs in $\mathbf{D}^s$. Compared to the previous two scenarios, under this scenario the amount of QID information that the adversary has access to is limited. For instance, the set of QIDs in $\mathbf{D}^s$ can be $A^s = \{$*FirstName, LastName, StreetAddress*$\}$ while the set of QID in $\mathbf{D}^p$ can be $A^p = \{$*FirstName, LastName*$\}$. An analysis of frequency distributions of QID values in this scenario can potentially be limited since the adversary does not have all the required QID attributes in $\mathbf{D}^p$.

(4) $A^p \cap A^s \neq \emptyset$: The adversary has access to a set of QIDs from $\mathbf{D}^p$ that has a non-empty intersection with the set of QIDs in $\mathbf{D}^s$. For instance, $\mathbf{D}^s$ has the set of QIDs $A^s = \{$*FirstName, LastName, StreetAddress*$\}$, whereas $\mathbf{D}^p$ has the set of QIDs $A^p =$

{*FirstName, LastName, Zipcode*}. In this scenario the adversary does not have access to all the QIDs used in the encoding process, while she has access to additional QIDs that are not present in the sensitive database. If the adversary does not know the QIDs used in the encoding process, conducting the attack will be difficult in this scenario compared to the previous scenarios.

(5) $A^p \cap A^s = \emptyset$: The adversary has access to a set of QIDs from $\mathbf{D}^p$ that is not equivalent to the set of QIDs in $\mathbf{D}^s$. If the adversary does not have access to any QIDs used in the encoding process, the availability of information that can be exploited in an attack, such as frequency information or relationships in similarities, will be less compared to the above four scenarios. Therefore, the success of a reidentification of encoded values by an attack is very unlikely in this scenario. However, having access to QIDs that are closely related to QIDs in the sensitive database such as *FirstName* $\in A^s$ and *MiddleName* $\in A^p$ can potentially lead to a frequency alignment of QID values or frequent q-grams.

Row-wise Overlap: Five scenarios of row-wise overlap can occur between the set of entities $\mathcal{E}^s$ from $\mathbf{D}^s$ and the set of entities $\mathcal{E}^p$ from $\mathbf{D}^p$.

(1) $\mathcal{E}^p \equiv \mathcal{E}^s$: The entities (rows) in both $\mathbf{D}^s$ and $\mathbf{D}^p$ are equivalent. If these databases also have similar QIDs (scenarios (1) to (4) as we discussed above in the column-wise overlap section) their frequency distributions can be highly similar in this scenario. Therefore, it might be easier to perform an attack using methods such as frequency alignments between values in records compared to the below scenarios.

(2) $\mathcal{E}^p \supset \mathcal{E}^s$: The entities recorded in $\mathbf{D}^p$ are a superset of the entities recorded in $\mathbf{D}^s$. Unlike the first scenario mentioned above, in this scenario the adversary has an extra set of entities in $\mathbf{D}^p$. The difficulty of conducting an attack in this scenario (and the next two scenarios) depends on two additional factors apart from the column-wise overlap: (a) the size of the row-wise overlap (number of common records) and (b) the sampling process of the two databases $\mathbf{D}^p$ and $\mathbf{D}^s$ (random vs non-random) as we discussed in Section 4.1.3. If the size of the row-wise overlap is larger than the number of non-overlapping records and if the two databases are assumed to be sampled at random from the global population $\mathbf{G}$, the frequency distributions of QID values can still be similar in both databases (Scheaffer et al., 2011). However, smaller row-wise overlaps as well as a non-random sampling of two databases (as we discussed in Section 4.1.3) can have an impact on the frequency distributions of values and therefore can reduce the accuracy of an attack.

(3) $\mathcal{E}^p \subset \mathcal{E}^s$: In this scenario the entities in the plaintext database are a subset of the entities in the sensitive database. Similar to the above scenario, the similarity in frequency distributions of the two databases $\mathbf{D}^p$ and $\mathbf{D}^s$ depend on the size of the row-wise overlap and the sampling process of the two databases.

(4) $\mathcal{E}^p \cap \mathcal{E}^s \neq \emptyset$: There is a non empty overlap between records (rows) in the encoded and plaintext databases. However, both databases contain entities that are not common. Hence, this scenario can be considered as a more practical scenario related to many real-world situations. Even with non-common records in both databases, if the number of overlapping records is larger than the number of non-overlapping records, there will likely be similar QID value distributions (Scheaffer et al., 2011) and the adversary can perform a frequency analysis of QID values to reidentify encoded values in $\mathbf{D}^e$.

(5) $\mathcal{E}^p \cap \mathcal{E}^s = \emptyset$: The overlap between records in the plaintext and sensitive database is empty. Hence, there are no real-world entities common in the two databases. Compared

to the previous scenarios, conducting an attack in this situation will likely be more difficult. Without enough frequency information on QID values, most existing privacy attacks cannot be performed (Kuzu et al., 2011; Niedermeyer et al., 2014; Christen et al., 2017). However, if the two databases are considered as samples from a population with similar QID value distributions across different categories (such as two samples from different age categories in the same state with similar name distributions), an attack can still likely be performed. For instance, if the sensitive database has two records with the name 'Peter Miller' and three records with the name 'Jackson Miller', and the plaintext database has five records with the name 'James Miller', even though the two databases do not contain records about the same entities the frequencies of the value 'Miller' will be the same. Conversely, if the two databases are from a population with different QID value distributions (for instance, two voter registration databases from the US states New York and Florida), identifying similarities in QID value distributions will likely be difficult.

Appendix C. Types of Vulnerabilities

Here we extend the discussion from Section 4.2.3 and describe five types of vulnerabilities that can exist in both encoded and plaintext databases. An adversary can exploit these vulnerabilities in order to reidentify sensitive plaintext values from encoded data.

(1) Frequency vulnerability is based on whether the frequency of a value is distinguishable from the frequencies of all other values in $\mathbf{D}$. For instance, if 'Smith' and 'Williams' are the most and the second most frequent last names in $\mathbf{D}$ with frequencies 500 and 450, respectively, then the value 'Smith' becomes frequency vulnerable under the privacy parameter settings $\varepsilon < 50$ and $k > 0$.

(2) Length vulnerability is based on whether the length of a value is distinguishable from the lengths of all other values in $\mathbf{D}$. For instance, if 'Eve' and 'Peter' are the shortest and the second shortest first names in $\mathbf{D}$ with lengths of 3 and 5 characters, respectively, then the value 'Eve' becomes length vulnerable under the privacy parameter settings $\varepsilon < 2$ and $k > 0$.

(3) Similarity neighbourhood vulnerability is based on whether the similarity neighbourhood of a value is distinguishable from the similarity neighbourhoods of all other values in $\mathbf{D}$. Given a similarity graph (Culnane et al., 2017a; Vidanage et al., 2020a) where nodes are the values in $\mathbf{D}$ and edges are the similarities calculated between these values, the neighbourhood of a value can be represented as a set of features calculated for that value based on its connected neighbours. These features can include the degree (number of neighbours), minimum, maximum, and average pairwise similarities between the value and its neighbours, the egonet and centrality of the value, and other graph characteristics (Vidanage et al., 2020a).

(4) Co-occurrence vulnerability is based on whether the co-occurrence frequency of a pair of values is distinguishable from the co-occurrence frequencies of all other pairs of values in $\mathbf{D}$. For instance, let us assume that the privacy parameters are set to $\varepsilon < 20$ and $k > 0$. The values 'Peter', 'Hank', and 'Parker' that have individual frequencies of 100, 95, and 85, respectively, will not be frequency vulnerable under the defined ε and k values. However, if the co-occurrence frequencies of ('Peter', 'Hank') and ('Peter', 'Parker') are 50 and 25, respectively, the value pair ('Peter', 'Hank') becomes co-occurrence vulnerable if no other pair of values in $\mathbf{D}$ has a co-occurrence frequency between 30 and 70.

(5) Similarity vulnerability is based on whether the similarity between a pair of values is distinguishable from the similarities between all other pairs of values in $\mathbf{D}$. For instance, let us assume that the privacy parameters are set to $\varepsilon < 0.05$ and $k > 0$. The Jaro string similarity (Christen, 2012) between 'Charlie' and 'Charles' is 0.91, and if no other pair of values in $\mathbf{D}$ has a similarity between 0.86 and 0.96, then the pair ('Charlie', 'Charles') becomes similarity vulnerable for these privacy parameter settings.

Appendix D. Existing Attacks on Privacy-Preserving Record Linkage

We now discuss, in chronological order, privacy attacks that have been proposed in the context of PPRL. We briefly describe how each of these attacks can reidentify encoded sensitive values by exploiting certain vulnerabilities in specific encoding methods. We also conceptually evaluate existing privacy attacks on PPRL in Table 1 with regard to the taxonomy illustrated in Figure 2, as we discussed in Section 4.

The first attack method for PPRL was proposed by Kuzu et al. (2011) for BF encoding (Schnell et al., 2009). The attack was based on two assumptions: (a) the sensitive database, $\mathbf{D}^s$, is a sample derived from a global database, $\mathbf{G}$, and the adversary has access to $\mathbf{G}$, and (a) the adversary knows the number of hash functions that were used to encode plaintext values into BFs. Using a frequency-aware constraint satisfaction problem (CSP) solver, the attack aligned frequent q-grams with matching BFs such that a certain set of constraints are satisfied. Once q-grams that have been hashed into different bit positions in BFs have been identified, the attack then reidentified the QID values that could have been encoded in each BF.

The second attack, as proposed by Niedermeyer et al. (2014), was a partially manual attack on BFs encoded using a single QID value. It exploits a weakness of the double hashing method used in the original BF encoding method proposed by Schnell et al. (2009). The attack assumed that the adversary knows the number of hash functions used for BF encoding, and has access to a plaintext database, $\mathbf{D}^p$, that has similar frequency distribution to one of the encoded values in $\mathbf{E}$. The attack aligned the most frequent q-grams with the most frequent bit patterns in BFs that can encode a single q-gram (called atoms) to reidentify encoded QID values. Kroll and Steinmetzer (2015) extended this attack into a fully automated cryptanalysis of BFs encoded using multiple QID values. Using an optimisation algorithm, the attack was able to identify the relationships between pairs of q-grams and pairs of atoms that were then used to reidentify encoded QID values.

Christen et al. (2017, 2018a) proposed a cryptanalysis attack on BF encoding that exploits the weaknesses of the BF construction principle. This attack can be considered a more practical and efficient attack method than the previous attacks. In the first step of the attack, the most frequent plaintext values and the most frequent BFs are aligned to identify sets of possible and not-possible q-grams for each bit position in BFs. Using the q-grams assigned to each bit position, the attack then reidentifies the QID values encoded into these frequent BFs. The attack assumes that the adversary has knowledge about the q-gram length and the QID combinations used for the BF encoding. Christen et al. (2018a) improved the accuracy of the attack by analysing the co-occurrences of bit positions and q-grams to expand the sets of q-grams that can be identified for each bit position.

A graph based attack on BF encoding was proposed by Mitchell et al. (2017). The attack assumes that the adversary has access to all the encoding information including the encoding function, $enc()$, and corresponding encoding algorithms, $\mathbf{a}$, encoding parameter values, $\mathbf{p}$, and the secret key, s, if any. Using a dictionary based technique, for each BF the attack first identifies a set of q-grams that can potentially be hashed into that BF. For each BF, a directed graph is built using the identified q-grams. These generated graphs are then traversed to reidentify encoded sensitive values by filtering only the feasible paths.

A similarity based graph attack was proposed by Culnane et al. (2017b) on a PPRL encoding method developed by the UK Office for National Statistics (2013a,b). This encoding method uses a keyed-hash message authentication code (HMAC) (Krawczyk et al., 1997) and similarity tables for the encoding of sensitive QID values. The similarity tables generated were used to build a graph where nodes in the graph represent the encoded QID values, and the edges represent the similarities between these values. A set of subgraphs was then generated using a plaintext database that is assumed to be accessible by the adversary. The attack then matched nodes across graphs using a graph isomorphism approach and reidentified encoded values from aligned nodes across two such graphs.

Christen et al. (2018b) and Vidanage et al. (2019) proposed a cryptanalysis attack on BF encoding using a pattern mining based approach. This is one of the most practical attacks on BF encoding for PPRL because it does not require knowledge of the encoding settings nor any frequent BFs in the encoded database. The attack only assumes that the adversary has access to a plaintext database, $\mathbf{D}^p$, and knows the q-gram length and QID combinations used for the BF encoding. First, the most frequent q-grams and their frequencies were identified in the plaintext database. Next, the attack employed a maximal frequent pattern mining approach to identify all co-occurring bit positions in BFs that can encode these frequent q-grams. The attack identified further q-grams and the bit positions they were hashed into using a language model where conditional probabilities between occurrences of q-gram pairs are analysed. All the identified q-grams and their corresponding bit positions are then used to reidentify encoded sensitive values in each BF.

Vidanage et al. (2020b) have recently proposed a privacy attack on the multiple dynamic match-key encoding method developed by Randall et al. (2019). This attack was based on aligning frequent plaintext QID combinations (called plaintext match-keys) with frequent encoded match-keys from the encoded database. The attack employs a set of statistical correlation measures to compare the frequency distributions of encoded match-key values with the frequency distributions of plaintext match-key values. The evaluation of this attack method using large real-world databases illustrated that the most frequent encoded sensitive values can be successfully reidentified by this frequency analysis.

The most recently proposed privacy attack on PPRL by Vidanage et al. (2020a) was a similarity attack which can be applied to any encoding method that calculates approximate similarities between encoded values. The attack assumes that the adversary has access to a plaintext database, $\mathbf{D}^p$, and knows the q-gram length and the QID combinations used for the encoding of sensitive values. Two similarity graphs are first built using the plaintext and the encoded databases where nodes in each graph represent the records in the corresponding database while edges represent the similarities calculated between those records. The attack aligns nodes in the encoded graph with nodes in the plaintext graph by comparing their similarity neighbourhoods based on a set of graph features such as degree, minimum, maximum, and average pairwise similarities between the value and its neighbours, and egonet and centrality of the value, and so on. This alignment allowed the attack to reidentify encoded sensitive values with high accuracy when applied on three different PPRL encoding techniques: Bloom filters (Schnell et al., 2009), tabulation min-hashing (Smith, 2017), and two-step hashing (Ranbaduge et al., 2020a).