Format-Preserving Encryption Creates a Privacy Attack Surface for Re-Identification

Martin Staal Boesgaard, Markus Juhl Larsen
{msb, mjl}@piiguard.com

Abstract

Format-preserving de-identification methods, for example format-preserving encryption, enable de-identified data to act as an in-place replacement for the original data by retaining syntactic properties. However, when applied to data types with multiple formats, format preservation introduces inherent information-theoretic leakage, as the format itself can reveal non-trivial information about the original data, creating an attack surface that can be realized when appropriate auxiliary information is available. We formalize format preservation and use Shannon entropy to quantify the resulting leakage. To illustrate the practical impact of this, we document real-world use of format-preserving de-identification on variable-format data types and apply the theory to a real-world dataset. Using personal data from Danish financial institutions, we find that a length and word-preserving transformation has a leakage of 10.12 bits for person names and 3.9 bits for cities, out of a maximum of 17.2 bits. While exploiting this leakage requires appropriate auxiliary information, such information is often readily available in practice. In the worst-case scenario, this can lead to re-identification of some data records; however, even in less extreme cases, it can significantly narrow down the search space for re-identification, e.g. by revealing the length of the original data, or the format of an e-mail domain.

Keywords: anonymization, pseudonymization, format-preserving encryption, de-identification, re-identification, misuse, formalization, information entropy, attack surface, privacy

1 Introduction

Seemingly every month, a new large-scale data breach is reported, often involving millions of personal records. These incidents represent only the publicly disclosed portion of a potentially much larger problem, as reporting criteria vary across countries and organizations and many breaches likely remain undiscovered or unreported for extended periods of time [1][2][3]. Some high-profile examples include the 2024 National Public Data breach [4], which leaked 2.9 billion records of US citizens—including full names, addresses, and Social Security numbers–and the 2013 Yahoo breach [5], which remains the largest publicly disclosed breach, affecting 3 billion accounts. While such breaches are well-known due to their scope, many smaller breaches also occur. The 2023 Data Breach Investigations Report [6], for example, analyzed 5199 confirmed breaches, highlighting that security incident and breach analysis has become a substantial field in its own right due to the prevalence of such events. Across these breaches, personal information and credentials were the most commonly compromised data types.

De-identification protects personal data by transforming it in ways that reduce or eliminate linkability to individuals. This can be achieved by removing direct identifiers (e.g., names, unique personal identifiers), obscuring the relationship between transformed and original values (e.g., through generalization), or rendering the data inaccessible without additional information (e.g., through encryption). However, even de-identified datasets may remain vulnerable to re-identification, where an adversary uses auxiliary information to link de-identified data back to individuals. Such attacks exploit any remaining information in the de-identified data that can be correlated with external data sources [7]. Massachusetts Governor William Weld's case is perhaps the most well-known example of re-identification, with health records being re-identified using voter records as the auxiliary information [8].

In many contexts—such as data sharing, testing, and analytics—data must remain useful in addition to being protected, which often necessitates preserving structural properties of the original data in the de-identified form. This means that it may not be possible to simply remove something like personally identifiable information (PII) or apply unconstrained de-identification transformations to it, as this can render the data unusable for its intended purposes. Preserving the type or structure of the original data in the de-identified data enhances the utility of the resulting data, ensures compatibility with existing systems and processes, and basically allows the de-identified

data to be used as a drop-in replacement for the original data. Methods that preserve the data type by, in some sense, preserving the *format* of the original data are termed format-preserving. This paper focuses on these format-preserving de-identification methods, with a specific focus on format-preserving encryption (FPE), a family of encryption algorithms designed to preserve data format from plaintext to ciphertext. There are different ways of constructing FPE schemes, such as using a Feistel network [9] or cycle-walking. It is a Feistel network approach that has been taken by the NIST-approved FPE specification FF1 [10]. FF1 defines a mode of operation for block ciphers (like AES) to operate as FPE. FF1 preserves format by ensuring that the input and output have the same length and the same character set.

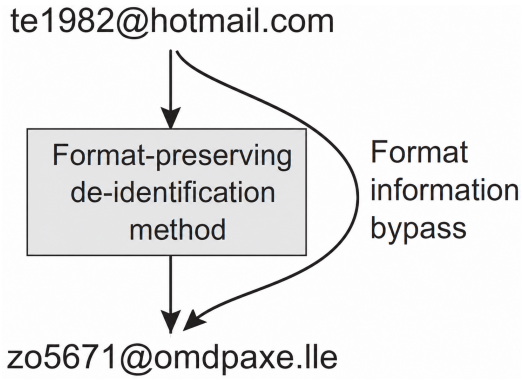

Figure 1: Text goes through de-identification whereas format passes through unmodified, leading to the format of the original data being leaked.

While format-preserving methods enable new use cases—for example, transforming research datasets to reduce re-identification risks without sacrificing usability—they also have inherent limitations. Specifically, preserving format necessarily reveals the format of the original data (see Figure 1). Continuing with FF1 as an format-preserving de-identification method, FF1 preserves the length of the input, and the output is constrained to a given character set. There is a significant information-theoretic leakage stemming from this, forming a potential attack surface for re-identification. The severity of the leakage depends on factors such as the diversity of formats for a given data type, which depends on the strictness of the format definition, and the data type itself. A data type like credit-card numbers (CCNs) with

length-preservation exhibits low vulnerability due to the small number of valid lengths and the uniqueness of CCNs. Personal names, on the other hand, exhibit substantial format variability—including differing numbers of middle names and lengths—and highly skewed distributions, making them significantly more susceptible to inference and educated guessing by adversaries. Exploiting this attack surface, however, requires appropriate auxiliary information containing sufficient data that can be linked based on format to the de-identified data, reducing the set of plausible matching candidates. In the worst case this can reduce the set of plausible candidates to just one, allowing for full re-identification. Re-identification is typically concerned with linking de-identified data directly to individuals, by matching auxiliary information with the de-identified data, in order to discover the individuals the data belongs to. The attack surface analyzed in this paper focuses on linking de-identified data to the original plaintext values, i.e., plaintext recovery. This recovery process resembles standard re-identification attacks, with auxiliary information used to link the de-identified data to the original data.

The format bypass inherent to format-preservation is well understood by practitioners, as it is the entire point of format-preserving de-identification. However, the information-theoretic leakage from this is not always fully appreciated, as the impact is very data type-dependent. This is largely because the use of format-preserving methods has evolved over time, originally designed for tightly constrained use cases such as encrypting credit card numbers. Now they are being applied to in much less constrained data types, as demonstrated in Section 4. These data types—e.g. names, addresses, e-mail addresses—exhibit substantial format variability, increasing the leakage. Due to this leakage being inherent to the methods, we classify this as a misuse of format-preserving de-identification methods, rather than a flaw in the methods themselves. It should be noted that we do not consider this leakage from a fully cryptographic perspective. In particular, we do not evaluate the computational security of any specific format-preserving de-identification method itself, nor do we present attacks on any such schemes. Instead, we focus on the information-theoretic leakage that can be exploited for re-identification. This leakage arises independently of the cryptographic strength of the underlying de-identification method, and can be viewed as a structural side-channel attack, in that it bypasses the ciphertext content, exploiting information—namely the preserved format—orthogonal to the ciphertext bits.

Structure of the paper: In Section 2, we detail some related work on re-identification, and some previous attempts at quantifying format-based leakage. Section 3 formalizes format-preserving de-identification and the associated leakage. In Section 4 we document real-world (mis)use of these de-identification techniques on variable-format data types. Section 5 presents a case study using real-world data from two Danish financial institutions, in which we analyze format-based leakage via a toy de-identification example on personal information (person names, e-mail addresses, addresses, cities). In Section 6, we discuss the broader implications of the identified problem and propose mitigation strategies, concluding the paper.

2 Related Work

There is extensive work on the dangers of re-identification, with Paul Ohm's article *Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization* [7] arguing that anonymization is fundamentally broken by re-identification. Published in 2010, this article brought the computer science understanding of re-identification into legal discourse on privacy and data protection. Ohm argues that legal definitions of anonymization are flawed because they fail to account for the possibility of re-identification, and that increasing digitalization and the widespread availability of data sources have made re-identification easier. He further argues that re-identification is *accretive* in nature, as successful re-identification abets future re-identification. Together, these observations suggest that traditional de-identification measures are insufficient to protect privacy, a problem further exacerbated by the rise of large-scale social media platforms since Ohm's original observations.

Two foundational examples of re-identification—in addition to that of Governor Weld—are the Netflix Prize dataset [11], and AOL search log release [12], both from 2006. In the Netflix case, Narayanan and Shmatikov demonstrated that users could be re-identified by correlating the supposedly anonymized movie ratings with non-anonymous public ratings from Internet Movie Database used as auxiliary information. In the AOL case, released search logs were used to construct behavioral profiles, ultimately linking user #4417749 to Thelma Arnold. In both cases, the de-identifying process underestimated the amount of information retained in the released data and failed to anticipate the amount and variety of data sources that could be leveraged for re-identification. These examples highlight general shortcom-

ings of with de-identification, with the re-identification risk arising from the presence of quasi-identifiers in the de-identified data. As we show in this paper, even format-preserving de-identification methods are vulnerable to such re-identification attacks; format can be used to recover partial information about the original values using auxiliary information, which may then be combined with additional auxiliary information in a standard re-identification attack. In this sense, format-constrained attributes can themselves function as quasi-identifers, even if they cannot be completely recovered.

Two recent papers that explicitly examine format-based information leakage in format-preserving de-identification methods are [13] and [14]. In [13], the authors analyze the privacy risks arising from preserved format information and propose a method for quantifying this leakage using a format-preserving encoding scheme to create format-based anonymity sets inspired by the concept of k-anonymity [15]. Using datasets containing personal information from two Danish financial institutions, they analyze the sizes of these anonymity sets and show that many records have unique formats for single attributes, with this more pronounced when combining attributes. They give a name-based re-identification example, where they focus on a 5-letter first-name and a 6-letter last-name, concluding based on this and name frequency data that the person is likely named some combination of {Mette, Hanne, Peter} as first name and {Jensen, Hansen} as last name. This is essentially an attempt at format-based re-identification, but it does does not attempt to mathematically quantify information-theoretic leakage (other than basic anonymity set statistics).

We attempt to rectify some of these shortcomings, with a focus on the formalization of the problem (see Section 3). We use a combined version of these two datasets, with a less strict preservation scheme in Section 5, where we use Shannon entropy as a leakage measure instead of focusing on format-based anonymity set sizes directly. We find the formats considered in [13] and [14] to be unnecessarily strict. The same format preservation is applied across different attributes, without accounting for the differences in data types. Furthermore, these formats distinguish between Nordic letters and non-Nordic letters, and between letters and digits, without clear justification for doing so. We detail the mapping we use to analyze the leakage for the same datasets in Section 5.

In [14], the pitfalls of using format-preserving de-identification methods

are analyzed using a format-based anonymity set approach similar to [13], using one of the same two datasets. Their methodology consists of constructing possible synthetic "victim datasets" capable of producing the format-based anonymity sets observed in the de-identified data. Two algorithms (Alg-Min and Alg-Max) are used to derive best- and worst-case victim datasets, forming lower and upper bounds for the re-identification difficulty. Even in the best-case scenario, they find that a significant portion of records across various combinations of attributes have unique formats, making many records uniquely distinguishable within that dataset.

Format-preserving encryption (FPE) is a well-studied field from a cryptographic perspective, with various security notions defined for FPE [16], and many different schemes proposed—some later shown to be insecure—culminating in the current NIST-approved FPE scheme FF1 [10]. However, FPE literature typically has a cryptographic outlook, in that it proposes new schemes, tries to break other schemes, and in general consider the security of the cipher itself, not the many use-cases where such methods are not appropriate. It is this perspective that we approach FPE with in this paper, and in Sections 3.1 and 3.2 we analyse the format-based leakage through a privacy lens, drawing inspiration from some of the FPE-focused security notions, with a focus on the practical implications for re-identification.

3 Format and Leakage Formalization

In this section we formalize what is meant by data type and format, and how format-preserving de-identification methods can be viewed as constrained transformations on strings. We then use Shannon entropy to quantify the information-theoretic leakage from format preservation, and detail what information can be leaked by format.

3.1 Format Definition

The purpose of format-preserving de-identification methods is to ensure the syntactic validity of the ciphertext, so that it can be used in place of the plaintext. In other words, the ciphertext should be of the same data type as the plaintext. The notion of data type, however, varies across fields, ranging from abstract types with associated operations in programming, to

constraint-based definitions used in data management and governance. We take the latter perspective, and define a data type as the set of strings that satisfy a given set of well-formedness constraints. Let Σ be an alphabet and R a decidable predicate of well-formedness (the "rules"), then the associated data type D is defined as

$$D = \{s \in \Sigma^* \mid R(s) = \mathsf{true}\}.$$

For U.S. Social Security numbers, Σ is the set of digits {0, 1, ..., 9} and R requires exactly nine digits. Additional constraints apply before the 2011 randomization change, and there may be dashes or spaces as separators (e.g., the ddd-dd-dddd format), which would change the alphabet and rules [17]. The specific data types depends on the specific use case, for instance, separators may be removed during preprocessing, and not considered part of the data type. For e-mail addresses, Σ includes letters, digits, special characters ('@' and '.'), and more, while R encodes the standard character placement constraints, minimum/maximum lengths and more, with multiple specification-level regular expressions for defining valid e-mail addresses [18]. In general, a data type can be more or less strict depending on the use case and the desired balance between utility and privacy (see Section 3.2). For example, for personal names, R may require that the string only contains letters and spaces, or it may require the names to correspond to known entities (as in synthetic data generation).

In contrast to unconstrained transformations (such as conventional encryption), format-preserving methods act on data types. We first introduce transformations acting on the data type itself, with no further restrictions:

$$T : D \to D.$$

We term such a function a *data type-preserving transformation*, as it maps every string of the data type to another string of the same data type. Using a lookup table to replace original values with pseudonyms is an example of such a transformation, if the pseudonyms are drawn from the same data type. As we focus on format-preserving encryption (FPE) as the de-identification method in this paper, we refer to inputs and outputs as plaintexts and ciphertexts, respectively. Many format-preserving methods are stricter than mere data type-preservation, in that they preserve the *format* of the specific plaintext—such as its length and character patterns—using

character-by-character replacement. Examples of such methods include FPE and character-by-character masking. We model this format-preserving behavior using a character-class mapping

$$g : \Sigma \to \mathcal{K},$$

with Σ the data type alphabet and $\mathcal{K}$ a set of character classes (e.g., letters, digits, special characters), each a subset of Σ. The specific mapping depends on both the data type and the desired level of format preservation, and can be more or less strict depending on the use case. For example, an e-mail address format definition may consider letters and digits as being in the same character class, whereas for postal addresses it may not. The format for a string $s = s_1 s_2 \dots s_n \in D$ is defined as the string of character classes obtained by applying g to each character:

$$\mathsf{Fmt}(s) = g(s_1)g(s_2)\dots g(s_n).$$

This induces an equivalence relation

$$x \sim y \iff \mathsf{Fmt}(x) = \mathsf{Fmt}(y),$$

partitioning the data type D into format classes

$$D = \bigcup_f D_f, \quad D_f = \{s \in D : \mathsf{Fmt}(s) = f\}.$$

A *format-preserving transformation* is then a function

$$\mathcal{T} : D \to D \quad \text{such that} \quad \forall s \in D : \ \mathcal{T}(s) \sim s,$$

meaning that the format class is the same for the plaintext and the ciphertext. How such a transformation outputs characters from the character classes defined by g for each character in the plaintext is not relevant for the format-based leakage, as any such choice will lead to the same format class for the ciphertext, and hence the same format-based information leakage.

This has also been formalized from an FPE perspective in [16], using the concept of slices of a domain (what we term data type), which are essentially the same as format classes. The main difference is that they focus on encryption-based permutations on these slices as the format-preserving transformation, whereas we consider any format-preserving transformation,

including non-cryptographic ones such as character-by-character masking. We do not go into any specific detail on how to construct computationally secure format-preserving transformations, as this is not the focus of the paper, but some approaches to this like cycle walking and rank-then-encipher are described in [16].

A typical example of a character-class mapping used in format-preserving de-identification methods separates letters and digits, while giving each format-defining special character its own class so that it is preserved (and an "Other" class to handle unexpected characters). For example, for e-mail addresses, we may have the following character classes:

$$\mathcal{K} = \{\text{Letter}, \text{Digit}, \text{Other}, @, .\},$$

with Letter $= \{a, \ldots, z, A, \ldots, Z\}$ (may include additional characters like accented letters) and Digit $= \{0, \ldots, 9\}$. This then gives the following character-class mapping:

$$g(s_i) = \begin{cases} \text{Letter} & \text{if } s_i \in \text{Letter} \\ \text{Digit} & \text{if } s_i \in \text{Digit} \\ s_i & \text{if } s_i \in \{@, .\} \\ \text{Other} & \text{otherwise} \end{cases}$$

A benefit of format preservation over just data type preservation is that it preserves the format-class distributions, which can be important for utility, as it ensures that the de-identified data matches the structural properties of the original data, such as distributions of separator characters, lengths and more. The primary disadvantage is then also the existence of these format classes, as these essentially encode information about the plaintext, which can lead to information leakage and potential re-identification as we detail in the following section. It is this format-based information leakage arising from format-preservation that we focus on in the paper.

3.2 Information-Theoretic Format Leakage

When considering the security of format-preserving de-identification methods, we are generally concerned with any potential information that can be learned about the plaintext from the ciphertext through the preserved format, potentially in combination with auxiliary information. Following

Kerckhoffs's Principle, we assume that the format classes are public, meaning that the character-class mapping defining them is known. Given that an adversary is assumed to know the format classes, the security of format-preserving methods is, from a cryptographic perspective, typically defined in terms of the indistinguishability of a ciphertext from a random permutation over its format class, and the adversary's ability to learn information about the plaintext from the ciphertext beyond the format itself. Several security notions have been defined for FPE [16], two of which we highlight here, as they can be reformulated to better reflect re-identification risks:

- **Message privacy (MP)**: An adversary cannot learn any information about the plaintext from the ciphertext other than the format.
- **Message recovery (MR)**: An adversary cannot recover the plaintext from the ciphertext.

These focus on the security of cryptographic schemes, and are not, by themselves, well-suited to reason about the re-identification risks of format-preserving de-identification methods. Instead, we can simply add the assumption of optimal auxiliary information to these notions. Formalizing these notions, we are concerned with how much information can be recovered by an adversary knowing the formats for a set of plaintexts $\mathcal{M}$ of data type D ($\mathcal{M} \subseteq D$), which comes from knowing the corresponding format-preserved ciphertexts $\mathcal{C}$. The adversary also has optimal auxiliary information, meaning that they have complete knowledge of the plaintext population and its distribution, but no record-level information linking a ciphertext to a specific plaintext. This is realistic for certain data types that are limited in size, and with public distributions (e.g. name registries). The adversary's goal is to recover as much information about the plaintexts as possible, given the formats and auxiliary information. Returning to the security notions, we can formally define the conditions for them to hold. Let $\mathcal{X}$ be the plaintext random variable, and $\mathcal{F}$ the format random variable, then if we let $\mathcal{M}_f = \mathcal{M} \cap D_f$ be the set of plaintexts in $\mathcal{M}$ with format f, then we have

- MP holds if and only if $|\mathcal{M}_f| > 1 \quad \forall f$, as otherwise there is only a single matching plaintext, and the distribution of plaintexts within each format class is uniform, meaning that

$$\forall f, \forall x \in \mathcal{M}_f : p(x \mid f) = \frac{1}{|\mathcal{M}_f|}.$$

- MR fails if and only if there exists a format class f such that $|\mathcal{M}_f| = 1$, meaning that only a single plaintext fits the format.

Whether these two security notions hold is primarily determined by the plaintext distribution within each format class, which is determined by the overall plaintext distribution and the format class definitions. However, even if MP holds, the candidate set of possible plaintext that can be matched to a ciphertext may still be very small, due to a small format class. Hence, we would ideally like the format to provide no information:

- **Format privacy (FP)**: Format does not provide any information about the plaintext.

This security notion does not necessarily imply the other two, as better than random guessing within the single format class may still be possible. We can measure the amount of information provided by format using Shannon entropy [19]. We are interested in the amount of information obtained about the plaintexts by observing the format, or, in other words, the mutual information of the two random variables for these. This information is given by

$$I(\mathcal{X}; \mathcal{F}) = H(\mathcal{X}) - H(\mathcal{X} \mid \mathcal{F}),$$

where $H(\mathcal{X})$ is the Shannon entropy of the plaintexts, and $H(\mathcal{X} \mid \mathcal{F})$ is the conditional Shannon entropy of the plaintexts given the format. The mutual information $I(\mathcal{X}; \mathcal{F})$ quantifies the reduction in uncertainty about the plaintexts $\mathcal{X}$ given the format $\mathcal{F}$, and thus serves as a measure of the information leakage from format preservation. The higher the mutual information, the more information about the plaintexts is revealed by the format. Conversely, a mutual information of zero would indicate that the format provides no information about the plaintexts, which is the ideal scenario for format privacy. The Shannon entropy of the plaintexts is then

$$H(\mathcal{X}) = -\sum_{x \in \mathcal{M}} p(x) \log_2 p(x),$$

with the conditional entropy of the plaintexts given the format

$$H(\mathcal{X} \mid \mathcal{F}) = -\sum_{x \in \mathcal{M}, f \in \mathcal{F}} p(x, f) \log_2 p(x \mid f).$$

If the distribution of $\mathcal{X}$ is uniform within each format class $\mathcal{M}_f$—the best case assumption given MP—the entropy becomes

$$H(\mathcal{X}) = -\sum_{x \in \mathcal{M}} \frac{1}{|\mathcal{M}|} \log_2 \frac{1}{|\mathcal{M}|} = \log_2 |\mathcal{M}|$$

and the conditional entropy

$$\begin{aligned} H(\mathcal{X} \mid \mathcal{F}) &= -\sum_{x \in \mathcal{M}, f \in F} \frac{\frac{|\mathcal{M}_f|}{|\mathcal{M}|}}{|\mathcal{M}_f|} \log_2 \frac{1}{|\mathcal{M}_f|} \\ &= -\sum_{x \in \mathcal{M}, f \in F} \frac{1}{|\mathcal{M}|} \log_2 \frac{1}{|\mathcal{M}_f|} \\ &= \sum_{x \in \mathcal{M}, f \in F} \frac{1}{|\mathcal{M}|} \log_2 |\mathcal{M}_f| \\ &= \sum_{f \in F} \sum_{x \in \mathcal{M}_f} \frac{1}{|\mathcal{M}|} \log_2 |\mathcal{M}_f| \\ &= \sum_{f \in F} \frac{|\mathcal{M}_f|}{|\mathcal{M}|} \log_2 |\mathcal{M}_f| \end{aligned}$$

Combining these gives the following formula for the format-based information leakage:

$$I(\mathcal{X}; \mathcal{F}) = \log_2 |\mathcal{M}| - \sum_{f \in \mathcal{F}} \frac{|\mathcal{M}_f|}{|\mathcal{M}|} \log_2 |\mathcal{M}_f|.$$

The leakage is then the expected log-reduction in candidate set size when moving from the set of plaintexts $\mathcal{M}$ to the format class M_f, and depends only on the format class partitioning of the plaintexts. The uniformity assumption makes this a lower bound on the leakage, as any non-uniformity within format classes would only increase the information revealed by the format. This follows because if the distribution of $\mathcal{X}$ is not uniform within a format class, then some plaintexts in that class are more likely than others, and thus observing the format would reveal even more information about the plaintext than in the uniform case.

We note that FP is achieved if $I(\mathcal{X}; \mathcal{F}) = 0$:

$$\log_2 |\mathcal{M}| - \sum_{f \in \mathcal{F}} \frac{|\mathcal{M}_f|}{|\mathcal{M}|} \log_2 |\mathcal{M}_f| = 0,$$

which rearranges to

$$\sum_{f \in \mathcal{F}} \frac{|\mathcal{M}_f|}{|\mathcal{M}|} (\log_2 |\mathcal{M}| - \log_2 |\mathcal{M}_f|) = 0,$$

which, as the summands are non-negative, can only be true if $\log_2 |\mathcal{M}| - \log_2 |\mathcal{M}_f| = 0$ for all f, which can only be true if $|\mathcal{M}_f| = |\mathcal{M}|$ for all f, which leads to the conclusion that there is a single format class. Hence, FP is only achievable if there is a single format class, meaning that the format-preserving equivalence relation is trivial (i.e. $\mathsf{Fmt}(x) = \mathsf{Fmt}(y)$ for all $x, y \in D$). In other words, any non-trivial format-preserving transformation will necessarily leak some information about the plaintexts through the preserved format.

It is important to note that this Shannon entropy calculation does not take into account the auxiliary information that an adversary is likely to have, as it assumes that the adversary has complete knowledge of the plaintext population and its distribution. In practice, the adversary may have much less information, and hence the actual information leakage may be much lower than this theoretical measure. However, it does provide a useful measure of the potential information leakage from format preservation, and can be used to compare different character-class mappings in terms of their potential for information leakage.

Lastly, these calculations are for the information leakage across the full dataset, and do not necessarily reflect the leakage for individual records. For example, if a record has a unique format, then the format-based information leakage for that record is maximal, as the format directly reveals the plaintext. On the other hand, if a record shares its format with many other records, then the format-based information leakage for that record is lower, as the format only narrows down the possible plaintexts to a larger set. Hence, the entropy calculations do not capture the potential for record-level re-identification for a subset of the dataset. Indeed, these calculations do not deal with the MP and MR security notions at all, and are more concerned with how close to the FP security notion a dataset is.

4 Misuse of Format-Preserving De-Identification

The formalizations in the previous section show that format-preserving de-identification methods have an inherent weakness when applied to variable-

format data types. In this section, we document that this problem is not merely theoretical, but is also a real-world issue, in that such methods are used in practice to create variable-format data types, by using chararacter-class mappings that induce multiple format classes, which, as explained in Section 3, can cause significant information leakage. Here, we show that format-preserving de-identification methods are used—or marketed—for data types in a way that creates multiple format classes, meaning that the format class is leaked. The motivation for this paper stems from observing that the risks associated with this should be obvious, yet are evidently insufficiently understood to stop this practice, or to prompt appropriate warnings.

One example is length-preserving redaction applied to credit card numbers (CCNs), which constitutes a variable-length data type ranging from 13 to 19 digits [20]. In common redaction practices, each digit is replaced by a masking symbol (e.g., '*'), thereby preserving the number of digits in the de-identified version [21]. This results in format classes that correspond directly to length-based equivalence classes: the length of the ciphertext reveals the length of the plaintext. This only creates a small number of format classes, meaning that the information leakage is small (assuming the length distribution is not too skewed). Concretely, certain card issuers use specific lengths for their CCNs, so preserving length reveals information about the card issuer, with American Express using length 15 and 16 [22] and Visa using lengths 13, 16, and 19 [23]. The leakage may be acceptable, as the format classes will still likely be large (unless some of the lengths are very uncommon in the data), but it is important to be aware of it and the potential risks it poses, especially when combined with auxiliary information. This example becomes more complicated when the data type also permits (but does not require) separator characters such as spaces or dashes, which may likewise be preserved in the de-identified version and thus leaking more information.

Credit-card numbers are also the typical example used for FPE, where they are often considered to consist of precisely 16 digits [24] [25], though the full length range is also considered in literature [16]. If FPE exhibits the same length-preserving property as the redaction method, then the same length-based leakage applies. We note that the NIST-approved FF1 [10] specification is length-preserving and in marketing materials from patent holder and licensor OpenText (previously Voltage Security/Micro Focus/HP) [26], its use for e-mail addresses, addresses and CCNs is promoted [27][28][29][30]. All of these are variable-length data types, and the examples provided in

these materials explicitly preserve length and special character positions in the output. Other data companies also depict FPE used in this manner [31] [32] [33]. Additionally, the FF1 specification itself does not warn of such uses. Even ENISA documentation presents examples in which FPE preserves length and special-character positions when encrypting e-mail addresses [34, p. 35]. However, the same document also cautions:

> It should be noted that, depending on the case, it might be needed to appropriately engineer FPE implementations, in order to avoid the emergence of patterns that may leak information on the individuals' identities. [34, p. 35]

This explicitly acknowledges that format-preserving transformation can reveal patterns in the original data, which can then lead to re-identification. Taken together, these examples show that format-preserving de-identification methods are used—or marketed to be used—in a way that preserves length and positions of format-defining characters, creating many format classes. While it is not clear how widespread these practices are, their existence alone is sufficient to motivate this paper, given the potential severity of the associated data-leakage risks. Furthermore, what is considered part of the "format" (i.e., the character-class mapping) is also not clear across different marketing materials, i.e. is there a distinction between letters and digits in e-mails and postal addresses? Clear cautionary guidance, such as that provided by ENISA, is generally absent. While authors of documentation and marketing material may assume that these problems are too obvious to require explicit warning, or that illustrative examples should not be interpreted literally, such problems should nevertheless be clearly stated to prevent less experienced practitioners from inadvertently falling into the pitfalls highlighted in this paper.

5 Leakage Example

In order to assess the potential severity of the format-based information leakage, we have obtained access to real customer data containing PII through collaborating organizations. Here, we can measure different properties, focusing on the distribution of format classes in the dataset, which allows us to calculate the information-theoretic leakage in bits as defined in Section 3.2. This data has several attributes to analyze for format-based leakage, which

we do for each attribute separately, without considering any potential interactions between attributes. This means that the results in this section form a lower bound on the record-level leakage, as the format classes for different attributes can be combined to create more specific format classes, which can lead to more information leakage. For example, even if a record has a non-unique format for both the name and the address, then the combination of these formats can be unique. This record-based leakage is more compelx to calculate, as it depends on the joint distribution of the attributes, and also requires auxiliary information for multiple attributes.

5.1 Acquiring and Processing Data

Two Danish financial organizations supported this work by executing our analysis scripts on a subset of their customer database and returning statistical data. We are highly appreciative of their assistance in making this research possible. The combined dataset contains 154 449 records. The attributes in focus were e-mail addresses, personal names, postal addresses, and city names. For each attribute, the data was first filtered to remove empty values and obvious errors, as real-world data is noisy and often contains misplaced or illegal entities. Some individuals may appear more than once in the merged dataset, since two independent datasets are merged, and a person may be represented in both datasets. However, this may also be the case for data from a single dataset, depending on its application, if explicit care has not been taken to prevent this. The analysis script produced the format for each string, using the following character-class mapping for city and personal names:

$$g(s_i) = \begin{cases} s_i & \text{if } s_i \in \{\text{Space}\} \\ \text{Other} & \text{otherwise} \end{cases}.$$

This makes the mapping quite loose, as it only preserves spaces, and treats all other characters as "Other", thereby preserving individual word lengths. For e-mail addresses the following character-class mapping was used:

$$g(s_i) = \begin{cases} s_i & \text{if } s_i \in \{@, .\} \\ \text{Other} & \text{otherwise} \end{cases},$$

as the '@' and '.' symbols are the most important format-defining characters for e-mail addresses, while all other characters are treated as "Other". For

postal addresses, the following character-class mapping was used, which is more strict than for names and cities, as it distinguishes between letters and digits, which are both important format-defining character classes for addresses, while only preserving spaces as special characters:

$$g(s_i) = \begin{cases} \text{Letter} \cup \text{Nordic Letter} & \text{if } s_i \in \text{Letter} \cup \text{Nordic Letter} \\ \text{Digit} & \text{if } s_i \in \text{Digit} \\ s_i & \text{if } s_i \in \{\text{Space}\} \\ \text{Other} & \text{otherwise} \end{cases}$$

with Letter = {a, ..., z, A, ..., Z} Nordic Letter = {æ, ø, å, ä, ö, Æ, Ø, Å, Ä, Ö} and Digit = $\{0, \ldots, 9\}$. As the data is from Danish companies, Nordic letters are part of the data, and they are treated as any other letters. After processing, the format and size of each format class for each attribute were shared with us for further processing. We are not attempting to re-identify individuals in the dataset, nor do we have access to the plaintexts or ciphertexts. We only have the format class and frequency pairs.

It is important to note that we have access to the specific format classes, which may be difficult to infer from ciphertexts alone, as there can be overlap between character classes, e.g., letters can be preserved but other character can still be mapped to letters. We use a specific "Other" class to signify non-specified characters, which can in theory be mapped to any character class, and thus make it difficult to infer the format classes from the ciphertexts. However, we assume that full character-class mapping is known, including all character classes and the characters mapping to them. Even if the precise mapping is not known, just knowing that individual word lengths are preserved is already a large leakage, as we show in the following section with the city and personal name attributes.

5.2 Results

When calculating the information leakage, we continued using the uniformity assumption, using the formula from Section 3.2:

$$H(\mathcal{X} \mid \mathcal{F}) = \sum_{f \in F} \frac{|\mathcal{M}_f|}{|\mathcal{M}|} \log_2 |\mathcal{M}_f|.$$

We let N, E, A, C be random variables for the four attributes (name, e-mail, address, city) and $\mathcal{F}_N, \mathcal{F}_E, \mathcal{F}_A, \mathcal{F}_C$ the corresponding format random vari-

ables. Given the size of the dataset we have the maximum possible entropy as

$$H_{\max} = \log_2(154\,449) \approx 17.24 \text{ bits.}$$

E-mail: Out of a total of 59 243 e-mail addresses, 3 165 different format classes were encountered, with 1 440 of size 1. This gives

$$H_{\max}(E) \approx 15.85 \text{ bits}, \quad H(E \mid \mathcal{F}_E) \approx 7.47 \text{ bits},$$

giving a mutual information of $I(E; \mathcal{F}_E) \approx 8.38$ bits. This means that format information makes guessing the correct e-mail address about $2^{8.38} \approx 333$ times easier.

Name: The datasets had 144 916 names (first name and last name, sometimes one or more middle names or initials) divided between 14 259 different format classes, with

$$H_{\max}(N) \approx 17.14 \text{ bits}, \quad H(N \mid \mathcal{F}_N) \approx 7.02 \text{ bits}.$$

This gives a mutual information of $I(N; \mathcal{F}_N) \approx 10.12$ bits, meaning that guessing the correct name is about $2^{10.12} \approx 1\,113$ times easier based on the format.

Address: The dataset included 144 803 addresses (street name, street number, and sometimes floor and door on floor info). These were distributed between 13 403 different format classes.

$$H_{\max}(A) \approx 17.14 \text{ bits}, \quad H(A \mid \mathcal{F}_A) \approx 7.78 \text{ bits}.$$

This gives a mutual information of $I(A; \mathcal{F}_A) \approx 9.36$ bits, meaning that guessing the correct address is about $2^{9.36} \approx 657$ times easier.

City: The dataset contained 83 469 city names as part of the postal code. As expected, the number of format classes was quite low due to the number of cities in Denmark, with only 166 different format classes.

$$H_{\max}(C) \approx 16.35 \text{ bits}, \quad H(C \mid \mathcal{F}_C) \approx 12.45 \text{ bits},$$

This gives a mutual information of $I(C; \mathcal{F}_C) \approx 3.9$ bits, meaning that guessing the correct city is about $2^{3.9} \approx 15$ times easier when knowing the format.

Type	Total	Different	Unique	Unique%	Leakage
E-mail	59 243	3 165	1 440	2.43%	8.38 bits
Name	144 916	14 259	9 052	6.25%	10.12 bits
Address	144 803	13 403	8 081	5.58%	9.36 bits
City	83 469	71	13	0.02%	3.9 bits

Table 1: For each of the attributes, the total number of values and the number of different and unique formats are counted and the mutual information of each attribute with its format class is calculated.

This is to be expected, due to a lower number of format classes. These results are summarized in Table 1.

Altogether, these results show that format preservation for these data types fails multiple of the security notions defined in Section 3.2. In particular, format privacy fails due to the observed single-attribute information leakage, which also constitutes a lower bound on the joint mutual information across attributes. Moreover, these results indicate that even message recovery—the weakest security notion—is violated: for at least 6.25% of the records the observed format uniquely identifies the corresponding plaintext value, enabling plaintext recovery using auxiliary information. Message privacy, however, may still hold, as we do not look at in-format class distributions, instead assuming uniformity.

In order to demonstrate the impact of the choice of character-class mapping we have tried another mapping for e-mail addresses and personal names respectively. For e-mail addresses we now preserve letters and digits, in addition to the '@' and '.' symbols, with all other characters treated as "Other":

$$g(s_i) = \begin{cases} \text{Letter} & \text{if } s_i \in \text{Letter} \\ \text{Digit} & \text{if } s_i \in \text{Digit} \\ s_i & \text{if } s_i \in \{@, .\} \\ \text{Other} & \text{otherwise} \end{cases}.$$

For personal names we now preserve letters specifically, with a separation between Nordic letters and other letters, with all other characters treated as "Other":

$$g(s_i) = \begin{cases} \text{Letter} & \text{if } s_i \in \text{Letter} \\ \text{Nordic Letter} & \text{if } s_i \in \text{Nordic Letter} \\ \text{Other} & \text{otherwise} \end{cases}$$

Re-doing the calculation with these mappings gives a leakage of 10.14 and 11.63 for e-mail addresses and personal names, respectively. This is higher than the previous mappings, showing that the choice of character-class mapping can have a significant impact on the information leakage, illustrating the importance of carefully considering the character-class mapping when using format-preserving de-identification methods.

6 Conclusion

In this paper, we formalized format preservation, and showed that preserving format inherently leaks information for variable-format data types. Building on this formalization, we quantified the information-theoretic leakage using mutual information with Shannon entropy. To demonstrate the practical relevance of this work, we identified the potential misuse of format-preserving methods for variable-format data types in documentation and marketing materials from several data vendors, as well as lack of use-case warnings for the NIST-approved FPE algorithm FF1. Such misuse can enable re-identification of individuals in de-identified datasets. In particular, this relies on the observation that certain de-identification methods, such as format-preserving encryption and redaction, in many configurations generate an output having the same format as the input, which for variable-format data can be very revealing.

To assess the severity of this leakage in practice, we analyzed 154 449 records of real personally identifiable information. Our analysis showed that at least 6.25%—likely more if considering multiple attributes—have a unique format, only based on a few preserved format characteristics—such as length and spaces in personal names—with the remaining records organized into small groups of candidates (format classes). This makes the de-identified data vulnerable to re-identification, due to small format classes allowing for an adversary to narrow down the candidate set significantly, and in some cases down to a single record. In total, the formats for each attribute significantly decrease the set of possible matching candidates on average, from a factor of 15 (city names) to a factor of 1 113 (personal names). Of course, the actual re-identification risk depends on the availability and quality of auxiliary information (whether a larger external dataset or singular individual information), but the format-based attack surface is significant and should not be underestimated. Looking at record-level formats and more datasets

with different attributes is an important direction for future work, as this can give a better picture of the re-identification risks.

In order to prevent this leakage, the format-based linkage between plaintext and ciphertext needs to be broken, since we showed that format-based information leakage can only be zero when just a single format class exist, or in other words, the format-preserving transformation needs to only preserve datatype, and not any more specific subsets of the data type. At the same time, one must keep in mind that there is a reason why format-preserving methods have been used: protected data must often retain the same behavioral and syntactic properties as the original. The key insight is therefore not that format-preservation should never be used, but that it should be applied only after proper consideration of the format classes and their potential for information leakage, and that it should be applied in a way that minimizes this leakage. This can be done by using the loosest possible character-class mapping, which creates the largest format classes and thus minimizes the information leakage. The impact of less strict character-class mappings can be seen in Section 5, where separating letters and digits for e-mail addresses and Nordic and non-Nordic letters for personal names in the character-class mapping significantly increases the information leakage, from 8.38 bits to 10.14 and 10.12 bits to 11.63 respectively.

If possible, fixed-length formats like the two-digit date format and zero-padded decimal notation should be used, as this completely eliminates the leakage problem presented in this paper. If these workarounds are not feasible (e.g., due to legacy system constraints), then it is still important to be aware of the re-identification risks, and not see the de-identified data as fully protected. Format-preserving de-identification should be seen as one step in a comprehensive de-identification and data protection strategy to make data safe to use, rather than a complete solution.

Disclosure of Interests: Martin Staal Boesgaard is Founder and CEO of the Danish company PII Guard, which develops and sells software and services specialized in data de-identification. Markus Juhl Larsen is an employee of PII Guard. Martin Staal Boesgaard is also the inventor of Type-Preserving Encryption, which is the next generation of FPE and has been designed to mitigate the problems described in this paper by preserving data type over format.

References

[1] Federal Trade Commission. Ftc takes action against marriott and starwood over multiple data breaches, October 2024. Last accessed March 24, 2026.

[2] Equifax Inc. Equifax releases details on cybersecurity incident, announces personnel changes. `https://investor.equifax.com/news-events/press-releases/detail/237/equifax-releases-details-on-cybersecurity-incident`, September 2017. Last accessed January 30, 2026.

[3] Pieter Arntz. Ransomware gang claims conduent breach: What you should watch for next, October 2025. Last accessed January 30, 2026.

[4] IBM Security. National public data breach publishes private data of 2.9b us citizens. No explicit publication date provided on the page. Last accessed February 20, 2026.

[5] Yahoo Security. Yahoo data breach affects 3 billion accounts, 2017. Last accessed February 20, 2026.

[6] Verizon. Verizon 2023 data breach investigation report, 2023. Last accessed April 30, 2024.

[7] Paul Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA l. Rev.*, 57:1701, 2009.

[8] Latanya Sweeney. Weaving technology and policy together to maintain confidentiality. *The Journal of Law, Medicine & Ethics*, 25(2-3):98–110, 1997.

[9] Horst Feistel. Cryptography and computer privacy. *Scientific american*, 228(5):15–23, 1973.

[10] Morris Dworkin and Niels Mouha. Recommendation for block cipher modes of operation: Methods for format-preserving encryption, second public draft. NIST Special Publication 800-38Gr1 2pd, National Institute of Standards and Technology, February 2025. Second Public Draft.

[11] Arvind Narayanan and Vitaly Shmatikov. How to break anonymity of the netflix prize dataset. *arXiv preprint cs/0610105*, 2006.

[12] Michael Barbaro and Tom Jr. Zeller. A face is exposed for aol searcher no. 4417749. *The New York Times*, August 2006. Last accessed March 17, 2026.

[13] Zaruhi Aslanyan and Martin Staal Boesgaard. Privacy analysis of format-preserving data-masking techniques. In *2019 12th CMI Conference on Cybersecurity and Privacy (CMI)*, pages 1–6. IEEE, 2019.

[14] Samir Hodžić, Andreas Brasen Kidmose, Brooke Elizabeth Kidmose, Lars Ramkilde Knudsen, and Weizhi Meng. Pitfalls of data masking techniques: Re-identification attacks. In *International Conference on Security and Privacy in Communication Systems*, pages 406–428. Springer, 2024.

[15] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, 10(05):557–570, 2002.

[16] Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In *International workshop on selected areas in cryptography*, pages 295–312. Springer, 2009.

[17] Microsoft Corporation. U.s. social security number (ssn) entity definition. `https://learn.microsoft.com/en-us/purview/sit-defn-us-social-security-number`, August 2024. Accessed: 2026-03-13.

[18] Html living standard — email state (type=email). `https://html.spec.whatwg.org/multipage/input.html#valid-e-mail-address`, 2026. Defines a commonly used regular expression for email address validation.

[19] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[20] ISO/IEC 7812-1:2017, Identification cards — Identification of issuers — Part 1: Numbering system, 2017.

[21] IBM Global Data Privacy. Ibm knowledge center: Redacting data example, 2025. Last accessed August 21, 2025.

[22] American Express. What is a credit card number?, January 2025. Last accessed March 13, 2026.

[23] Visa. Card input pattern: Card number field. Last accessed March 13, 2026.

[24] Mor Weiss, Boris Rozenberg, and Muhammad Barham. Practical solutions for format-preserving encryption. *arXiv:1506.04113*, 2015.

[25] Shikha Gupta, Satbir Jain, and Mohit Agarwal. Ensuring data security in databases using format preserving encryption. In *2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 1–5. IEEE, 2018.

[26] Voltage Security. Ffx mode of operation for format-preserving encryption, revised letter of assurance for essential patent claims, 2025. Last accessed August 21, 2025.

[27] Voltage Security. Voltage security: Securedata enterprise, data sheet, 2025. Last accessed August 21, 2025.

[28] Voltage Security. Voltage securedata, presentation, 2025. Last accessed August 21, 2025.

[29] OpenText. Opentext: Tokens and tokenization, solution overview, 2025. Last accessed August 21, 2025.

[30] OpenText. Opentext data privacy and protection foundation, product overview, 2025. Last accessed August 21, 2025.

[31] SnapLogic. Snaplogic blog: Securing personally identifiable information using format preserving encryption, 2025. Last accessed August 21, 2025.

[32] Cosmian. Cosmian: Format-preserving encryption, documentation, 2025. Last accessed August 21, 2025.

[33] IRI. Format-preserving encryption (fpe) - encrypt data without changing its appearance, 2025. Last accessed August 29, 2025.

[34] Meiko Jensen, Cedric Lauradoux, and Konstantinos Limniotis. Pseudonymisation techniques and best practices. *Technical Report*, page 35, 2019.