Journal of Universal Computer Science, vol. 30, no. 9 (2024), 1224-1243
submitted: 27/9/2023, accepted: 30/5/2024, appeared: 14/9/2024 CC BY-ND 4.0

Bloom Encodings in DGA Detection: Improving Machine Learning Privacy by Building on Privacy-Preserving Record Linkage

Lasse Nitz
(Fraunhofer FIT, Aachen, Germany
RWTH Aachen University, Aachen, Germany
https://orcid.org/0000-0002-3131-7444, lasse.nitz@fit.fraunhofer.de)

Avikarsha Mandal
(Fraunhofer FIT, Aachen, Germany
https://orcid.org/0000-0002-8641-7207, avikarsha.mandal@fit.fraunhofer.de)

Abstract: The use of machine learning has shown to benefit a wide range of applications, especially for classification tasks. As such, the detection of algorithmically generated domains to identify corrupted machines has proven itself to be a mature use case with good classification performance. The use of privacy and security sensitive data, however, raises concerns in scenarios that require interaction with external parties. As one of such scenarios, we consider the training of domain generation algorithm detection classifiers in a Machine-Learning-as-a-Service (MLaaS) scenario. We evaluate the use of a Bloom encoding approach from the area of privacy-preserving record linkage to prevent the MLaaS provider from getting to know the exact classification task as well as the data samples transmitted for training and classification. We investigate the threat associated with pattern mining attacks by performing a privacy analysis for two versions of these encodings (basic and randomized). We further identify sets of parameter values which we find to provide an adequate level of protection against these attacks. We see the potential for this approach in machine learning use cases dealing with sensitive data or tasks, especially for MLaaS scenarios dealing with short data samples that lack a clear structure.

Keywords: Security, Privacy, DGA Detection, Machine Learning, Bloom Encoding, Threat Detection, Data Obfuscation, Machine-Learning-as-a-Service, Malware
Categories: C.2.0, E.4, I.5.2, K.6.5
DOI: 10.3897/jucs.134762

1 Introduction

As in many other application areas, machine learning has become a popular approach for solving classification tasks in the cybersecurity domain, often involving the use of sensitive data. This, however, raises concerns when considering collaborative settings, such as federated learning, Machine-Learning-as-a-Service (MLaaS) or the collaborative creation of diverse data sets. Domain generation algorithm (DGA) detection is one example of a cybersecurity use case dealing with sensitive information. A malicious application of DGAs lies in enabling communication between a bot herder (i.e., the controller of a botnet) and bots (i.e., the corrupted machines) in a botnet via a command-and-control server. Instead of hard-coding the domain of a command-and-control server into the malware, new suitable domains are frequently generated via the DGA by both

the bots and the bot herder locally. The properties of the DGA enable a connection between a bot and a command-and-control server due to a likely overlap of domains generated by bots and domains generated by the bot herder. For the bot herder, this has the benefit that blocklisting domains used for command-and-control servers in the past does not affect future communication, ensuring that control over the botnet is preserved. In order to detect potential bots in a monitored network, the use of machine learning has been proposed to identify algorithmically generated domains (AGDs), viewing them as indicators of compromise (see, e.g., [Drichel et al., 2020]). As the classifiers need to distinguish AGDs from non-AGDs, the training data sets also have to include benign samples, such as non-existent (NX) domain requests. These NX domains, however, pose both a privacy and security risk. NX domains collected in real-world networks can reveal information about browsing behavior, which may conflict with legal obligations such as customer contracts, or with company guidelines. Further, the leakage of NX domains can pose a threat to security, since respective requests can reveal misconfigured devices in a network, which try to connect to web-services that are no longer available. This may be abused by malicious actors, who register such a domain to mimic the respective web-service. NX domains can further provide information about specific benign applications using AGDs (such as endpoint security software) in the network.

The sanitization of samples in the DGA detection use case represents a class of problems that poses special requirements on sanitization approaches. Specifically challenging is that AGDs and NX domains are relatively short strings that do not provide a clear distinction between sensitive and non-sensitive parts, ruling out redaction-based sanitization approaches. The overall lack of structure beyond a separation into different domain elements via dots (the number of which can vary) further rules out classical anonymization approaches as they are commonly used for relational data. The benefit of using data sanitization over approaches such as secure multi-party computation (see [Drichel et al., 2021] for an application in DGA detection) is that they are applied to data rather than algorithms. While this commonly leads to weaker formal guarantees in regard to the protective qualities of the approach, data sanitization approaches are often more efficient than cryptographic approaches and provide the data owner with more flexibility, since they can be applied locally without requiring other parties to take part in a specific privacy-preserving protocol.

To protect these samples in different collaborative settings, [Nitz and Mandal, 2023] have proposed the use of Bloom encodings as a sanitization measure, utilizing an encoding approach from the area of privacy-preserving record linkage (PPRL) based on Bloom filters. While this work has demonstrated the feasibility of performing machine learning tasks on these encodings, respective data leakage risks were primarily considered on a conceptual level. With this work, we aim to examine this risk on a more technical level.

Our contribution: We perform the first technical analysis of the data leakage risk associated with using Bloom encodings in a MLaaS setting. Our investigation focuses on the DGA detection use case and consists of two parts. In Section 5 we analyze the impact of the randomization procedure proposed in [Schnell and Borgs, 2016] on pattern preservation in the encodings. In Section 6 we apply the pattern mining attack proposed in [Christen et al., 2018] and evaluate the quality of its intermediate results as well as the re-identified cleartext candidates. In Section 7 we provide suggestions for choosing parameter values when applying the Bloom encoding approach in MLaaS settings based on our findings. Our analysis concludes that the parameter values used in [Nitz and Mandal, 2023] prevent reliable cleartext re-identification through this attack in the considered setting.

The remainder of this work is structured as follows. In Section 2 we briefly summarize related work and introduce definitions used throughout this work. In Section 3 we discuss the key properties of the Bloom encoding approach when applied in a general MLaaS setting. We then describe the scope and threat model of our analysis in Section 4. In Section 5 we perform the analysis of bit pattern preservation under special consideration of randomized Bloom encodings. We complement this analysis by evaluating the practical application of pattern mining attacks in Section 6. Section 7 provides suggestions for choosing encoding parameters based on the result of the previous two sections. Lastly, we discuss our findings and provide directions for future work in Section 8.

2 Background

The sanitization of training data for DGA detection poses some challenges that set it apart from more traditional data such as relational or graph data. The lack of clear structure beyond a required domain and top-level domain does not allow to directly distinguish between sensitive and non-sensitive parts of a sample. Related work on protecting respective samples has hence taken other approaches. In [Drichel et al., 2021] different frameworks implementing cryptographic protocols have been evaluated to protect samples for DGA detection in a Classification-as-a-Service setting. While the utility was only affected marginally through model simplifications, the communication overhead introduced by the protocols has been considered to be prohibitive for real-world setting. In [Holmes et al., 2021] the general reversibility of FANCI feature vectors [Schüppen et al., 2018] for the DGA detection use case has been considered. While the quality of reconstructed samples was of poor quality, the analysis only considered general reversibility via machine learning, without accounting for potentially available background knowledge or other forms of data leakage. In [Nitz and Mandal, 2023] the use of a Bloom filter based encoding approach from the area of PPRL has been discussed for machine-learning based DGA detection. The protective properties of this approach, however, have only been discussed on a conceptual level. In this work, we examine the protective properties of this Bloom encoding approach more thoroughly.

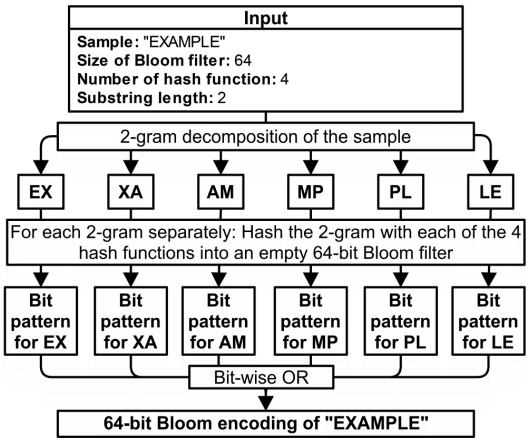


Figure 1: Visualized example of the flow for creating basic Bloom encodings, adapted from [Nitz and Mandal, 2023].

2.1 Bloom Encodings

Similar to [Nitz and Mandal, 2023], we focus on two variants of Bloom encodings: The basic encodings initially proposed in [Schnell et al., 2009] and encodings hardened via randomization [Schnell and Borgs, 2016].

Basic: Formally, the basic Bloom encoding approach can be defined as follows: Let a cleartext record P be given, as well as values for the Bloom filter size l, a set of k many independent cryptographic hash functions $H = \{h_1, ..., h_k\}$, and the substring length q. Then the basic Bloom encoding B of P is the bit string of length l, in which $B[i] = 1$ with bit position $i \in \{1, ..., l\}$, if, and only if, there exists a hash function $h \in H$ and a substring Q of P of length q such that $i = (h(Q) \mod l) + 1$. This definition is extended to data sets by encoding each record of a data set into a separate Bloom filter. The values of the parameters l, k and q as well as the set of hash functions H remain consistent across different records. Figure 1 provides a visualization of the flow for creating basic Bloom encodings for a given cleartext.

Randomized: Motivated by decoding attacks on basic Bloom encodings, the use of RAPPOR's permanent randomized response step [Erlingsson et al., 2014] has been proposed as a hardening approach for Bloom encodings [Schnell and Borgs, 2016]. A randomized encoding E is derived from a basic encoding B by applying the following procedure for a given $f \in [0, 1]$ to its individual bit positions $i \in \{1, ..., l\}$:

$$E[i] = \begin{cases} 0, & \text{with probability } 0.5 \cdot f \\ 1, & \text{with probability } 0.5 \cdot f \\ B[i], & \text{with probability } 1 - f \end{cases} \quad (1)$$

We refer to a Bloom encoding as *sparse*, if it contains more 0-bits than 1-bits, and we refer to it as *dense* in the vice versa case. We further say that a bit pattern $\mathcal{I} \subseteq \{1, ..., l\}$ defining a set of bit positions containing 1-bits *matches* a Bloom encoding B, if $B[i] = 1$ for all $i \in \mathcal{I}$. We use this terminology for both basic and randomized encodings. For the basic Bloom encoding of a single q-gram, we refer to the set of bit positions containing 1-bits as the *q-gram pattern* of this q-gram. For convenience, we say that a q-gram matches a Bloom encoding, if its q-gram pattern matches this encoding. Note that the basic Bloom encoding of a cleartext is the bitwise OR of the individual basic Bloom encodings of all of its q-grams. This also means that the bit pattern of a cleartext is the union of the individual q-gram patterns of all of its q-grams. When we refer to q-gram frequencies or the number of q-gram occurrences, we do not count duplicate occurrences within the same cleartext.

2.2 Bloom Encodings in DGA Detection

While the Bloom encoding approach was originally proposed to link records of different data holders without revealing sensitive information, it provides properties which make it suitable for machine learning. Specifically, it preserves similarity of samples through its substring decomposition step and provides encodings of uniform length. Since similarity is preserved on a structural level rather than a semantic one (in contrast to, e.g., feature extraction), it can also be applied without requiring to first understand which properties of a sample are relevant for a specific classification task. In [Nitz and Mandal, 2023] binary DGA detection classifiers have been trained using $l \in \{64, 128\}$, $k = 2$, $q = 2$, and $f \in \{0.0, 0.4\}$, where $f = 0.0$ implies that no randomization is used. For evaluating utility of the encodings, the NYU model [Yu et al., 2018] has been used due to its good

performance in the featureless binary DGA detection task (malicious/benign) [Drichel et al., 2020]. The data set used for this evaluation consisted of 134 036 samples with a 50/50 label distribution (malicious/benign in regard to AGDs), which was split into a 101 866 sample training set, a 5 362 sample holdout set, and a 26 808 sample evaluation set with a 50/50 label distribution each. The benign samples were NX domains collected in a university environment, and the malicious samples originated from DGArchive [Plohmann et al., 2016]. The results are depicted in Figure 2. It can be seen that both shorter Bloom filters and the use of randomization have a negative impact on accuracy and recall. Surprisingly, the precision held up well except for the case of using both shorter Bloom filters and randomization ($l = 64$, $f = 0.4$).

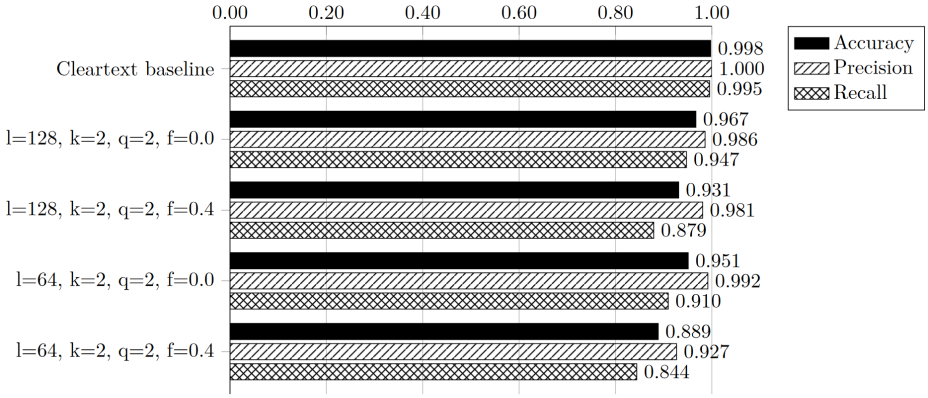


Figure 2: Utility evaluation results for binary DGA detection on Bloom encodings with cleartext results for reference, adapted from [Nitz and Mandal, 2023].

3 Bloom Encodings in Machine-Learning-as-a-Service

As discussed in [Nitz and Mandal, 2023], using Bloom encodings in a MLaaS setting with an honest-but-curious service provider does not require to share any parameter values with this service provider. It further has the benefit that the MLaaS provider only gets to see the encodings, which keeps the actual underlying classification task (such as DGA detection) hidden. Knowledge of the encoding parameters further acts as an access control mechanism: Because the model has been trained on encoded data only, it will only provide suitable classification results for encodings generated via the same parameter values. The data owner can hence ensure that neither the MLaaS provider nor any other unintended party can use the trained classified without first gaining access to the encoding parameters.

Recent work in PPRL such as [Armknecht et al., 2023] has considered two primary threats to using Bloom encodings in settings where the adversary tries to decode them without having access to the parameter values: Graph matching attacks [Vidanage et al., 2020] and pattern mining attacks [Christen et al., 2018][Vidanage et al., 2019][Christen et al., 2019]. However, since graph matching attacks require a significant overlap between the cleartexts behind the targeted encoding and a background knowledge data set available to the adversary [Armknecht et al., 2023][Vidanage et al., 2020], we consider the threat

of pattern mining attacks to be the greater risk in the MLaaS setting. For a pattern mining attack, the adversary aims to find frequent co-occurring bit patterns in the encodings and link those patterns to q-grams in a background knowledge data set. This background knowledge needs to have comparable q-gram frequencies to the cleartext data set behind the targeted encodings, and is ideally taken from a comparable domain. In contrast to graph matching attacks, it does not require a direct overlap with the targeted data set.

4 Scope of the Analysis and Threat Model

We split our analysis into two parts: The analysis of the impact of randomization on preserving bit patterns (compared to basic Bloom encodings) in Section 5 and the practical evaluation of pattern mining attacks in Section 6. The analysis of pattern preservation primarily investigates properties that could be exploited in attacks based on pattern mining. As the results of this analysis, however, remain abstract (e.g., by quantifying certain biases or correlations), the question of what these values mean for practical attacks arises. We thus also evaluate the success of pattern mining attacks. In the following, we briefly summarize the scope of our privacy analysis and the threat model assumed for the pattern mining attacks. For both parts, we utilize DGA detection data sets.

Analysis of pattern preservation: We first analyze the impact of the randomization procedure as given by Equation 1 on the preservation of patterns present in the basic Bloom encodings. For this, we look at bit patterns on different levels:

1. We begin by examining the impact on the bias exhibited by individual bit positions. Specifically, we consider this bias to be determined by how much more (or less) likely it is that a certain bit position is set compared to a uniform distribution (i.e., a 0.5 chance).

2. Secondly, we investigate the likelihood that complete bit patterns are preserved during randomization. Since especially q-gram patterns are of interest for pattern mining, we focus on these patterns, but our analysis can be analogously applied to the preservation of other substring patterns as well. We also examine how the introduction of false bit patterns to an encoding relates to the preservation of q-gram patterns by considering the correlation between the number of instances in which a q-gram pattern occurs in encodings and the number of instances in which the q-gram is part of respective cleartexts. This is motivated by the observation that this correlation is used in pattern mining attacks to associate frequently occurring bit patterns to frequently occurring q-grams in a background knowledge data set.

3. Thirdly, we consider the preservation of partial patterns. This is motivated by the idea that decoding attempts may not need to re-identify full q-gram patterns, if these patterns contain unique subpatterns. For this, we analyse bit value co-occurrence (meaning patterns of length 2) and consider how much more (or less) likely two bits are to co-occur in an encoding compared to a uniform bit distribution, as well as the impact that randomization has on these biases.

We use a $2\,000$ sample DGA data set to also practically evaluate these biases and correlations.

Evaluation of pattern mining attacks: For the practical evaluation of pattern mining attacks, we focus on the use of Bloom encodings in the MLaaS scenario. As such, the data owner intends to use a MLaaS solution by an honest-but-curious provider (the adversary)

to train and afterwards use a DGA detection classifier. To prevent the MLaaS provider from learning the samples transmitted for training and classification, the data owner encodes the samples as Bloom encodings. The adversary is assumed to be aware that the Bloom encoding approach has been used. As motivated in Section 3, we focus on pattern mining attacks as the attack method. We further assume that the adversary has knowledge of the q-gram length (since it is commonly $q = 2$) and access to a background knowledge data set taken from a similar domain as the data set behind the encodings. As the encodings can hide the actual classification task from the MLaaS provider, the latter assumption may not necessarily be realistic. But it does also account for settings in which the adversary can find out the classification task, e.g., by relating the classification timeline to events in the real world. The goal of the attack is to reconstruct as many cleartexts as possible from the encodings by re-identifying individual domain elements. We evaluate the success of the attack by the quantity and quality of re-identified q-gram patterns, the quality of the assignment of re-identified patterns to encodings (both matching and non-matching), and the quality of the re-identified cleartext candidates (meaning domain elements) per encoding.

5 Analysis of Pattern Preservation

In the following, we perform a privacy analysis of the Bloom encoding approach hardened by the randomized response step of the RAPPOR mechanism as given by Equation 1. We start by analyzing the impact of the randomization procedure on the probability of individual bit positions being set. Afterwards, we discuss the effect of the randomization procedure on the preservation of complete q-gram patterns and the introduction of false patterns. Lastly, we discuss the risk of partial patterns and the impact the randomization procedure has on bit value co-occurrence in encoded data sets. Throughout our analysis, we practically evaluate our findings on a 2 000 record subset of the DGA data set that was used for the utility evaluation in [Nitz and Mandal, 2023] as referred to in Section 2.2. This 2 000 record subset, which in the following is referred to as the DGA-2000 data set, was obtained by randomly sampling 1 000 malicious (AGDs) and 1 000 benign (NX domains) records from the data set used for the utility evaluation.

5.1 Individual Bit Positions

We begin by considering the impact of the randomization procedure on the distribution of values per bit position. For this, we focus on the probability that a bit is set in an encoding for a given data set and given encoding parameters, and compare this probability between the basic Bloom encoding and the randomized Bloom encoding. Because pattern mining attacks aim to exploit the circumstance that certain q-grams are more likely to occur than others, matching more likely q-grams to bit positions which are more likely to be set does provide an initial starting point for a pattern mining attack. Similarly to [Armknecht et al., 2023], we consider the bit position bias as the distance from the uniform value distribution of a bit position, meaning that we define it for a bit position i in a basic Bloom encoding B and the randomized Bloom encoding E, respectively, as follows

$$\epsilon_i^B \coloneqq \Pr\left(B[i] = 0\right) - \frac{1}{2} \tag{2}$$

$$\epsilon_i^R \coloneqq \Pr\left(E[i] = 0\right) - \frac{1}{2} \tag{3}$$

Per definition of the randomization procedure, a fraction f of bits is expected to change values to an even distribution, since a fraction $0.5f$ of bits is expected to be set, and a fraction $0.5f$ of bits is expected to be unset, independent of the respective values before randomization. The remaining fraction $1 - f$ of bits will not change in value, and is thus expected to maintain the bias ϵ_i^B. Following this argument, the relationship between ϵ_i^B and ϵ_i^R is expressed by

$$\begin{aligned} \epsilon_i^R &= (1-f) \cdot \epsilon_i^B + f \cdot 0 \\ &= (1-f) \cdot \epsilon_i^B \end{aligned} \tag{4}$$

and hence decreases linearly at rate $1 - f$ with growing values of f. If we consider the bias of an encoded data set as the maximum bias of its bit positions, also the bias of the data set decreases at this rate, meaning that $\epsilon^R = (1-f) \cdot \epsilon^B$. Note that this relationship is independent of the chosen data set, but that the value of ϵ_i^B is dependent on the data set and the encoding parameters.

5.2 Complete Patterns

Next, we consider the probability of preserving full q-gram patterns during randomization as well as the probability of introducing false pattern. We then discuss how these two cases relate to each other in the context of pattern mining attacks.

5.2.1 Preservation of Complete Patterns

To analyze the degree to which patterns are preserved during the randomization process, we consider a basic Bloom encoding B, which encodes a q-gram Q (potentially among others). We denote the set of bit positions set by Q as $\mathcal{I}_Q \subseteq \{1, ..., l\}$, meaning $i \in \mathcal{I}_Q$ if, and only if, there exists a hash function $h \in H$ such that $i = (h(Q) \mod l) + 1$. We further denote the event of B exhibiting the pattern $\mathcal{I}_Q$ as $\mathbb{E}_{\mathcal{I}_Q}$. Since the number of 1-bits of a q-gram pattern is restricted by the number k of hash functions, it holds that $1 \leq |\mathcal{I}_Q| \leq k$.

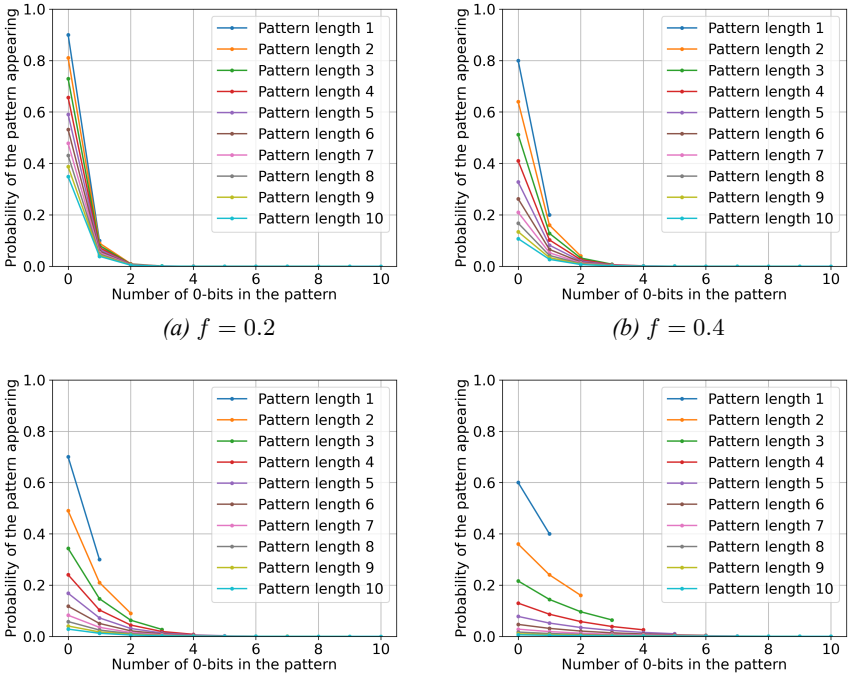
The probability that the pattern $\mathcal{I}_Q$ is preserved during randomization, meaning that for all $i \in \mathcal{I}_Q$ it holds that $E[i] = 1$, is

$$\Pr\left(E[i] = 1 \text{ for all } i \in \mathcal{I}_Q \mid \mathbb{E}_{\mathcal{I}_Q}\right) = \left(1 - \frac{1}{2} \cdot f\right)^{|\mathcal{I}_Q|} \tag{5}$$

5.2.2 Introduction of False Complete Patterns

The randomization procedure, however, cannot just damage patterns of q-grams contained in the respective cleartext, but also introduce patterns to the encoding which represent q-grams that are not part of the cleartext. While this can already happen within the basic Bloom encoding due to collisions, the randomization procedure can introduce false patterns in a non-deterministic fashion.

In the following, we assume that we have some basic Bloom encoding B, and a q-gram Q with pattern $\mathcal{I}_Q$, which is not encoded in B. Specifically, we consider the case



(a) $f = 0.2$ *(b)* $f = 0.4$

(c) $f = 0.6$ *(d)* $f = 0.8$

Figure 3: The probability of falsely introducing bit patterns of varying length during randomization for different values of f independently of any data set.

that there exists at least one $i \in \mathcal{I}_Q$, such that $B[i] = 0$. We denote this event by $\neg\mathbb{E}_{\mathcal{I}_Q}$. Then the probability that $\mathcal{I}_Q$ is falsely introduced to E is

$$\Pr\left(E[i] = 1 \text{ for all } i \in \mathcal{I}_Q \mid \neg\mathbb{E}_{\mathcal{I}_Q}\right) = \left(1 - \frac{1}{2} \cdot f\right)^{|\mathcal{I}_{B=1} \cap \mathcal{I}_Q|} \cdot \left(\frac{1}{2} \cdot f\right)^{|\mathcal{I}_{B=0} \cap \mathcal{I}_Q|} \tag{6}$$

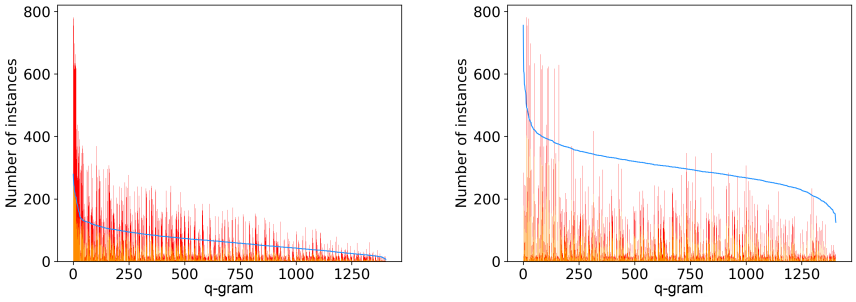
where $\mathcal{I}_{B=1}$ and $\mathcal{I}_{B=0}$ denote the sets of bit positions of B containing 1-bits and 0-bits, respectively. The probability hence depends on how many bits of the pattern are already set. Specifically, it is less likely that the pattern is introduced in the randomized Bloom encoding when fewer bits of the pattern are set in the basic Bloom encoding (unless $f = 1.0$). This relationship is visualized in Figure 3.

5.2.3 Relationship of Pattern Preservation and Pattern Introduction

Figure 3 and Table 1 show that unless the randomization parameter f is chosen close to 1, it cannot be expected that a notable number of false patterns is generated if the pattern differs from the basic Bloom encoding in more than two bits. This indicates that forcing the generation of false patterns during randomization requires the basic Bloom encodings to be dense, and/or a small number of hash functions (i.e., $k \leq 3$) to ensure less distinct patterns.

Encoding parameters	Noise	Pearson corr.	Spearman corr.
$l = 64,\ k = 2, q = 2$	$f = 0.0$	0.1603	0.0901
$l = 64,\ k = 2, q = 2$	$f = 0.4$	0.1213	0.0813
$l = 128, k = 2, q = 2$	$f = 0.0$	0.2707	0.1678
$l = 128, k = 2, q = 2$	$f = 0.4$	0.2194	0.1431
$l = 150, k = 3, q = 2$	$f = 0.0$	0.3373	0.2115
$l = 150, k = 3, q = 2$	$f = 0.4$	0.2912	0.1881
$l = 500, k = 5, q = 2$	$f = 0.0$	0.7726	0.5544
$l = 500, k = 5, q = 2$	$f = 0.4$	0.7090	0.4937

Table 1: The sample Pearson correlation coefficient and Spearman's rank correlation coefficient between the number of cleartext occurrences of q-grams and the number of respective pattern occurrences in different encoded versions of the DGA-2000 data set.



(a) $l = 500, k = 5, q = 2, f = 0.4$ *(b) $l = 150, k = 3, q = 2, f = 0.4$*

Figure 4: Visualization of the number of cleartext records containing specific q-grams (red bars incl. orange part), number of instances in which the q-gram patterns were preserved during randomization (orange bar) and total number of occurrences of the q-gram patterns in the randomized Bloom encodings (blue line). Each value on the x-axis represents a q-gram, and the y-axis shows the total number of respective instances in the DGA-2000 data set. The white area below the blue line represents falsely introduced q-gram patterns.

We have experimentally validated this observation by encoding the same data set with differing Bloom filter sizes and differing numbers of hash functions, but the same value for the randomization parameter. The result is shown in Figure 4 and indicates that the introduction of false patterns is crucial to impede pattern mining attacks, as the destruction of patterns on its own is not sufficient to break the correlation between the frequency of a q-gram in the cleartext and the observed frequency of its pattern in the randomized encoding.

We suspect that this is the case, because most q-grams will have a pattern size of k, resulting in exactly the same probability of a pattern being destructed during randomization for the large majority of q-grams. As previous publications (e.g., [Christen et al., 2018][Vidanage et al., 2019]) have pointed out, patterns shorter than k are relatively unlikely to occur. Because the probability of bit collisions during pattern generation is a

birthday problem, the probability of hash functions colliding during pattern generation is given as

$$1 - \frac{\frac{l!}{(l-k)!}}{l^k} \tag{7}$$

As such, the probability of having a collision in a pattern with $l = 150$ and $k = 3$ is less than 0.02 (the same also holds for $l = 500$ and $k = 5$), and for $l = 500$ and $k = 10$ the probability is still less than 0.087. We thus conclude that relying solely on pattern destruction is insufficient to break the correlation between the number of cleartexts in which a q-gram appears and the number of randomized encodings that exhibit this q-gram's pattern when considering these parameter ranges.

5.3 Partial Patterns

Even though the first part of our analysis indicates that the correlation between q-gram frequency and q-gram pattern frequency in the randomized encoding can be disassociated by forcing dense encodings and only moderately distinctive patterns (i.e., $k \leq 3$), this property on its own is not sufficient to impede pattern mining attacks. In the following, we will thus analyze the potential risk of partial patterns providing sufficient information for pattern mining attacks.

5.3.1 Probability of Preserving Partial Patterns

In the following, we assume that the basic Bloom encoding exhibits a q-gram pattern $\mathcal{I}_Q$. For a sub-pattern $\mathcal{I}_S \subset \mathcal{I}_Q$, the probability that it is preserved during randomization is

$$\Pr\left(E[i] = 1 \text{ for all } i \in \mathcal{I}_S \mid \mathbb{E}_{\mathcal{I}_Q}\right) = \left(1 - \frac{1}{2} \cdot f\right)^{|\mathcal{I}_S|} \tag{8}$$

Preserving the full pattern $\mathcal{I}_Q$ is hence by the factor $(1 - \frac{1}{2} \cdot f)^{|\mathcal{I}_Q| - |\mathcal{I}_S|}$ less likely than preserving a sub-pattern $\mathcal{I}_S$. This means that shorter sub-patterns are more likely to be preserved. They are, however, also more likely to be falsely introduced via collisions and randomization, as indicated by Figure 4.

To consider this threat in more detail, we analyze the conditional probability of a bit position being set in co-occurrence with another bit position. To account for biases of different bit positions, we also consider the probability of an individual bit position being set in an encoded data set, as well as respective differences between different bit positions.

Note that the probability of a bit position being set depends on the length of the cleartext, the q-gram frequencies in the cleartext data set, and the q-gram patterns (which in turn depend on l, q, k, and the specific hash functions chosen). As this provides us with many different variables that can drastically influence the outcome and hence make general statements difficult, we perform this part of our analysis more practically.

5.3.2 Bit Value Co-Occurrence

To evaluate the co-occurrence of bits, we define the co-occurrence matrix as an $l \times l$ matrix, which we denote by C_B for the basic encoding and by C_E for the randomized encoding.

Encoding parameters	Max. co-occ. bias ($f = 0.0$)	Max. co-occ. bias ($f = 0.4$)
$l = 64,\ k = 2, q = 2$	0.2450	0.0951
$l = 128, k = 2, q = 2$	0.4714	0.1468
$l = 150, k = 3, q = 2$	0.3854	0.1230
$l = 500, k = 5, q = 2$	0.6039	0.1879

Table 2: The maximum co-occurrence bias observed for non-diagonal entries of the co-occurrence bias matrices for different encoded versions of the DGA-2000 data set.

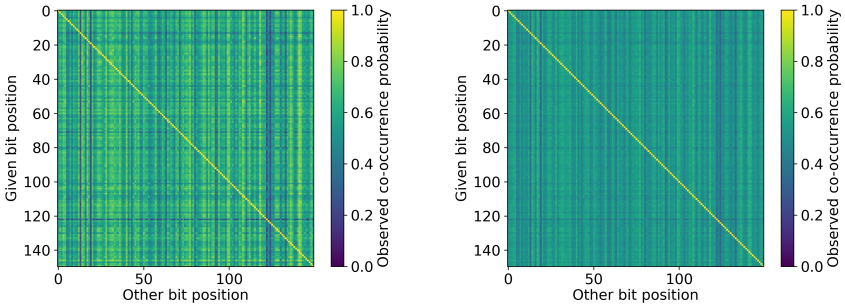
We define the co-occurrence matrix via its entries as $C_B[i,j] \coloneqq \Pr(B[j] = 1 \mid B[i] = 1)$, and analogously $C_E[i,j] \coloneqq \Pr(E[j] = 1 \mid E[i] = 1)$ for $i, j \in \{1, ..., l\}$. The entry at position $[i,j]$ in the co-occurrence matrix hence shows the probability of the bit at position j being set, given that the bit at position i is set. A value of 1 implies that the bit at position j is always set, if the bit at position i is set, and a value of 0 means that the bit at position j is never set, if the bit at position i is set.

Figure 5 shows a plot of the co-occurrence matrices (both basic and randomized) for the DGA-2000 data set. It indicates that the probability of a bit being set in co-occurrence with a given bit is primarily influenced by how likely it is to be set in general. As seen in Figure 6, this value distribution is not uniform. Nevertheless, the randomization procedure weakens this effect, as is expected based on the effect of the randomization procedure on the value distribution of a bit position.

For a clearer interpretation of the co-occurrence of 1-bits, we define the co-occurrence bias matrix. Based on the definition of the co-occurrence matrix, we define the co-occurrence bias matrix as an $l \times l$ matrix, which is denoted by $\hat{C}_B$ and defined as $\hat{C}_B[i,j] \coloneqq C_B[i,j] - \Pr(B[j] = 1)$ for the basic encoding procedure. Analogously, we denote the co-occurrence matrix for the randomized encoding procedure by $\hat{C}_E$ and define it as $\hat{C}_E[i,j] \coloneqq C_E[i,j] - \Pr(E[j] = 1)$. The co-occurrence bias matrix hence considers the underlying (and potentially skewed) value distribution of j, and filters it out to isolate the observable impact that a 1-bit at i has on the value of j.

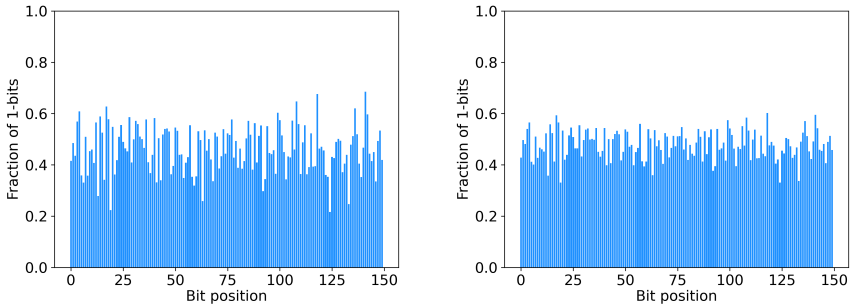
Figure 7 visualizes the co-occurrence bias matrices for the co-occurrence matrices shown in Figure 5. The plots indicate that the randomization procedure drastically reduces the co-occurrence bias, but does not fully eliminate it. Entries that are highly biased in the co-occurrence bias matrix for the basic encodings, however, are significantly less distinct in the co-occurrence bias matrix for the randomized encodings (with the expected exception of the diagonal entries). Nevertheless, some patterns visible in the co-occurrence bias matrix for the basic encodings are also visible in the randomized counterpart, but to a lesser extent. As supported by additional results shown in Table 2, this effect can be weakened by forcing more collisions in the basic Bloom encodings, which ensures that every bit position is associated with more q-gram patterns.

A smaller co-occurrence bias is beneficial, since pattern mining attacks commonly require distinct observable co-occurrence probabilities [Vidanage et al., 2019]. The randomization procedure hence seems to address the most problematic entries, and the degree of noise can be adjusted via f to force the co-occurrence bias below a desired threshold. If aligned with the previous observation that small values for k are required to break the correlation between q-gram occurrence in cleartext and the occurrence of their respective patterns in the encoding, also potential effects of aggregating co-occurrence biases across multiple bit positions remain negligible.



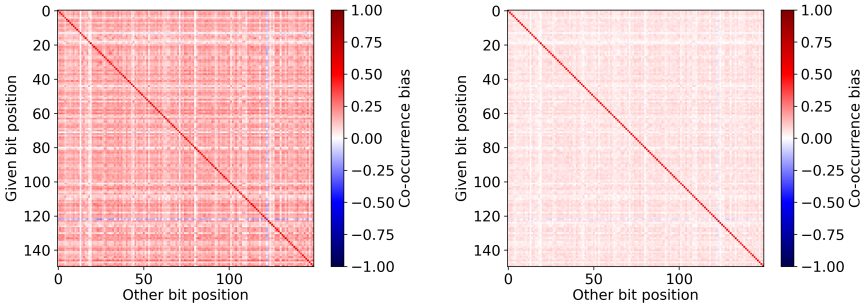
(a) C_B *for* $l = 150, k = 3, q = 2, f = 0.0$ *(b)* C_E *for* $l = 150, k = 3, q = 2, f = 0.4$

Figure 5: Visualization of the co-occurrence matrices C_B *(without randomization) and* C_E *(with randomization) for the DGA-2000 data set. For given bit positions* i *and other bit positions* j*, Subfigure 5a shows the values* $C_B[i, j] = \Pr(B[j] = 1 \mid B[i] = 1)$ *and Subfigure 5b the values* $C_E[i, j] = \Pr(E[j] = 1 \mid E[i] = 1)$*.*



(a) $l = 150, k = 3, q = 2, f = 0.0$ *(b)* $l = 150, k = 3, q = 2, f = 0.4$

Figure 6: Fraction of 1-bits per bit position without (Subfigure 6a) and with (Subfigure 6b) randomization on the DGA-2000 data set.



(a) $\hat{C}_B$ *for* $l = 150, k = 3, q = 2, f = 0.0$ *(b)* $\hat{C}_E$ *for* $l = 150, k = 3, q = 2, f = 0.4$

Figure 7: Visualization of the co-occurrence bias matrices $\hat{C}_B$ *(without randomization) and* $\hat{C}_E$ *(with randomization) on the DGA-2000 data set, based on the co-occurrence matrices depicted in Figure 5.*

6 Practical Impact on Pattern Mining Attacks

Our analysis showed that the use of dense encodings, short q-gram patterns and randomization seem to beneficially impact the correlation between bit positions from a privacy point of view. To further evaluate whether this is reflected by the success of a pattern mining attack, we have carried out respective attacks on differently encoded data sets. In the following, we discuss our choice of attack, the steps it applies, the specific setup we chose for our experiment and the results of the attack.

6.1 Choice of the Attack

Different kinds of pattern mining attacks on Bloom encodings have been proposed in the area of PPRL. For our evaluation, we chose the one presented in [Christen et al., 2018], which takes a 2-step approach to re-identify cleartexts from encodings. First, it mines bit patterns based on bit position co-occurrences in the encodings and correlates them with the most frequent q-grams in a background knowledge data set to identify the patterns of frequent q-grams. The background knowledge data set is assumed to have a q-gram distribution similar to the data set behind the encodings. In the second step, cleartext candidates are then re-identified for each encoding based on the previously identified q-gram patterns.

While more sophisticated pattern mining attacks have shown to provide a higher success rate in the PPRL setting, these attacks also rely on stronger assumptions. The attack described in [Vidanage et al., 2019] extends the attack we chose by adding an intermediate step in which additional less frequent q-grams are re-identified based on the initially re-identified ones. The quality of this extended re-identification step, however, is expected to rely on the quality of the initial q-gram re-identification step, which we evaluate via the chosen attack. Another pattern mining attack has been described in [Christen et al., 2019]. This attack, however, assumes the existence of duplicate samples in the used data sets and more heavily relies on the property of 0-bits disproving set membership in Bloom filters. Since we focus on the use of Bloom encodings in machine learning, we do not expect duplicate samples in our data. The use of randomization as a hardening technique additionally breaks the basic Bloom filter property of disproving set membership via 0-bits. We thus considered the attack described in [Christen et al., 2018] to be suited best for evaluating the general impact on pattern mining in our setting.

6.2 Attack Description

The chosen pattern mining attack described in [Christen et al., 2018] consists of two stages, a pattern mining stage and a cleartext re-identification stage.

Stage 1: The pattern mining stage utilizes a background knowledge data set which is assumed to have a q-gram distribution similar to the cleartext data set behind the targeted set of encodings. Additionally, the first stage of the attack is parameterized via the q-gram length, the minimum degree of required distinctness between two q-grams, and a minimum partition size. The minimum required distinctness of two q-grams is used to determine when the frequencies of two q-grams in the background knowledge data set are sufficiently distinct to confidently map them to mined bit patterns. Bit patterns are mined iteratively on sub-sets of the target encoding data set (referred to as partitions), where each partition is at most as large as the one of the previous iteration. The minimum partition size defines a lower limit on the size of these partitions, as the q-gram frequency

distribution in very small partitions may deviate too much from the frequency distribution in the background knowledge data set. The pattern mining stage returns a mapping $\mathbf{F}$ of re-identified frequent q-grams and their patterns, and two mappings $\mathbf{A}^+$ and $\mathbf{A}^-$ containing the matching and non-matching re-identified q-grams, respectively, for each encoding in the target data set.

Stage 2: For the cleartext re-identification stage, the mappings of matching and non-matching q-grams $\mathbf{A}^+$ and $\mathbf{A}^-$ are utilized to identify cleartext candidates from the background knowledge. A cleartext sample is considered to be a candidate of an encoding, if it does not contain any non-matching q-grams from $\mathbf{A}^-$ and a minimum required number of matching q-grams from $\mathbf{A}^+$ for that encoding. The minimum required number of matching q-grams from $\mathbf{A}^+$ is provided as a parameter for the cleartext reconstruction.

We slightly adjusted the cleartext re-identification procedure compared to its original description in [Christen et al., 2018] to make it fit the DGA use case. Specifically, instead of using records from the background knowledge in full for identifying cleartext candidates, we split each record at its dots. As such, we target the re-identification of individual domain elements. To preserve basic information about the observed position of a domain element (first, middle, or last), we preserve dots when splitting a domain. Specifically, this means that the sample `'test.example.org'` would be split into the three domain elements `'test.'`, `'.example.'` and `'.org'`. In principle, this adjustment allows to re-identify cleartext candidates, even if the cleartext data set behind the encodings and the background knowledge data set are intersection-free, as long as a majority of their samples are taken from a comparable pool of samples.

6.3 Experiment Setup

We have implemented the attack described above using Python 3.9. As cleartext and background knowledge data sets, we randomly sampled two subsets of the data set used for the utility evaluation in [Nitz and Mandal, 2023], which is also briefly summarized in Section 2.2. Each of the two data sets contains 10 000 samples with a 50/50 label distribution (malicious/benign). The data sets do not contain any duplicate samples and are intersection-free. Across all experiments, we consistently used the same data set as the cleartext data set (from which we generated the target encodings with the respective parameter values) and the other data set as the background knowledge data set.

For generating the target encodings, we focused on the parameter values used in the utility evaluation summarized in Section 2.2. As such, we considered a q-gram length of $q = 2$, Bloom filter sizes of $l \in \{64, 128\}$, $k = 2$ many hash functions, and noise levels of $f \in \{0.0, 0.4\}$. For the attack parameters, we used a minimum partition size of 500 samples (5% of the size of the target data set) and considered two q-gram frequency counts f_1 and f_2 with $f_1 \geq f_2$ to be sufficiently distinct, if $100 \cdot 2 \cdot (f_1 - f_2)/(f_1 + f_2) \geq 0.01$. For cleartext re-identification, we required a cleartext candidate to contain at least 2 matching re-identified q-grams. We ran the attack without restricting the length of the mined patterns and used the same support threshold definition as [Christen et al., 2018].

The attack parameters have been adjusted to fit the DGA use case. While the evaluation of the original attack in [Christen et al., 2018] required significantly more distinct q-gram frequencies by considering distinction thresholds of 1 and 5, it is to note that the respective evaluation has been carried out on personal data (first name, last name, street address, city). In the DGA use case, however, we observed that even some of the most frequent q-grams in the background knowledge have comparable frequencies. Without respectively lowering the distinctness threshold, the pattern mining attack terminated without being able to mine any q-gram patterns, since the frequencies of the

Parameters	q-grams	TP	FP	FN	Avg. precision	Avg. recall
$l = 128, k = 2, q = 2, f = 0.0$	19	12	23	26	0.3860	0.3158
$l = 128, k = 2, q = 2, f = 0.4$	2	0	2	4	0	0
$l = 64, \ k = 2, q = 2, f = 0.0$	11	1	27	21	0.0303	0.0455
$l = 64, \ k = 2, q = 2, f = 0.4$	7	1	12	13	0.1429	0.0714

Table 3: Evaluation of the re-identified q-gram patterns in **F**.

most common q-grams were not considered to be sufficiently distinct to continue the attack. For the cleartext re-identification stage, we also lowered the minimum required number of matching re-identified q-grams to 2 (compared to 3 in [Christen et al., 2018]) to conceptually allow for the re-identification of a sample's top-level domain, which in some cases only consists of two 2-grams (e.g., `'.de'`, `'.ru'`, `'.tk'`).

6.4 Results

For our evaluation, we focused on the quality of the different individual results of the pattern mining attack. These consist of the mapping **F** of q-grams to their re-identified pattern, the mappings $\mathbf{A}^+$ of matching and $\mathbf{A}^-$ of non-matching re-identified q-grams to each target encoding, and the mapping **R** of cleartext candidates to each target encoding. For reasons of readability, we refer to the encodings as *long* if $l = 128$, as *short* if $l = 64$, and as *randomized* if $f = 0.4$.

6.4.1 Identification of q-Gram Patterns

We evaluated the quality of **F** by comparing the number of re-identified q-grams, the number of true positive, false positive and false negative pattern bits as the sum of the respective values per q-gram pattern, and (similar to [Christen et al., 2018]) the average precision and recall of the q-gram patterns. As we did not force patterns of a specific length, re-identified patterns could be of varying length, allowing for the identification of sub- and supersets of bit patterns. The results are summarized in Table 3.

The results show that the application of the Bloom encoding procedure with longer Bloom filters and without applying randomization does indeed provide the best q-gram re-identification results, both in terms of number of re-identified patterns and their quality. Nevertheless, it should be noted that even these attack results are significantly worse than the ones in the PPRL setting [Christen et al., 2018], which considers different encoding parameters (such as longer patterns) and targets personally identifying information. Salient in our evaluation is the case of longer encodings with randomization, which provided the worst re-identification result on every metric.

6.4.2 Matching and Non-Matching q-Grams per Encoding

As the re-identified q-gram patterns in **F** are used to identify suitable cleartext candidates, we evaluate the quality of the respective mappings $\mathbf{A}^+$ of matching and $\mathbf{A}^-$ of non-matching q-grams to the encodings. We determine the quality of $\mathbf{A}^+$ and $\mathbf{A}^-$ based on whether or not the respective q-grams have been part of the cleartext behind the encoding. We compare the quality of $\mathbf{A}^+$ in regard to average precision and recall, and the quality of $\mathbf{A}^-$ in regard to average precision only, where the average is taken over the respective

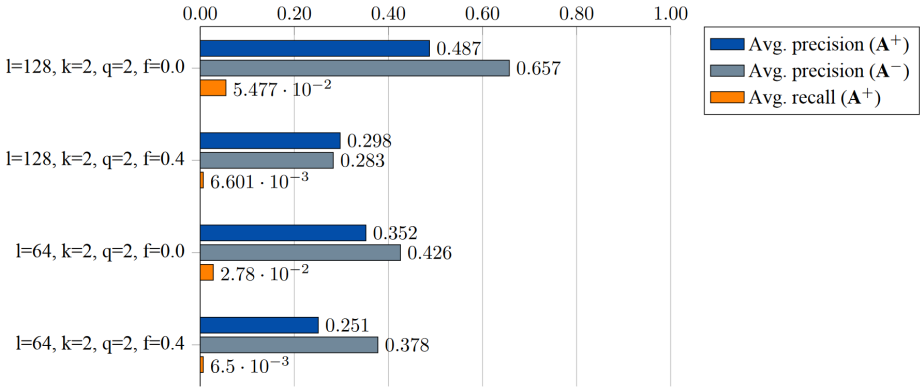


Figure 8: Evaluation of the matching ($\mathbf{A}^+$) and non-matching ($\mathbf{A}^-$) frequent q-grams identified by the pattern mining attack.

value of the individual encodings. The results are depicted by Figure 8. We did not evaluate the recall for $\mathbf{A}^-$, as the number of false negative non-matching q-grams (i.e., q-grams which do not appear in the cleartext behind the target encoding, but also not in the respective entry in $\mathbf{A}^-$) is primarily determined by the number of all possible q-grams, which dwarfs the number of true positives.

As for the re-identification of q-gram patterns before, the attack performed best on the longer non-randomized encodings. The low quality of the re-identified q-gram patterns, however, seemed to impact the assignment of matching and non-matching q-grams, as even in this case the average precision of matching and non-matching patterns is below 0.5 and 0.66, respectively. In general, we observe that shorter (and in this case also denser) encodings and randomization decrease the quality of $\mathbf{A}^+$ and $\mathbf{A}^-$. Interestingly, the precision does not drop below 0.25 even in the cases in which at most one bit was successfully re-identified across all re-identified q-gram patterns. Despite being rather low in all observed cases, the recall seems to be sensitive to the use of randomization.

6.4.3 Cleartext Re-Identification

The cleartext re-identification via $\mathbf{A}^+$ and $\mathbf{A}^-$ results in a mapping $\mathbf{R}$ which assigns a cleartext candidate set to each encoding. Similar to [Christen et al., 2018], we categorized the quality of these candidate sets by whether all (correct), some (partial), or none (wrong) of their elements are substrings of the cleartext behind the respective encoding. Additionally, we also documented the number of empty candidate sets, and considered the average precision of the cleartext re-identification stage, taken as the average of the individual precision per non-empty candidate set. The results are summarized in Table 4.

The evaluation of the cleartext re-identification exhibits a similar pattern as the re-identification of q-gram patterns and determination of matches and non-matches above. The highest success rate is achieved for the longer non-randomized encodings. However, in none of the cases, all re-identified candidates for an encoding were actually present in the cleartext behind the encoding. This is likely due to the overall low quality of re-identified q-grams as well as the condition that a cleartext candidate only requires two matching q-grams. Partial matches only occurred in the case of long non-randomized encodings and short randomized encodings, making up 8.55% and 0.67% of candidate

Parameters	Correct	Partial	Wrong	Empty	Avg. precision
$l = 128, k = 2, q = 2, f = 0.0$	0	855	3 323	5 822	0.0517
$l = 128, k = 2, q = 2, f = 0.4$	0	0	0	10 000	0
$l = 64,\ k = 2, q = 2, f = 0.0$	0	0	1 645	8 355	0
$l = 64,\ k = 2, q = 2, f = 0.4$	0	67	1 348	8 585	0.0237

Table 4: Evaluation of the identified cleartext candidates in **R**.

sets, respectively. Considering the low quality of **F**, $\mathbf{A}^+$ and $\mathbf{A}^-$ in the case of short randomized encodings, we suspect that the partial matches are the result of chance. Overall, the average precision of non-empty candidate sets showed to be very low, scoring just over 0.05 in the highest case.

7 Implications for Parameter Selection

In the first part of our analysis we saw that randomization can be used to weaken the correlation between q-gram frequencies in cleartexts and the frequencies of their pattern in the encodings. Nevertheless, we observed that randomization on its own may not be sufficient to break this correlation, since it is unlikely that false patterns are introduced, if more than 2 bit positions need to be set by the randomization mechanism. We hence recommend the use of shorter q-gram patterns by choosing $k \leq 3$. Alternatively, dense encodings can be forced to achieve the same effect for longer patterns.

In the second part of our analysis, we observed that the pattern mining attacks on encodings using the parameter values chosen in [Nitz and Mandal, 2023] fail to both reliably re-identify q-gram patterns and reconstruct cleartext values. While randomization seemed to lead to a lower number of mined q-gram patterns, the quality of these patterns was too low to measure the impact of the randomization procedure. We suspect that the lower number of re-identified q-gram patterns is a result of the randomization procedure bringing the distribution of 1-bits per bit position closer to uniform (see also Figure 6), which lets the mining of co-occurring bit positions run into the support threshold earlier. Except for the case of choosing $l = 128, k = 2, q = 2, f = 0.0$, the quality of the re-identified q-gram patterns seemed to be primarily determined by chance.

We would like to emphasize that the data used for DGA detection exhibits a q-gram distribution that makes it difficult to relate mined bit patterns to q-grams. Specifically, the frequencies of some of the most frequent q-grams are almost identical. This difficulty in cleartext re-identification also remains when choosing less protective values as encoding parameter, but could potentially be addressed by more sophisticated pattern mining attacks that do relate patterns to q-grams not just solely based on their frequency. It should be kept in mind that using different data may require different parameter values, especially for the Bloom filter size l, to avoid sparse encodings. We thus recommend to analyze, whether a chosen set of parameter values breaks the correlation between q-gram frequencies in cleartext and q-gram pattern frequencies in the randomized Bloom encodings, if other kinds of data are used.

Lastly, we want to point out that we assumed a rather strong adversary who has access to a suitable background knowledge data set. As discussed above, the MLaaS setting does not require the MLaaS provider to know which underlying classification task is solved or which kind of data is encoded, hence further limiting the associated risk of cleartext re-identification in the MLaaS setting.

8 Discussion and Future Work

We have considered the use of both basic and randomized Bloom encodings in the MLaaS setting for the DGA detection use case. Specifically, this scenario aims to keep the data samples used for training and classification hidden from the MLaaS provider. As the most severe threat in this scenario, we have considered pattern mining attacks by an honest-but-curious MLaaS provider, assuming a strong adversary who has access to a suitable background knowledge data set and the substring length q.

Our analysis of pattern preservation comparing randomized Bloom encodings to basic ones revealed that relying solely on randomization via RAPPOR's randomized response step will likely not suffice to achieve an adequate level of protection. This is because randomization does weaken certain biases and correlations between cleartext and encoded data, but does not fully eliminate them unless close to all bits are randomized. Our findings indicate that the use of $k \leq 3$ aligns best with the protection provided by the randomization procedure. Further, l should be chosen such that sparse encodings are avoided. Based on the analysis of privacy properties associated with the length of q-grams carried out by [Mitchell et al., 2017], we recommend the use of $q = 2$.

By practically applying pattern mining attacks on both basic and randomized Bloom encodings of DGA detection data using the parameter values from [Nitz and Mandal, 2023], we found that the quality of re-identified cleartext candidates per encoding was poor. As these parameter values are also consistent with our recommendations based on the analysis of preserved patterns, we conclude that the parameter values used in [Nitz and Mandal, 2023] provide an adequate level of protection against pattern mining attacks in the DGA use case.

Nevertheless, we acknowledge that further investigations into both pattern mining attacks and other threats in the MLaaS scenario are required for a better understanding of a suitable privacy-utility trade-off. Specifically, the design of novel pattern mining attacks which specifically consider the targeted encodings as randomized may allow for better re-identification results. Based on the analysis of pattern preservation, we do however suspect that the quality of re-identification results for the parameter ranges considered in [Nitz and Mandal, 2023] still protect against respective advances.

In the context of future work, the evaluation of other hardening techniques (such as diffusion [Armknecht et al., 2023]) in machine learning use cases is of interest, as they may provide a better privacy-utility trade-off. We also consider the use of Bloom encodings as a sanitization approach for classification tasks beyond DGA detection (such as phishing detection [Nitz et al., 2021]) as an interesting extension of this work.

Acknowledgements

We would like to thank the IT-Security Research Group at RWTH Aachen University for providing us access to their PoC data set and implementation for training DGA classifiers. We would further like to thank the anonymous reviewers for the helpful feedback and suggestions. This work has received funding from the EU H2020 projects SAPPAN under grant agreement No. 833418 and CyberSEAS under grant agreement No. 101020560.

References

[Armknecht et al., 2023] Armknecht, F., Heng, Y., and Schnell, R. (2023). Strengthening privacy-preserving record linkage using diffusion. *Proceedings on Privacy Enhancing Technologies*, 2:298–311.

[Christen et al., 2019] Christen, P., Ranbaduge, T., Vatsalan, D., and Schnell, R. (2019). Precise and fast cryptanalysis for Bloom filter based privacy-preserving record linkage. *IEEE Transactions on Knowledge and Data Engineering*, 31(11):2164–2177.

[Christen et al., 2018] Christen, P., Vidanage, A., Ranbaduge, T., and Schnell, R. (2018). Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In *Advances in Knowledge Discovery and Data Mining*, pages 530–542, Cham. Springer International Publishing.

[Drichel et al., 2021] Drichel, A., Gurabi, M. A., Amelung, T., and Meyer, U. (2021). Towards privacy-preserving classification-as-a-service for DGA detection. In *2021 18th International Conference on Privacy, Security and Trust (PST)*, pages 1–10.

[Drichel et al., 2020] Drichel, A., Meyer, U., Schüppen, S., and Teubert, D. (2020). Analyzing the real-world applicability of DGA classifiers. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, ARES '20, New York, NY, USA. ACM.

[Erlingsson et al., 2014] Erlingsson, U., Pihur, V., and Korolova, A. (2014). RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 1054–1067, New York, NY, USA. ACM.

[Holmes et al., 2021] Holmes, B., Drichel, A., and Meyer, U. (2021). Sharing FANCI features: A privacy analysis of feature extraction for DGA detection. In *6. International Conference on Cyber-Technologies and Cyber-Systems*, pages 58–64. IARIA XPS Press.

[Mitchell et al., 2017] Mitchell, W., Dewri, R., Thurimella, R., and Roschke, M. (2017). A graph traversal attack on Bloom filter-based medical data aggregation. *Int. J. Big Data Intell.*, 4(4):217–226.

[Nitz et al., 2021] Nitz, L., Gurabi, M. A., Mandal, A., and Heitmann, B. (2021). Towards privacy-preserving sharing of cyber threat intelligence for effective response and recovery. *ERCIM NEWS*, 126:33–34.

[Nitz and Mandal, 2023] Nitz, L. and Mandal, A. (2023). DGA detection using similarity-preserving Bloom encodings. In *Proceedings of the 2023 European Interdisciplinary Cybersecurity Conference*, EICC '23, page 116–120, New York, NY, USA. ACM.

[Plohmann et al., 2016] Plohmann, D., Yakdan, K., Klatt, M., Bader, J., and Gerhards-Padilla, E. (2016). A comprehensive measurement study of domain generating malware. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 263–278.

[Schnell et al., 2009] Schnell, R., Bachteler, T., and Reiher, J. (2009). Privacy-preserving record linkage using bloom filters. *BMC Medical Informatics and Decision Making*, 9(1):41.

[Schnell and Borgs, 2016] Schnell, R. and Borgs, C. (2016). Randomized response and balanced Bloom filters for privacy preserving record linkage. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 218–224.

[Schüppen et al., 2018] Schüppen, S., Teubert, D., Herrmann, P., and Meyer, U. (2018). FANCI: Feature-based automated NXDomain classification and intelligence. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1165–1181, Baltimore, MD. USENIX Association.

[Vidanage et al., 2020] Vidanage, A., Christen, P., Ranbaduge, T., and Schnell, R. (2020). A graph matching attack on privacy-preserving record linkage. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 1485–1494, New York, NY, USA. ACM.

[Vidanage et al., 2019] Vidanage, A., Ranbaduge, T., Christen, P., and Schnell, R. (2019). Efficient pattern mining based cryptanalysis for privacy-preserving record linkage. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1698–1701.

[Yu et al., 2018] Yu, B., Pan, J., Hu, J., Nascimento, A., and De Cock, M. (2018). Character level based detection of DGA domain names. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.