Taming the Devil: Techniques for Evaluating Anonymized Network Data

S. E. Coull* C. V. Wright* A. D. Keromytis‡ F. Monrose* M. K. Reiter†

*Johns Hopkins University
Baltimore, MD
{coulls,cwright,fabian}@cs.jhu.edu

‡Columbia University
New York, NY
angelos@cs.columbia.edu

†University of North Carolina
Chapel Hill, NC
reiter@cs.unc.edu

Abstract

Anonymization plays a key role in enabling the public release of network datasets, and yet there are few, if any, techniques for evaluating the efficacy of network data anonymization techniques with respect to the privacy they afford. In fact, recent work suggests that many state-of-the-art anonymization techniques may leak more information than first thought. In this paper, we propose techniques for evaluating the anonymity of network data. Specifically, we simulate the behavior of an adversary whose goal is to deanonymize objects, such as hosts or web pages, within the network data. By doing so, we are able to quantify the anonymity of the data using information theoretic metrics, objectively compare the efficacy of anonymization techniques, and examine the impact of selective deanonymization on the anonymity of the data. Moreover, we provide several concrete applications of our approach on real network data in the hope of underscoring its usefulness to data publishers.

1 Introduction

The availability of realistic network data plays a key role in fostering new solutions to the latest network and security research problems. Unfortunately, due to the sensitive nature of the information that may be contained within such data, organizations are often reluctant to make network data available. Therefore, significant emphasis has been placed on developing techniques for *network data anonymization*, *i.e.*, the process of sanitizing data before release so that sensitive information cannot be extracted [20, 28, 29, 30]. Emboldened by the availability of these tools, the community has developed several data repositories to encourage more sharing among researchers and practitioners [22, 26, 10]. However, when it comes to the efficacy of these anonymization tools, the devil is in the details [20].

[1]This paper has been modified from its published version to correct notational errors in Section 6.

Indeed, the anonymization systems relied upon today provide few, if any, guarantees on privacy. Moreover, they do not allow the publisher to quantify the risks involved with publishing a particular dataset. For the most part, progress in the development of anonymization systems has occurred primarily in reaction to the evolving threats to privacy or the need for greater utility (*e.g.*, [19, 14]). Given the reactionary nature of network trace sanitization, it comes as no surprise that several recent works have shown that, in certain cases, it is possible to infer network topology [8], deanonymize public hosts [8, 15, 2, 3, 23], and identify web browsing behaviors [7, 15] despite the use of state-of-the-art anonymization. The existence of such attacks and the fact that there are, at present, no rigorous methods for evaluating the anonymization of network data may lead to a loss of confidence in anonymization techniques and a decrease in available network data.

Intuitively, it would seem that the problems faced in anonymizing network data can be overcome by applying advances in other domains. For instance, the problem of evaluating anonymity in network data would appear to be closely related to that of inference control in databases (*e.g.*, [24, 13, 4, 12]); each packet or flow in the network data might be considered as a row in a database, lending itself to the direct application of a vast body of work on privacy preserving techniques for databases. Unfortunately, the peculiar nature of network data makes the direct application of these techniques problematic. Most notably, when dealing with network data, it is difficult to know *a priori* which fields should be considered sensitive. Finding such channels of information leakage lies at the heart of the network data sanitization problem. Furthermore, some of the most sensitive information in the network data, such as web browsing activities, are encoded in the distributions of values over several packets or flows. These differences warrant the development of new techniques tailored to evaluating network data.

In this paper, we frame the analysis of anonymized network data as the simulation of an adversary whose

goal is to distinguish the true identity of an anonymized object, such as a host, from among all possible unanonymized identities. Generally speaking, attacks on network data anonymization proceed as follows. The adversary approximates the expected feature distributions (*e.g.*, distribution of local port numbers) for unanonymized objects by using public information sources, such as DNS records or web search engines. She then calculates the feature distributions for each of the anonymized objects, and compares them to the approximated distributions for the unanonymized objects. If a one-to-one mapping exists between the anonymized and unanonymized distributions, then she can infer the identity of the object. This general algorithm for the adversary's behavior describes a broad and meaningful class of inference attacks, including several attacks against anonymized network data [8, 15, 2, 3, 7, 23].

Our approach simulates such an adversary by comparing objects from the unanonymized and anonymized data directly, thereby assuming that she has perfect knowledge of the unanonymized distributions. To analyze the anonymity of an object, we calculate the feature distributions for that anonymized object, and compare them to the feature distributions of all objects in the unanonymized data that are of the same type (*e.g.*, host objects). From this comparison, we derive a probability distribution over the object's possible unanonymized identities. To quantify the anonymity of the object, we calculate the information entropy of this derived probability distribution. Furthermore, we model the auxiliary information gained by the adversary from meta-data provided by the data publisher, and from deanonymization of objects within the data.

The main contributions of this paper are in showing that, through the use of entropy metrics, publishers can identify objects that are at risk of deanonymization, and can objectively compare the performance of various anonymization systems and policies. Moreover, by simulating the adversary's auxiliary information, a publisher can examine the impact of deanonymization on the anonymity of other objects. To underscore the utility of our approach, we provide several concrete applications of this analysis to real network data.

2 Preliminaries

The analysis techniques presented in this paper make heavy use of concepts from probability and information theory. For ease of exposition, we first review the concepts and definitions used throughout the remainder of this paper.

Probability Distributions and Random Variables

Our analysis is founded on the examination of joint and marginal probability distributions computed from the set of values found in the various fields of the network data. In general, we can consider a *random variable* X that describes the probability distribution over values from a single field in the network data. A *joint distribution* on fields represented by random variables X and Y describes the probability distribution of the values from X and Y that occur together, and the probability mass function for the joint distribution is denoted as $p(x, y)$, where x and y are values taken from the sample spaces of X and Y, respectively. A joint distribution can be created from an arbitrary number of random variables by generalizing the definition accordingly. Similarly, a *marginal distribution* on a field represented by the random variable X would describe the probability distribution of the values from X alone, and the probability mass function for the marginal distribution is denoted as $p(x)$. Since the fields found in network data contain discrete values, we represent marginal distributions as a histogram where each bin represents the probability mass for a single value in the sample space of the distribution. Likewise, a discrete joint distribution on random variables X and Y would be described such that each bin represents the probability mass for the pair of values x and y taken from their respective sample spaces, and so on for increasing numbers of random variables.

In some cases, the values in the network data may change slightly, although their underlying meanings are equivalent. For instance, if two TCP connections send the exact same data, then their histograms of packet sizes should be exactly the same (*i.e.*, their bin values and amplitudes should be the same). However, various changes could occur, such as TCP retransmissions, which skew the histograms. One way of overcoming this problem is to implement a smoothing technique on the distribution [5]. In practice, any number of smoothing strategies may be applicable, but for our analysis we implement a very simple smoothing strategy that proceeds as follows. For a random variable Z representing the distribution to be smoothed, we find the standard deviation of Z and initially create one bin for each value such that the bin represents the range of that value plus/minus the standard deviation. Those bins with overlapping ranges are merged into a single bin such that the smoothed histogram generated contains bins representing the probability mass for each permissible range of values.

Information Entropy and Mutual Information

The concept of *information entropy* [9] is central to our analysis in that it provides a single value that quantifies the amount of uncertainty in a random variable, and acts as an intuitive indicator of anonymity [25, 11]. The entropy of a random variable X represents the amount of information gained by learning the outcome of X, and is calculated as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \qquad (1)$$

The entropy of X takes its minimum value, zero, when all probability mass is centered on a single value, and it takes its maximum value, $\log N$ where N is the size of the sample space, when all values are equiprobable.

Likewise, the *mutual information* [9] between two random variables, X and Y, represents the amount of information gained about one variable by learning the outcome of the other, and is calculated as:

$$I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \quad (2)$$

The mutual information between two random variables is symmetric, and takes its minimum value of zero when the two variables are statistically independent. Naturally, since mutual information indicates the information shared by the two variables, it is limited by the minimum of their information entropies. To transform mutual information into a more intuitive notion of correlation, we can normalize it by its maximum value, as shown in Equation 3. The *normalized mutual information* takes values near one when the two variables are heavily correlated, and values near zero when they are independent. This normalized version makes it far more intuitive to set a defining threshold between correlation and independence.

$$\overline{I(X;Y)} = \frac{I(X;Y)}{min(H(X), H(Y))} \quad (3)$$

L1 Similarity Metric

In order to appropriately quantify the anonymity of the network data, we require a similarity metric to compare distributions to one another. Since we focus our analysis on discrete distributions represented by histograms, an intuitive choice for this similarity metric is the *L1 similarity*. The L1 similarity represents the difference between the maximum L1 distance minus the sum of the absolute differences between each of the corresponding bins in the two distributions, X and Y, as shown in Equation 4. In general, the bins in X and Y correspond to one another if they represent the same value.

$$sim(X,Y) = 2 - \sum_{z \in \mathcal{X} \cup \mathcal{Y}} |P(X=z) - P(Y=z)| \quad (4)$$

The L1 similarity takes its maximum value, two, when both distributions are identical, and its minimum value, zero, when the sample spaces of the distributions are completely disjoint.

In our analysis, the L1 similarity calculation is complicated by the fact that the anonymization process may have permuted or otherwise changed the values in the anonymized data (*e.g.*, shuffling port numbers). In this case, the values that the bins represent will no longer map directly between the unanonymized and anonymized distributions. We will reexamine this issue in Sections 3 and 6.1, and show how to use a mapping relation to correctly simulate the adversary's information about the correct mappings in this L1 similarity calculation.

3 Adversarial Model

The underlying principle of our analysis is a realistic simulation of an adversary whose goal is to deanonymize the network data. For clarity, we explicitly define the goal of our simulated adversary, state our assumptions on the adversary's power, and describe how those assumptions relate to real-life adversaries.

The Adversary's Goal

In our analysis, an *object* is characterized by a set of distributions on the features of the network data (*i.e.*, fields or groups of fields), which represent its presence in the data. Thus, an *(un)anonymized object* is an object with its distributions calculated from the (un)anonymized version of the network data. Given an anonymized object and a set of unanonymized objects, the adversary's goal is to create a mapping between the anonymized object and its unanonymized counterpart by comparing the feature distributions of the anonymized object to all unanonymized objects.

This definition of the adversary's goal maps closely to that of a real-life adversary mounting an inference attack. In both cases, the adversary compares feature distributions to infer the identity of an anonymized object. The primary difference between the two is that the real adversary must approximate the set of unanonymized objects and their distributions, while our simulated adversary has exact information about the objects and their distributions, drawn directly from the unanonymized network data. In doing so, we perform a *worst-case* analysis of the indistinguishability of the objects.

Auxiliary Information

Beyond the feature distributions themselves, the adversary also has access to *auxiliary information*, which she can use to refine the accuracy of her inferences. One of the primary problems for the adversary in deanonymizing the data lies in accurately comparing the anonymized distributions to the unanonymized distributions. The anonymization process is intended to make it difficult for the adversary to directly compare the distributions, and so a relation must be applied which maps the anonymized values to their potential unanonymized counterparts. The auxiliary information is used to refine this mapping relation.

One piece of auxiliary information available to a real-life adversary is expert knowledge of the network data semantics. From knowledge of the semantics, the adversary can examine the relationships among the fields in the network data to select features that best describe the

Object Type	Definition
Host	Distinct Local IP
Web Page	Distinct Local IP and (Interarrival time $<$ n seconds and Remote Port = 80)
User	Distinct Local IP and (User,Distinct IP,Time in Authentication Log)

Table 1: Example definitions used to group records into objects

objects. In our simulation, the fields in the network data are annotated with their semantic type, and an automated feature selection algorithm is used to extract the features upon which we perform our analysis. The annotation and feature selection process is discussed in Sections 4 and 5, respectively.

The real-life adversary also has access to meta-data, which the data publisher provides to researchers to explain the context of the anonymized network data [21]. This meta-data might describe which packets had incorrect checksums, the anonymized IP prefixes for the network where the data was collected, or the types of anonymization (if any) applied to each field in the data. Information, such as the anonymized prefixes, can be learned directly from the meta-data, which allows the adversary to refine the mapping relation and increase the accuracy of her inference attack. Also, other information from the meta-data, such as the type of anonymization used, helps in defining a set of algorithms that the adversary can use to extract additional refinements to the mappings after objects are deanonymized. The meta-data information is added to our simulation as annotations to the data, and we simulate the adversary's use of the learned information by implementing a mapping relation, which is described in greater detail in Section 6.1.

4 Annotating the Data

For our purposes, network data consists of n *records*, where each record is a tuple of m *fields* derived from a single packet or network flow [6]. A packet record might contain fields taken from the link, network, and transport layer headers of a packet. Flow records, on the other hand, contain fields which summarize information about all packets that were transmitted during a single session between two endpoints.

Network data can contain records generated due to the presence of several types of objects. At a high level, we might define a host object by the records sent or received by a single host, a web page object might consist of those records created due to the download of a specific web page, or a user object could be defined by all of the records sent or received by a host while a specific user was logged on to that system. To define an object in our analysis, the data publisher describes constraints on the fields within the network data (*e.g.*, each IP address from a given prefix is to be considered a host object). Clearly, the definition of an object is limited only by the data publisher's knowledge of the object when the data was collected. Some objects, like hosts, can be defined with relatively little information by considering each unique IP address from the data publisher's administrative domain to be a host object. Conversely, defining objects like web pages or users may require extra information, such as web proxy or authentication logs. Example object definitions can be found in Table 1.

Once the objects are defined, the data publisher describes the characteristics and semantics of each field in the data. This annotation of the data ensures that the analysis treats each field appropriately with respect to the values it contains. For our purposes, three annotations on the data are required, which are also required by current anonymization tools [20, 28]. First, the publisher provides the IP prefixes of the local networks that appear in the data, along with their anonymized counterparts. This annotation ensures that our techniques only analyze the anonymity of those objects within the publisher's administrative domain, and allows us to reformat the data in terms of local and remote, rather than source and destination. Second, the publisher lists the semantic types for each field, including a difference operator for that semantic type and the type of smoothing applied (if any). For instance, fields containing IP addresses would have the type `IP`, with a difference operator of XOR and no smoothing. By assigning types to the fields, we can automatically extract semantically meaningful information about how these fields relate to one another. Third, the data publisher lists the type of anonymization applied to each field. The anonymization type, such as prefix-preserving IP anonymization or deterministic mapping, dictates the algorithm to be used in reversing the anonymization and augmenting the adversary's auxiliary information. Note that in many cases, the semantic and anonymization types can be set to use default settings for the given field, thereby limiting the burden on the data publisher.

5 Feature Extraction and Selection

Many of the fields in the network data share complex and semantically meaningful relationships which can be considered as *features*. For instance, port scanning behavior that might fingerprint a host is visible in the

Original		Inter-record		Intra-record		
Local IP	Remote IP	Local IP	Remote IP	Original Local IP, Original Remote IP	Original Local IP, Inter-record Local IP	...
192.168.0.10	192.168.0.250	0.0.0.0	0.0.0.0	0.0.0.240	192.168.0.10	
192.168.1.50	192.168.10.20	0.0.1.56	0.0.10.238	0.0.11.38	192.168.0.10	...
⋮	⋮	⋮	⋮	⋮	⋮	

Figure 1: Example data after creating inter- and intra- record fields with semantic type of IP and difference operator of XOR

way the destination port numbers change from record to record. Also, combinations of fields may produce a unique distribution where the constituent marginal distributions on those fields are not unique. Before applying our analysis techniques, it is important to examine the semantically meaningful relationships among fields, and extract the features that best represent those relationships.

5.1 Discovering Semantically Meaningful Relationships

The semantic types of the fields are used to characterize their relationships within the data, both within a record (*intra-record*) and across two or more records (*inter-record*). Initially, each of the n records in the network data contains m fields annotated with their semantic type. For each of the m initial fields, a new inter-record field is added to each row that inherits the type of the initial field. Likewise, we create a new intra-record field for each pair of fields, both initial and inter-record, with the same semantic type. The newly created intra-record field inherits the semantic type of the fields used to create it.

To populate the value for an intra-record field in record k, we use the difference operator defined for its semantic type and calculate the difference between its two constituent fields in record k. For the value of an inter-record field in record k, we compute the difference between the value of its initial field at row k with the same field's value at row $k-i$ (in our case, $i = 1$), where the records are sequential and all belong to a single object. For the inter-record field values at record $k = 0$, we set the difference to be zero since there is no predecessor record.

Figure 1, for instance, shows the resulting network data after inserting the inter- and intra-record fields. The data initially contains the Local and Remote IP fields. Then two inter-record fields are created to calculate the difference between values in the Local and Remote IP fields across consecutive rows (*i.e.*, $i = 1$), respectively. Finally, $\binom{4}{2}$ intra-record fields are created for every pair of initial and inter-record fields with the same type to calculate the difference between values within the same record.

We note that the network data may contain some classes of subliminal channels that are not covered by our discovery process, such as complex active probing attacks [1, 27]. Discovering all possible subliminal channels in network data quickly becomes computationally infeasible since it requires the examination of *all* inter-record and intra-record relationships, whether they are semantically meaningful or not. Instead, we focus our analysis on a much narrower subset of these relationships that represent typical areas of information leakage that naturally occur within network data.

5.2 Feature Selection

When examined in isolation, the distribution of values in each of the fields may be similar among the objects being analyzed. However, when examined in aggregate these fields may be unique and could be used to distinguish the object. To see why, consider the case where two hosts have a variety of local port and remote IP address values in their respective records. Taken individually, the distributions on the local port and remote IP values for these hosts will be similar in an information theoretic sense. However, if one host always communicates with a single remote IP on a given port, while the other host communicates with a variety of remote IPs on that same port, then the two hosts will be distinguishable.

A naïve solution to this problem is to analyze all possible joint distributions on these fields, but as with the inter-record relationships, this will quickly lead to an intractable number of distributions to examine for each object. Instead, we select those joint distributions whose constituent marginal distributions are heavily correlated with one another, since the scenario described above can only occur when the marginal distributions are correlated to some degree. Moreover, if two fields are highly correlated, then analyzing both fields for anonymity is redundant, and thus we can reduce the computational requirements of the analysis by examining the joint distribution instead.

The process of selecting a mutually independent set of features proceeds as follows. For each pair of fields (including the newly created inter- and intra-record fields), we calculate the normalized mutual information between their marginal distributions, as shown in Section 2. If the normalized mutual information is above

Anonymization Type	Deanonymized Values		aux_{t+1} Relation
	Anonymized	Unanonymized	
Deterministic Mapping	port 150	port 80	150 → 80
CryptoPAn [14]	200.120.10.10	128.2.250.220	200.120.10.6 → 128.2.250.208/29
Pang *et al.* [20]	200.120.10.10	128.2.250.220	200.120.10.6 → 128.2.250.0/24

Table 2: Examples of reversing anonymization and updating aux_t relation

some threshold (0.99 in our evaluation) we group the two fields together. We further combine groups of fields if they share at least one field in common. Finally, we create a singleton group for each field which has not been added to any other group. When the feature selection process completes, we arrive at several groups of fields where the fields within a group are transitively correlated to one another, and fields in different groups are mutually independent according to our threshold of correlation. We take this set of mutually independent groups of fields to be the *features* upon which we perform our analysis.

6 Analyzing Network Data Anonymity

The primary goal of our analysis is to quantify the degree to which anonymized objects are distinguishable within the provided network data. To perform this analysis, we compare the features of an anonymized object to those of all unanonymized objects using the L1 similarity (see Section 2), and use this comparison to derive a probability distribution on the potential true identities of the object. Finally, we use the resultant probability distribution to calculate an entropy measure that quantifies the distinguishability of the anonymized object. However, it is important that the simulation of the adversary's behavior accurately reflect the auxiliary information available to the adversary at each step in the analysis.

6.1 Modeling Auxiliary Information

The auxiliary information available to the adversary can be modeled as a relation $aux_t : A \rightarrow \mathcal{P}(U)$, where t is the current time step in our simulation of the adversary, A is the set of anonymized values, and $\mathcal{P}(U)$ is the power set of unanonymized values. Thus, for some anonymized input, aux_t outputs a set of one or more potential unanonymized values given the information available to the adversary at time t. At the start of the analysis ($t = 0$), the adversary only has information gained from annotations on the data. Thus, for fields with no anonymization applied to them, $aux_{t=0}$ will output the same value that was used as input, since both the anonymized and unanonymized values are the same. For fields with deterministic mappings, the relation would return all unanonymized values for that field since the adversary does not know which is the correct mapping for the anonymized input. For example, in the case of prefix-preserving IP anonymization, the relation would output a set of unanonymized IPs that is consistent with the adversary's knowledge of the local enterprise IP prefix.

As the analysis progresses, we simulate the knowledge gained by the adversary after deanonymizing objects in the data. Upon deanonymizing an object, the adversary can reverse the anonymization by running an algorithm that compares anonymized and unanonymized values for the deanonymized object. A new relation aux_{t+1} is generated from the previous relation at time t and the newly learned information from the latest deanonymization. For instance, if a field were anonymized using deterministic mapping, the adversary learns the mapping between pairs of anonymized and unanonymized values, and the new relation is created such that those pairs have a one-to-one mapping. For prefix-preserving IP anonymization, the adversary checks the length of the prefix match between the anonymized IPs of the deanonymized object and all remaining anonymized IPs. The new relation then requires that those remaining anonymized IPs match their unanonymized counterparts up to the length of the calculated longest prefix match. Concrete examples of learning information from deanonymization are given in Table 2.

The auxiliary information relation is used to constrain our comparison between anonymized and unanonymized values in two ways. First, as mentioned in Section 2, L1 similarity requires the existence of a mapping between the bins of the distributions being compared. For comparison between anonymized and unanonymized values, the mapping between bins must respect the information available to the adversary in the relation aux_t. From among all of the possible mappings allowed by aux_t, we choose the mapping that provides the highest L1 similarity. Second, objects are defined by various constraints on the fields in the network data (*e.g.*, local IPs defining host objects). In order to provide a correct analysis of the anonymity of an object, each anonymized object should only be compared to those unanonymized objects for which a mapping is possible given the relation aux_t.

6.2 Object Anonymity

Our analysis begins by examining the anonymity of each anonymized object in isolation, called *object*

anonymity, where we consider only the information found in the data annotation. To perform this analysis, we begin with the relation $aux_{t=0}$ and the ℓ feature distributions $F_{i=1...\ell}$ identified using the technique from Section 5.2. For each feature distribution F_i, we calculate the L1 similarity between distribution $F_{i,A}$ for anonymized object A and distribution F_{i,U_j} for all unanonymized objects U_j. We use the similarity values as a count to approximate a probability distribution, represented as the random variable $X_{i,A}$, on the true unanonymized identity of A with respect to feature F_i as:

$$P(X_{i,A} = U_j) = \frac{sim(F_{i,A}, F_{i,U_j})}{\sum_{\forall U_k} sim(F_{i,A}, F_{i,U_k})} \quad (5)$$

Notice that this probability distribution considers only those objects present within the data, which follows from our assumption that the adversary knows the set of objects in the data and their exact unanonymized distributions. From this probability distribution on the identity of A, we can calculate the entropy of A with respect to feature F_i using Equation 1. In practice, there are many cases where a single unique distribution may lead to deanonymization. The entropy measures calculated for each of the feature distributions can be used to examine cases of specific attacks on those distributions, and whether the anonymization policy applied to those distributions was effective in providing anonymity.

We also calculate the overall entropy of an object from the entropy of the true identity probability distribution for each of its constituent feature distributions. Recall that the set of features produced from the feature selection process are *mutually independent* according to the threshold on the normalized mutual information. Therefore, given the entropy for each of the identity distributions for features $F_{i=1...\ell}$, we calculate the overall entropy of the identity of A as:

$$H(A) = \sum_{i=1}^{\ell} H(X_{i,A}) \quad (6)$$

Since we are summing the entropy value of each feature, the overall entropy value for a single object can be as high as $\ell \log N$, where ℓ is the number of features and N is the number of objects. The overall entropy of the object assumes that learning the identity of an object requires the adversary to learn the true identity for *all* feature distributions of that object. Though this may not be the case in many attack scenarios, we believe that the overall entropy values are still beneficial in understanding the general risk of deanonymization, and performing comparisons among different anonymization techniques.

6.3 Conditional Object Anonymity

Though the object anonymity provides useful information about the anonymity of the data, it does not capture the resilience of an anonymized dataset to deanonymization. For instance, it may be possible that the anonymity of objects within the data appears reasonable, but upon deanonymizing a single object the adversary learns enough information to undermine the remainder of the data. Therefore, it is imperative that data publishers have a method of detecting and quantifying the effects of deanonymization so that comparisons can be made among various anonymization techniques. In order to achieve this, we need to consider the anonymity of hosts conditioned upon the deanonymization of other objects, which we call *conditional object anonymity*.

To perform our conditional anonymity analysis, we first begin with the object anonymity analysis described above using the relation $aux_{t=0}$. We then choose the object with the lowest entropy value, and assume that it has been deanonymized by the adversary. In doing so, we create a new relation aux_{t+1} from the information learned from the deanonymization, and again perform the same analysis as the object anonymity case using aux_{t+1} to constrain the mapping of anonymized values. We continue this process iteratively, choosing the lowest entropy object at each step in the process, learning its information, and augmenting the mapping relation. The process completes when all anonymized objects have been deanonymized. The sequence of deanonymizations provides the data publisher with a notion of the adversary's strategy for deanonymizing the data, and allows different anonymization techniques to be compared to better understand their performance after deanonymization occurs.

7 Evaluation

To underscore the utility of our analysis techniques, we examine several concrete uses of those techniques in quantifying the anonymity of network data. We apply our techniques to a network trace recorded over a period of 24 hours at the edge of the Johns Hopkins University Information Security Institute (JHUISI) network. That network contains three subnets with a total of 237 hosts present in the data. For our analysis, we treat each of the 27,753 flows in the data as a single record with 19 features derived (as described in Section 5) from the original fields: `start time`, `end time`, `local IP`, `local port`, `local size`, `remote IP`, `remote port`, `remote size`, and `protocol`. In our data, the `remote size` and `local size` fields are the sum of the packet sizes sent to the remote and local hosts in the given flow, respectively. Our subsequent analysis focuses on the anonymity of host objects in the network data, where each distinct local IP address is treated as a different host object. In what follows, we show how a data publisher can (*i*) examine which objects may risk deanonymization due to unique distributions, (*ii*) objectively compare the anonymity provided by different anonymization techniques, and (*iii*) examine the impact

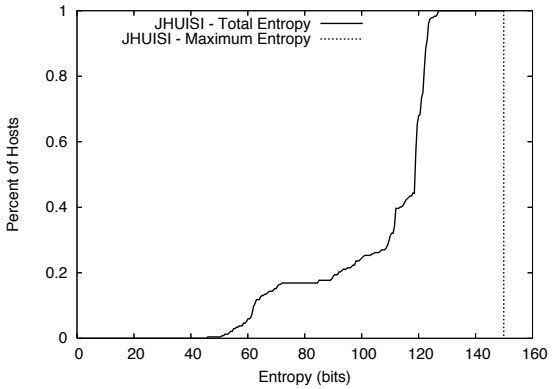

Figure 2: CDF of the total entropy of host objects, assuming all features are considered independently

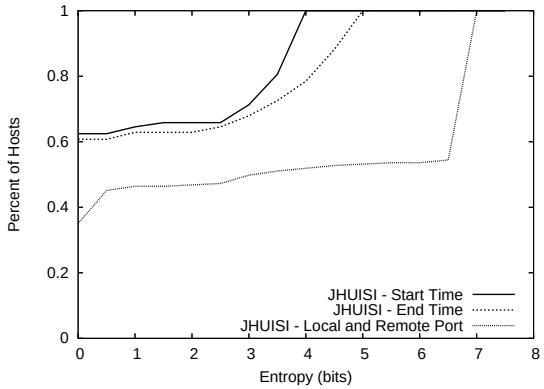

Figure 3: CDF of the three worst features

of selective deanonymizations on the anonymity of other objects in the data.

7.1 Quantifying overall anonymity

One of the most obvious applications of our analysis is in quantifying the anonymity of the objects in a particular dataset. Figure 2 shows the cumulative distribution function (CDF) of the total entropy for the host objects in the dataset, as calculated by Equation 6. A CDF, such as the one provided here, allows the data publisher to quickly and intuitively quantify the overall anonymity of the objects in the dataset. The graph shows, for example, that a substantial fraction of hosts in the data have relatively low entropy compared to the maximal entropy—assuming, of course, that all features are independent of one another. An equally useful view is the CDF of the lowest entropy features across all hosts in the dataset, as shown in Figure 3. This view provides the data publisher with a method for verifying the soundness of the chosen anonymization policy for each feature. For instance, Figure 3 shows that the remote and local port feature provides distinguishing information for many of the hosts in the data, and as such, it may be prudent to re-examine the anonymization policy for these features.

7.2 Comparative analysis

Another important contribution of this type of analysis is that it allows for an objective comparison of anonymization techniques and the impact that selective deanonymizations have on their soundness. As an example, we compare the prefix-preserving IP anonymization system of Pang *et al.* [20] to the CryptoPAn system [14], as applied to the JHUISI dataset. For context, recall that the CryptoPAn system is strictly prefix-preserving, meaning that if two unanonymized IP addresses share a k-bit prefix, then so will their anonymized counterparts. The anonymization system of Pang *et al.*, however, uses a pseudo-random permutation to anonymize subnet and host portions of the IPs separately. Therefore, the approach of Pang *et al.* only guarantees that two IPs in the same unanonymized subnet will also be in the same anonymized subnet, but all other prefix relationships among the IPs are broken. In this comparison, we examine the average entropy across all hosts at each step in the conditional anonymity analysis, as shown in Figure 4. The graph illustrates several interesting properties of these two anonymization techniques, which allow a data publisher to make informed decisions on their use. Most notably, the graph shows that the approach of Pang *et al.* provides greater privacy than the CryptoPAn approach, especially after a few deanonymizations (three, in this case) have occurred.

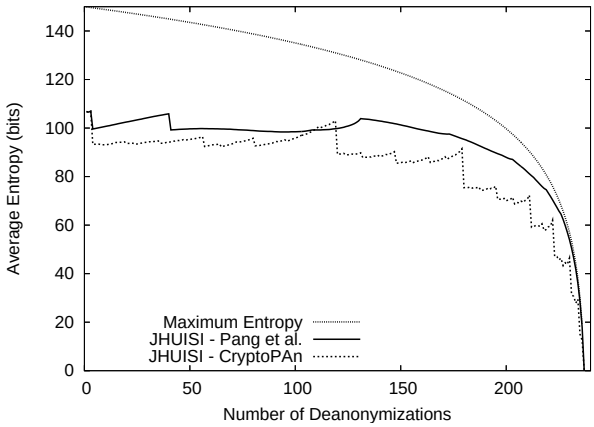

Figure 4: Comparison of Pang *et al.* anonymization to CryptoPAn

Additionally, Figure 4 shows an interesting pattern of increasing average entropy followed by an abrupt drop, which occurs repeatedly for CryptoPAn, and less often for the approach of Pang *et al.* To see why this pattern emerges, it is instructive to recall how we model the process of deanonymization. In particular, recall that at each iteration of the conditional anonymity analysis we choose the anonymized host with the lowest entropy

Hostname	Local, Remote Port Entropy	Search Engine References
simnet	0.0	77
spar	0.0	71
skdnssec	0.004	32
mirror	3.50	10
cable	6.707	4

Table 3: Entropy and references in a popular search engine for hosts in the example dataset

and deanonymize it, thus revealing its true identity. This deanonymization can result in two possible outcomes for the adversary's auxiliary information. In the case where the deanonymization results in learning prefix information about a substantial number of hosts, the average entropy will drop significantly due to the change in entropy for each of the affected hosts (*e.g.*, iteration 40 for Pang *et al.* and 120 for CryptoPAn). When the deanonymization provides no information or information on very few hosts, the result is an increase in the average entropy (*e.g.*, iterations 4–39 for Pang *et al.* and 81–119 for CryptoPAn). This somewhat unintuitive increase occurs in part due to the removal of very unique hosts that skew the average entropy downward and the fact that we gain little or no additional information about the remaining hosts. Once all unique hosts are exhausted and the adversary learns all available information about the anonymized subnets, the average entropy decreases in a smooth logarithmic curve, as seen in the Pang *et al.* plot. By contrast, the average entropy under CryptoPAn continues the pattern of increased entropy followed by drastic drops because additional information about the prefix is continually learned throughout the deanonymization process.

7.3 On the impact of selective deanonymizations

Given the continued threat to data privacy, one intriguing benefit of our analysis is that it allows data publishers to explore the effect of selective deanonymizations. As a concrete example, we apply our analysis techniques to the JHUISI dataset in the context of a recently discovered behavioral profiling attack [8]. In that attack, the adversary uses public information to estimate the services provided by hosts (as indicated by port numbers) within the network where the anonymized data was captured. By comparing the estimated services with those indicated in the anonymized data, the adversary can reveal the host's identity.

To evaluate the impact of this attack on the data, we examine the entropy for the local and remote port feature for each host in the anonymized data with publicly available information. For our purposes, we simply use the number of references found in a popular search engine for each hostname in our network data to approximate its available public information. Table 3 shows each of the hosts with at least one reference, along with their entropy values for the local and remote port feature. Notice that there is a substantial gap between the first three hosts and all others in Table 3 in terms of their public references and entropy values. The results indicate that *simnet*, *spar*, and *skdnssec* are at risk of deanonymization from the behavioral profiling attack, and, in fact, these were the same three hosts that were deanonymized in previous work [8].

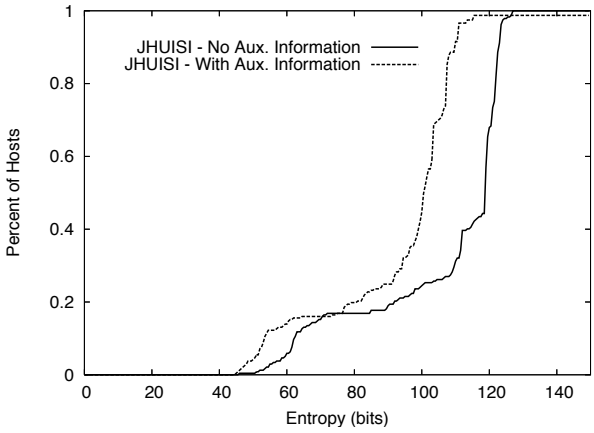

Figure 5: CDF of the the total entropy with and without deanonymized servers in the auxiliary information

Armed with insights regarding the hosts likely to succumb to the behavioral profiling attack, we can examine the effect that these deanonymizations would have on the rest of the data. We do so by assuming these hosts have been deanonymized and are now in the adversary's auxillary information, then repeat the computation in Section 6.3. Specifically, for our JHUISI data anonymized with the Pang *et al.* anonymization system, we take the anonymized values associated *simnet*, *spar*, and *skdnssec*, add their correct mappings to aux_t as if they have been deanonymized, and then recompute the total anonymity of the dataset.

Figure 5 shows the CDF for the JHUISI dataset without deanonymizations compared with the case where we assume the three hosts have been deanonymized and added to the adversary's auxiliary information. The graph shows a significant reduction in the anonymity of the remaining hosts in the dataset, with over 50% of hosts experiencing a reduction in entropy of approximately 20 bits. Intuitively, the reason for this reduction is that the aux_t relation has been augmented with more accurate information about the anonymized prefixes. Consequently, the remaining hosts are placed into their correct subnets and the set of possible true identities is reduced substantially in some cases. This same approach can be used by data publishers to anticipate attacks, and examine various "what-if" scenarios in an

objective and quantifiable manner.

Discussion and Future Work

The techniques provided in this paper can be computationally intensive, requiring significant time to fully compute the object anonymity and conditional object anonymity. Informally, our analysis is quadratic in the number of objects being analyzed, and when applied to the JHUISI dataset the analysis completed in approximately 8 hours on a single 2.4GHz processor.

One avenue of future work lies in the creation of methods for reducing the computational expense of our analysis while maintaining its correctness. We do note, however, that network data collection and anonymization is generally an offline process performed over the course of several weeks or months, and once published, the data may remain available to the public for several years. Consequently, the time spent evaluating the privacy implications of the anonymized network data is short relative to the other steps in the process, and this evaluation pays substantial dividends to the data publisher in the form of increased confidence in the sanitization of the data. Other important areas of future work include methods for efficiently examining a wider range of relationships within the data, and the application of our analysis technique to other object types, such as web pages.

8 Related Work

Notions of anonymity similar in spirit to those developed in this paper have also been used in the database privacy literature. However, unlike in the network data setting, database attributes are typically explicitly labeled as either sensitive or non-sensitive. The most widely known definition of privacy in that setting is k-Anonymity [24], which requires that each record in the database be indistinguishable from at least $k-1$ other records, with respect to every set of potentially-identifiable non-sensitive attributes. These k indistinguishable records are then referred to as forming an equivalence class, or an anonymity set.

Machanavajjhala *et al.* [18] point out two possible attacks on k-anonymous databases that can occur when parts of the data are homogeneous with respect to the sensitive attribute(s), or when the adversary has some prior knowledge of the population statistics. To address these shortcomings in k-Anonymity, they propose the notion of l-Diversity, which requires that each equivalence class in the database have at least l "well-represented" values for each sensitive attribute. Li *et al.* [17] point out weaknesses in l-Diversity that occur when dealing with numerical data or skewed categorical data, and they in turn propose the notion of t-Closeness as a more robust definition of privacy. They define an equivalence class to have t-Closeness if the distance between the distribution of a sensitive attribute within the class and the distribution of the sensitive attribute in the whole database is below some threshold t.

Unfortunately, the unique nature of network data makes it difficult to apply these notions directly to packet traces or NetFlow logs. One of the main challenges when dealing with network data is that it can be very difficult to know *a priori* which fields should be considered sensitive and which of the non-sensitive fields could be potentially identifying. Moreover, the most sensitive pieces of information, such as hosts' or users' behaviors, are often not well described by single records.

Other work has focused on assessing the privacy provided by various kinds of anonymization schemes. Distantly related is the work of Diaz *et al.* [11] and Serjantov and Danezis [25], which uses metrics based on information entropy to measure the connection-level anonymity provided to message senders in anonymizing mix networks. Lebanon *et al.* [16] apply statistical decision theory to assess the risk to privacy associated with the publication of anonymized databases. Similar to this work, the approach of Lebanon *et al.* allows for the modeling of the adversary's auxiliary information, although such information must be configured *manually* by the data publisher.

Concurrently with our work, Ribeiro *et al.* [23] proposed a new technique for undermining the privacy of enterprise network traces anonymized with prefix-preserving anonymization. Like the approach presented in this paper, their work also suggests methods for quickly analyzing network data for hosts that are vulnerable to their proposed attack.

9 Conclusion

Since the continued availability of anonymized network data relies heavily on the successful application of anonymization techniques—and the data publisher's confidence in the efficacy of those techniques—we believe that the analysis methods in this paper provide several tangible benefits. Specifically, we present the first methods, of which we are aware, for analyzing the anonymity of network data. Using our analytical techniques, we show how data publishers can make informed decisions about the appropriate use of anonymization tools and the publication of anonymized data, thereby gaining confidence in the privacy of that data. We further demonstrate the utility of our techniques by showing their use in quantifying the anonymity of host objects, objectively comparing datasets and anonymization techniques, and examining the effects of deanonymization. It is our hope that the methodology presented in this paper, along with the continued evolution of anonymization techniques and public data repositories, will further encourage the sharing of anonymized network data.

Acknowledgments

The authors would like to thank Michael Bailey for his insightful comments on an early draft of this work. This work was funded in part by NSF grants CNS-0546350 and CNS-0430338.

References

[1] J. Bethencourt, J. Franklin, and M. Vernon. Mapping Internet Sensors With Probe Response Attacks. In *Proceedings of the* 14^{th} *USENIX Security Symposium*, pages 193–208, 2005.

[2] T. Brekne and A. Årnes. Circumventing IP-Address Pseudonymization. In *Proceedings of the* 3^{rd} *IASTED International Conference on Communications and Computer Networks*, pages 43–48, October 2005.

[3] T. Brekne, A. Årnes, and A. Øslebø. Anonymization of IP Traffic Monitoring Data – Attacks on Two Prefix-preserving Anonymization Schemes and Some Proposed Remedies. In *Proceedings of the Workshop on Privacy Enhancing Technologies*, pages 179–196, May 2005.

[4] S. Chawla, C. Dwork, F. McSherry, and K. Talwar. On the Utility of Privacy-Preserving Histograms. In *Proceedings of the* 21^{st} *Conference on Uncertainty in Artificial Intelligence*, 2005.

[5] S. F. Chen and J. T. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Speech and Language*, 13:359–393, 1999.

[6] Cisco IOS NetFlow. `http://www.cisco.com/go/netflow`.

[7] S. Coull, M. Collins, C. Wright, F. Monrose, and M. K. Reiter. On Web Browsing Privacy in Anonymized NetFlows. In *Proceedings of the* 16^{th} *USENIX Security Symposium*, pages 339–352, August 2007.

[8] S. Coull, C. Wright, F. Monrose, M. Collins, and M. K. Reiter. Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces. In *Proceedings of the* 14^{th} *Annual Network and Distributed System Security Symposium*, pages 35–47, February 2007.

[9] T. Cover, J. Thomas, and M. Burns. *Elements of Information Theory, Vol. 1, (revised edition)*. Wiley Series in Telecommunications and Signal Processing, John Wiley & Sons, Inc., 2006.

[10] CRAWDAD: A Community Resource for Archiving Wireless Data at Dartmouth. `http://crawdad.cs.dartmouth.edu`.

[11] C. Dıaz, B. Seys, J. Claessens, and B. Preneel. Towards Measuring Anonymity. In *Proceedings of Privacy Enhacing Technologies*, pages 54–68, 2002.

[12] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our Data, Ourselves: Privacy via Distributed Noise Generation. In *Proceedings of Advances in Cryptology–EUROCRYPT*, pages 486–503, 2006.

[13] C. Dwork and K. Nissim. Privacy-Preserving Datamining on Vertically Partitioned Databases. In *Proceedings of Advances in Cryptology – CRYPTO*, pages 528–544, 2004.

[14] J. Fan, J. Xu, M. Ammar, and S. Moon. Prefix-preserving IP Address Anonymization: Measurement-based Security Evaluation and a New Cryptography-based Scheme. *Computer Networks*, 46(2):263–272, October 2004.

[15] D. Koukis, S. Antonatos, and K. Anagnostakis. On the Privacy Risks of Publishing Anonymized IP Network Traces. In *Proceedings of Communications and Multimedia Security*, pages 22–32, October 2006.

[16] G. Lebanon, M. Scannapieco, M. R. Fouad, and E. Bertino. Beyond k-Anonymity: A Decision Theoretic Framework for Assessing Privacy Risk. In *Privacy in Statistical Databases*, December 2006.

[17] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *Proceedings of the* 23^{rd} *IEEE International Conference on Data Engineering*, pages 106–115, April 2007.

[18] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-Diversity: Privacy Beyond k-Anonymity. In *Proceedings of the* 22^{nd} *IEEE International Conference on Data Engineering*, pages 24–35, April 2006.

[19] L. Øverlier, T. Brekne, and A. Årnes. Non-Expanding Transaction Specific Pseudonymization for IP Traffic Monitoring. In *Proceedings of the* 4^{th} *International Conference on Cryptology and Network Security*, pages 261–273, December 2005.

[20] R. Pang, M. Allman, V. Paxson, and J. Lee. The Devil and Packet Trace Anonymization. *ACM Computer Communication Review*, 36(1):29–38, January 2006.

[21] V. Paxson. Strategies for Sound Internet Measurement. In *Proceedings of the* 4^{th} *ACM SIGCOMM Conference on Internet Measurement*, pages 263–271, 2004.

[22] PREDICT: Protected Repository for the Defense of Infrastructure Against Cyber Threats. `http://www.predict.org`.

[23] B. Ribeiro, W. Chen, G. Miklau, and D. Towsley. Analyzing Privacy in Enterprise Packet Trace Anonymization. In *Proceedings of the* 15^{th} *Network and Distributed Systems Security Symposium, to appear*, 2008.

[24] P. Samarati and L. Sweeney. Protecting Privacy When Disclosing Information: k-Anonymity and its Enforcement Through Generalization and Suppression. Technical Report SRI-CSL-98-04, SRI Computer Science Laboratory, 1998.

[25] A. Serjantov and G. Danezis. Towards an Information Theoretic Metric for Anonymity. In *Proceedings of Privacy Enhacing Technologies*, pages 41–53, 2002.

[26] C. Shannon, D. Mooore, and K. Keys. The Internet Measurement Data Catalog. *ACM SIGCOMM Computer Communications Review*, 35(5):97–100, Oct. 2005. See `http://imdc.datcat.org/browse`.

[27] Y. Shinoda, K. Ikai, and M. Itoh. Vulnerabilities of Passive Internet Threat Monitors. In *Proceedings of the* 14^{th} *USENIX Security Symposium*, pages 209–224, 2005.

[28] A. Slagell, K. Lakkaraju, and K. Luo. FLAIM: A Multi-level Anonymization Framework for Computer and Network Logs. In *Proceedings of the* 20^{th} *USENIX Large Installation System Administration Conference*, pages 63–77, 2006.

[29] TCPdPriv. `http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html`.

[30] TCPurify. `http://irg.cs.ohiou.edu/~eblanton/tcpurify/`.