Commoner Privacy And A Study On Network Traces

Xiyue Deng and Jelena Mirkovic
USC/ISI
4676 Admiralty Way ste 1001
Marina del Rey, CA 90292
deng,sunshine@isi.edu

ABSTRACT

Differential privacy has emerged as a promising mechanism for privacy-safe data mining. One popular differential privacy mechanism allows researchers to pose queries over a dataset, and adds random noise to all output points to protect privacy. While differential privacy produces useful data in many scenarios, added noise may jeopardize utility for queries posed over small populations or over long-tailed datasets. Gehrke et al. proposed crowd-blending privacy, with random noise added only to those output points where fewer than k individuals (a configurable parameter) contribute to the point in the same manner. This approach has a lower privacy guarantee, but preserves more research utility than differential privacy.

We propose an even more liberal privacy goal—*commoner privacy*—which fuzzes (omits, aggregates or adds noise to) only those output points where an individual's contribution to this point is an outlier. By hiding outliers, our mechanism hides the presence or absence of an individual in a dataset. We propose one mechanism that achieves commoner privacy—interactive k-anonymity. We also discuss query composition and show how we can guarantee privacy via either a pre-sampling step or via query introspection. We implement interactive k-anonymity and query introspection in a system called *Patrol* for network trace processing. Our evaluation shows that commoner privacy prevents common attacks while preserving orders of magnitude higher research utility than differential privacy, and at least 9–49 times the utility of crowd-blending privacy.

CCS CONCEPTS

• **Security and privacy → Data anonymization and sanitization**; *Privacy protections*;

KEYWORDS

privacy, data sharing, network traces

ACSAC 2017, December 4–8, 2017, San Juan, PR, USA
© 2017 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
ACM ISBN 978-1-4503-5345-8/17/12...$15.00
https://doi.org/10.1145/3134600.3134644

ACM Reference format:
Xiyue Deng and Jelena Mirkovic. 2017. Commoner Privacy And A Study On Network Traces. In *Proceedings of ACSAC 2017, San Juan, PR, USA, December 4–8, 2017,* 12 pages.
https://doi.org/10.1145/3134600.3134644

1 INTRODUCTION

Privacy-safe data mining is a research topic of increasing importance. Today's communications and human interactions generate a wealth of data that can be used by researchers to mine common patterns, identify anomalies and study new phenomena in almost any field of science. Yet, such data is directly or indirectly generated by humans, whose privacy must be protected.

Differential privacy [11] has emerged as a promising privacy-protection approach, and is increasingly being adopted by industry.

Differential privacy allows for *aggregate* calculations over datasets, while protecting privacy of individuals and small populations [11]. Queries are submitted by researchers to the system hosting the dataset. The system runs the queries and returns noisy outputs to researchers. One popular mechanism to achieve differential privacy—the Laplace mechanism—adds random noise to *all* the points of the output, and several approaches exist to reduce the magnitude of this noise (e.g., [28]). Differential privacy yields very accurate calculations over large and well-balanced datasets, and holds under a composition of queries. But differential privacy may significantly reduce research utility of outputs, when queries are ran on long-tailed or large-value-range datasets. In these cases large noise must be added to achieve privacy guarantees and that drives the research utility down.

Crowd-blending privacy [14] was proposed as an alternative. It offers a lower privacy guarantee than differential privacy, but preserves more of the research utility. An individual I is said to *blend* in a crowd if her contributions to the outputs of some query f run on the dataset are identical to the contributions of $k-1$ other individuals. For example, if we had data about customer visits to a store and wanted to calculate a histogram of visits per month (e.g., y customers made x visits), an individual I that visited a given store 10 times in a month would blend if there were at least $k-1$ other individuals who also made 10 visits in a month. The parameter k controls the level of privacy achieved, i.e. the size of the crowd. Crowd-blending privacy releases outputs over well-blended populations without any noise, and adds random noise to (or omits) other outputs.

Our first contribution is to propose a more liberal privacy definition — *commoner privacy*, which achieves significantly higher research utility than crowd-blending privacy for select queries and datasets, at the cost of a slightly lower privacy protection. An individual I is said to *blend* in a crowd of size k if she contributes to some output with at least $k-1$ other individuals, and her contributions are not outliers among contributions of the others in that group of individuals. Commoner privacy fuzzes (adds noise, aggregates or omits) contributions of individuals that do not blend, and leaves other contributions unchanged. Therein lies a key difference between crowd-blending and commoner privacy: commoner privacy considers that an individual I blends in a group of size k if its contributions are *not outliers* among contributions of others in the group, while crowd-blending privacy requires these contributions to be the *same*. Thus commoner privacy offers lower protections to individuals for some queries, at the benefit of higher research utility. We discuss in Section 2.1 some domains where this privacy loss may be acceptable in favor of higher utility. In our evaluation in Section 5 we qualitatively and quantitatively evaluate privacy and utility of differential, crowd-blending, and commoner privacy. We find that commoner privacy prevents common attacks, while preserving orders of magnitude higher research utility than differential privacy, and at 9–49 times the utility of crowd-blending privacy.

As our second contribution we propose one mechanism to achieve commoner privacy — *interactive k-anonymity* . Privacy checks are applied just before query outputs are released to the user. Every output point with contributions by k or more identities is checked for outliers, and those outliers are removed. After this, if the contributor set contains at least k identities, the output point is released unchanged. Other points are fuzzed to protect privacy.

Our third contribution is to show how commoner privacy can hold under query composition. Like Gehrke et al. [14], we could use pre-sampling of individuals prior to processing the query to protect query compositions. However, this loses input data and thus research utility, even when a composition of queries does not jeopardize privacy. Instead, we propose *query introspection* —careful record-keeping of identities and their contributions to outputs for all the queries posed by a given researcher. At run time we perform automated checks of combinations of past queries and the current query to detect tracker attacks. If the check detects a query composition that may jeopardize privacy, the current query is either rejected or some of its output points are fuzzed to protect privacy.

Throughout the paper we use network traces as our motivating example, and we implement a system for network trace processing with commoner privacy, called *Patrol*. Network traces contain a record for each packet seen on the network, which includes source and destination IP addresses that are considered personal and identifying information (PII), and can be linked to human users using the associated machine. Network traces can thus reveal sensitive information about a user's communication patterns (e.g., "a user John Smith visited a pornographic site while at work"), or they can reveal data about vulnerabilities on a user's machine that can be exploited in cyberattacks (e.g., "machine with IP address 1.2.3.4 has ports 80 and 22 open"). We show that commoner privacy provides sufficient protection against known attacks on user privacy in network trace data. While we use network traces to illustrate our findings, commoner privacy is general enough to apply to any dataset. Our current implementation supports network traces and can be extended to support more formats in the future.

2 DIFFERENTIAL AND CROWD-BLENDING PRIVACY

In this section we provide a brief overview of two popular privacy definitions and of the mechanisms that achieve them: differential and crowd-blending privacy. Both of these mechanisms and our commoner privacy can be applied in a setting where original data resides with the data provider. Users (researchers) are allowed to pose queries over data and receive outputs of these queries. The privacy mechanisms modify these outputs to protect the privacy of individuals that have contributed the data. It is the nature of this modification that may lead to stronger or weaker privacy guarantees, and lower or higher research utility of the outputs.

Identities vs. Records. Privacy protections should always be applied over individuals and not over records in the dataset. This distinction is important for datasets where an individual may contribute multiple records; e.g., an individual may submit multiple movie reviews or make multiple purchases or hospital visits. In such cases, a highly contributing individual may be exposed if privacy protections are applied over records instead of over identities. In this overview, we revise some existing definitions, where necessary, to make it clear which operations occur over individuals and which over dataset records. To this effect, we introduce two functions to map between individuals and their records: (1) $Id(x_i)$ returns the identity or identities whose data is in record(s) x_i, or whose data has been used to calculate the output data point x_i, and (2) $rec(Y)$ returns all the records associated with identity or identities Y.

Differential privacy [10] ensures that an adversary can learn, roughly, the same data about an individual, regardless of the individual's participation in the dataset. This is achieved through a randomized algorithm *San*.

Definition 2.1. **Differential privacy [11].** A randomized algorithm *San* is said to be (ϵ, δ)-differentially private if for all $S \in Range(San)$ and for all $x, y \in \mathcal{D}$ such that $||Id(x) - Id(y)||_1 \leq 1$ it holds:

$$Pr[San(x) \in S] \leq exp(\epsilon) \cdot Pr[San(y) \in S] + \delta \qquad (1)$$

Differential Privacy Mechanism. One popular mechanism to achieve (ϵ, 0)-differential privacy is the Laplace mechanism [11]. This approach adds random noise to each output data point. The noise is drawn from the Laplace distribution with parameter $\Delta f/\epsilon$, where Δf is the *global sensitivity* of a desired query f to perturbation of inputs (largest possible change in outputs should one individual join or leave the set), and ϵ is the privacy parameter, usually set to 0.1 [24]. In datasets with unbalanced contributions by individuals, global sensitivity may be large, and thus a large amount of noise must be added to all output points. This loses much of the research utility. Nissim et al. propose a sample-and-aggregate framework as a way to lower the noise by tailoring it to the dataset [28]. However, noise is still added to all output points.

Crowd-blending privacy [14] relaxes the notion of differential privacy, allowing an attacker to learn something about an individual, if this feature is sufficiently common to be considered not sensitive. Crowd-blending privacy employs the notion of ϵ-blending.

Definition 2.2. **ϵ-blending [14].** An individual x ϵ-blends with individual y in dataset $\mathcal{D}$ with respect to privacy-preserving mechanism San if it holds that for $\mathcal{D}'$, where x's data is replaced by y's data:

$$Pr[San(\mathcal{D}) \in S] \leq exp(\epsilon) \cdot Pr[San(\mathcal{D}') \in S] \qquad (2)$$

Definition 2.3. **Crowd-blending privacy [14].** A mechanism San is (k, ϵ)-crowd-blending private if for every dataset $\mathcal{D}$ and every individual $t \in \mathcal{D}$, either t ϵ-blends in a crowd of k individuals in $\mathcal{D}$, or the mechanism San ignores it.

Crowd-Blending Privacy Mechanism. Gehrke et al. [14] propose a specific crowd-blending mechanism for histogram queries over identities, which can be extended into a general mechanism as follows. The crowd-blending privacy mechanism releases unchanged those output points where it holds that a *crowd* of at least k individuals has the same contributions to the output point. Contributions from the individuals that do not blend in the crowd are omitted, or the output points that contain them are fuzzed.

Crowd-blending privacy does not compose under multiple queries; i.e.. it is possible to design t queries that are each crowd-blending private, but their combination is not [14]. Gehrke et al. propose a pre-sampling step, where an algorithm chooses at random which individuals' data will be used for each query. They prove that crowd-blending privacy with pre-sampling holds under query composition.

2.1 Motivation for Commoner Privacy

We now advance the need for commoner privacy by discussing two cases where differential and crowd-blending privacy do not perform well. In the first case, counting queries are run over records in a dataset, where a single individual could contribute multiple records (e.g., a network trace, hospital visit logs, movie reviews). For example, such a query could ask for the number of packets sent from the Web server port in a network trace. In the second case, aggregation queries (average, sum, product, etc.) are run over data fields in any dataset. For example, such a query could ask for the sum of all the savings on accounts of a given type, in a given bank. Depending on the data that is being counted or aggregated, two phenomena may occur: (1) *Long tail.* Some individuals may contribute a large value to some output points, e.g., an active server may generate so much traffic in the trace that it dominates the count of packets or bytes in the entire dataset. This will make global sensitivity large, leading to a large amount of noise added to all output points by differential privacy, and thus low research utility. (2) *Large value range.* If the field has many possible values, many individuals may have similar but not the same contributions to the output; e.g., there may be many possible values for the amount on a savings account. In this case crowd-blending privacy will fuzz many output points, leading to low research utility.

Day	M	T	W	R	F	Sa	Su	M	T	W	R	F	Sa	Su
Water	30	130	28	31	33	51	48	44	30	37	516	31	58	54
Bread	10	12	19	6	10	14	15	13	10	18	12	14	12	11
Milk	8	7	12	11	22	8	14	10	10	12	10	12	21	6

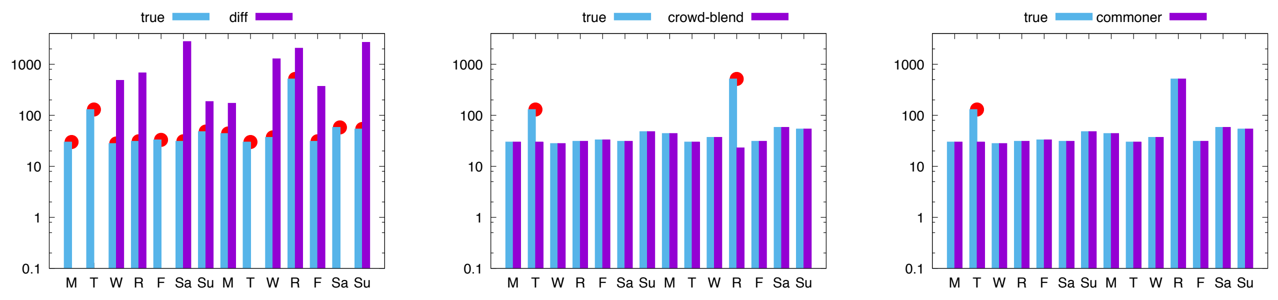
Table 1: Daily sales per product from a small shop

These cases call for commoner privacy; i.e., we need a privacy definition that allows a release of aggregate outputs as long as an individual's contributions to these outputs are similar to the contributions of a sufficient number of other individuals—so that neither can be identified from the aggregate *by a realistic adversary.*

We now use an example to further illustrate the need for commoner privacy. Imagine that a small store releases records with a product and quantity purchased for each day over two weeks, illustrated in Table 1. Most products will be purchased by people in small quantities. For simplicity, assume that there are between 30 and 60 customers each day, and each customer buys exactly 1 or 0 of any given product at any given day, with two exceptions. On the first Tuesday, customer A buys 100 water bottles, e.g., because he is throwing a large party. On the second Thursday, customers B, C, D, E and F buy 80, 99, 101, 103 and 110 water bottles, respectively. A researcher wants to run a counting query on the data to learn the count of each type of product bought for each day over the two weeks.

Let us first observe the first week's worth of data. Customer A's data skews the total count of water bottles on Tuesday from 30–60 to 130. Thus releasing this count may suffice to detect A's presence in the dataset. To hide this behavior, differential privacy would have to add a noise proportional to the highest number of water bottles that a customer can buy at any day at that store. In the worst case this would be the entire store's inventory of water, and at the best case it would be a value close to 100. Moreover, this noise will be added not just for the problematic day but for all days. The noise would be too large, and it would drown out any trend in the data, e.g., "people buy more water on weekends." Outputs protected by differential privacy (assuming $\Delta f = 100$ and $\epsilon = 0.1$) are shown in Figure 1(a). Note the large difference between true and outputted values (y axis is in the log scale), and note that all points have been fuzzed (denoted by red circles in the figure).

Crowd-blending privacy (assuming $k \leq 30$), on the other hand, would remove A's data and release the count of 30



(a) Diff privacy (b) Crowd-blending (c) Commoner

Figure 1: Water bottles sold per day on logarithmic scale: counts protected by differential, crowd-blending and commoner privacy. Cyan bars show the true values and purple bars show the outputs of a privacy mechanism. Red circles denote data points that had to be fuzzed by the mechanism to protect privacy.

bottles for that first Tuesday and true counts for any other day in the first week. This is shown in Figure 1(b). Commoner privacy would do the same on this example, shown in Figure 1(c). Thus, commoner and crowd-blending privacy will provide identical outputs on the first week of data. In both cases, the outputs will have much higher research utility than those produced by differential privacy.

Now let us observe the second week's worth of data. Differential privacy will again add the same large amount of noise to all data points. Crowd-blending privacy with $k = 5$ will remove data for customers B, C, D, E and F from Thursday, because they purchased different quantities of water (80, 99, 101, 103, 110). But this may be too restrictive, and it may unnecessarily reduce research value. Note that ordered values are so close together that, while each is unique, they probably cannot be gleaned from the sum. Commoner privacy will help us identify cases when values are so close together that they do not pose the risk of re-identification. Figures 1(b) and 1(c) illustrate how the data would be processed by crowd-blending ($k = 5$) and commoner privacy ($k = 5$). Note that outputted values for both mechanisms are close or identical to the true values. Also note that commoner privacy provides higher research utility on this dataset and for this query, because it only fuzzes one output point, while crowd-blending privacy fuzzes two.

3 COMMONER PRIVACY

We start by defining two adversary models and positioning commoner privacy with respect to differential and crowd-blending privacy. We then define commoner privacy, and a mechanism that achieves it.

3.1 Adversary Model

We consider two types of adversaries.

All-but-one adversary can observe all but one individual's records in the dataset, or in a group of records. This adversary is not realistic, as there is nothing that prevents it from learning the remaining individual's records in the same manner that he learned the others.

Interactive adversary learns about a dataset and its participants by posing queries and analyzing their outputs. The adversary can also learn some information about an individual from auxiliary channels that he can try to cross-correlate with the system outputs. Auxiliary information can be of two kinds: (1) adversary can use some unique feature of the individual to establish presence or absence of an individual in the dataset and to possibly learn new information about the individual; or (2) certainty that the individual's data is in the released dataset. The interactive adversary is weaker than the all-but-one adversary, but it is realistic. Published attacks on various privacy mechanisms all use interactive adversaries [4, 5, 12, 29, 32].

Differential privacy protects against all adversaries—the interactive adversary and the stronger all-but-one adversary. Commoner privacy and crowd-blending protect an individual against an interactive adversary but not against the all-but-one adversary. The all-but-one adversary can learn those features of an individual that are sufficiently common, e.g., those where the individual blends with a sufficiently large crowd.

Gehrke et al. prove [14] that crowd-blending can achieve differential privacy (and protect against the all-but-one adversary) if data is sampled at random prior to running a query. Intuitively, this random sampling eliminates the certainty that the adversary may have about an individual's data being in the dataset. Similarly, pre-sampling could be applied prior to commoner privacy to achieve resilience against the all-but-one adversary. However, pre-sampling loses research utility, and we leave research into quantifying this trade-off in case of commoner privacy for future work. In this paper we focus on commoner privacy without pre-sampling.

3.2 Privacy Goal

We assume that an individual I whose data is in the dataset has the following privacy goals: (1) an adversary should not be able to learn if I participated in the dataset, (2) an adversary who knows that I participated in the dataset should not be able to learn the exact value or the range of values for any field v in I's records, except when this value (or range) is shared by at least $k-1$ other individuals. It is this exception that lowers the privacy guarantee from differential privacy. Both crowd-blending and commoner privacy assume that such exception is acceptable to data contributors.

Under which circumstances would an individual agree to this exception? We believe that this would happen when data itself is not sensitive. In our example of the store releasing its sales data, B may not mind if the adversary learns that he bought a lot of water on Thursday, because this behavior is shared by four other customers, and thus it is common. However, if these were health diagnoses instead of prchased products, B would not want it known which disease he had, even if that disease were very common. In that case a pre-sampling step would be needed to remove the adversary's certainty that B's data is in the dataset.

3.3 Commoner Privacy Definition

We first define k-blending, which establishes a measure of closeness between individuals in a dataset for a given query. We then define commoner privacy using k-blending. Finally, we discuss how to identify outliers.

Definition 3.1. **k-blending.** An individual x k-blends with individuals $y_1, ..., y_{k-1}$ in dataset $\mathcal{D}$ with respect to privacy-preserving mechanism San if San(x) is not an outlier in the set of values $\{San(\text{x}),\ San(y_1),\ ...\ ,\ San(y_{k-1})\}$.

Definition 3.2. **Commoner privacy.** A mechanism San is k-commoner private if for every dataset $\mathcal{D}$ and every individual $t \in \mathcal{D}$, either t k-blends with individuals in $\mathcal{D}$ or the mechanism San ignores it.

Commoner privacy is a strictly lower goal than crowd-blending privacy, since an individual that ϵ-blends with $k-1$ other individuals also k-blends with them, while the opposite holds only when the mechanism San has identical outputs for all k individuals.

THEOREM 3.3. *Differential privacy $\rightarrow$ crowd-blending privacy $\rightarrow$ commoner privacy.*

PROOF. In [14] Gehrke et al. prove the first part of this relation—that differential privacy implies crowd-blending privacy. We now prove the second part—that crowd-blending privacy implies commoner privacy. This follows directly from the definitions of crowd-blending and commoner privacy. When t ϵ-blends with $k-1$ other individuals in $\mathcal{D}$, this means that their records (with respect to San) are so similar that they are interchangeable. Thus t is not an outlier in the group, and it k-blends within that group. □

Outlier detection. There are several outlier detection algorithms in research literature [15, 21]. We envision that the data provider will select the one that best fits her data. We explored two possible implementations in Section 5: (STDEV)—the commonly used approach where a value is an outlier in a group if it lies outside $avg \pm 3 \cdot stdev$ range, where avg is the average, and $stdev$ is the standard deviation calculated over the entire group; and (MAD)—the approach proposed in [21] where a value is an outlier in a group if it lies outside $med \pm 3 \cdot b \cdot med(abs(dev))$, where med is the median and $med(abs(dev))$ is the median absolute deviation, calculated over all members of the group. The parameter b is set to 1.4826 as the constant scale factor for normal distribution.

3.4 Commoner Privacy Mechanism

We now define one mechanism to achieve commoner privacy—interactive k-anonymity.

Definition 3.4. **Interactive k-anonymity.** Given any function f, interactive k-anonymity is defined as:

$$M_k(x, f(\cdot), k) = \begin{cases} f(rec(Id(x_i)) & Id(x_i)\ k\text{-blends in } x \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

It is obvious that interactive k-anonymity achieves k-commoner privacy, as it returns non-zero outputs only for individuals that k-blend in the dataset. Further, one could release a noisy output, instead of zero, for those individuals that do not k-blend or aggregate them into a larger group. Individuals that do not k-blend enjoy differential privacy, while those that blend enjoy the lower, commoner privacy.

With regard to the expected research utility, k-commoner privacy preserves higher utility than (k, ϵ)-crowd-blending privacy, because it does not fuzz records of individuals that k-blend in the dataset. Since k-blending does not imply ϵ-blending, but ϵ-blending implies k-blending, the number of individuals ignored by commoner privacy will be smaller than or equal to the number of individuals ignored by crowd-blending privacy, for the same parameter k. This means that commoner privacy will preserve more research utility than crowd-blending privacy.

3.5 Discussion

We now briefly discuss some remaining details about commoner privacy.

Setting of the parameter k. The parameter k controls the size of the group, whose features are considered sufficiently common to be revealed to a user. A data provider could set this parameter based on his/her understanding of the dataset, the sensitivity of information contained therein, and the common uses of the data. Another model for setting of the parameter k would have each individual I, which contributes to the dataset, set its own acceptable value – k_I. Figure 2 illustrates how outlier removal would work in this second case for each data point x.

Use Scenarios. We see commoner privacy as a good alternative to differential or crowd-blending privacy for datasets with a long tail or with a large range of feature values, where it would be acceptable that queries leak some data about common features or behaviors of individuals in the dataset. Network traces and system logs are a good example of this type of dataset. These datasets usually contain very diverse, rich information about network traffic and network hosts. The usual concern when allowing researcher access to these datasets is to preserve anonymity of source and destination IP addresses, and this is currently achieved through sanitization [35]. Yet, sanitization can always be broken with access to auxiliary data [11]. On the other hand, allowing differentially

```
$k \leftarrow max(k_I), I \in Id(x)$
while ——Id(x)—— $\geq k$ do
  for all $i \in Id(x)$ do
    if $f(rec(i))$ is an outlier then
      remove $rec(i)$
      $k \leftarrow max(k_I), I \in Id(x)$
    end if
  end for
end while
$x \leftarrow$ calculate from remaining data
```

Figure 2: Outlier removal algorithm for a data point x

private access to these datasets may lose too much utility since network traffic and host behaviors usually exhibit long tails. Commoner privacy offers stronger privacy guarantees than sanitization and better utility than differential privacy. The utility gain comes at the risk of the adversary learning some common network traffic and host features or behaviors. This may be acceptable to data providers, as they can set the appropriate k value to control the level of privacy risk to any individual host.

4 QUERY INTROSPECTION

Composition of queries that all satisfy k-anonymity, crowd-blending, or commoner privacy may jeopardize the privacy of individuals due to the existence of trackers. Tracker attacks have been investigated in depth with regard to k-anonymity [7–9, 31]. The general idea is to ask a set of queries, such that outputs of each query meet the privacy criteria (k-anonymity, crowd-blending or commoner privacy), but the outputs of their combination do not.

Gehrke et al. prove that a pre-sampling step prior to the processing of each query can make crowd-blending hold under query composition. We expect that a similar proof could be devised for commoner privacy and pre-sampling, but leave this for future work. In the following we describe our approach to protecting query composition through query introspection, and we prove that it can detect all trackers from prior work [8].

Let q be a query run on dataset $\mathcal{D}$. The *field set* for this query, denoted by $F(q)$, is a set of data fields that are used in the query. The query may be arbitrarily complex and composed of multiple SQL statements, such as SELECT, GROUP BY, arithmetic and logical operations, etc., but it ends with a single OUTPUT statement, which produces either a single numerical output point or a sequence of (x, y) output points, where x is a string or a number, and y is a number.

When processing each query, we keep track of identities that relate to each variable or group created in the query, and their contributions to the points in the output. These contributions start as values of some data field (e.g., for network traces this could be packet size, time, port number) or 1 (for record-counting queries) and are transformed as query statements are processed. We call this a set of tuples, where each tuple contains some data point, and the set of identities and their contributions to this data point a "contributing identity set" (CIS).

Table 2 shows how the processing of some common SQL statements affects the contributing identity sets of the related variables and groups. We further keep track of data fields associated with each group and each variable, and we keep track of all queries $Q = \{q_1, ..., q_n\}$ posed by a given user for a given dataset $\mathcal{D}$. Only query specifications are kept, not the query outputs. When a new query q_{n+1} arrives, we perform the following steps to check for tracker attacks:

Step 1: Identify related queries. For each field f in field set $F(q_{n+1})$, identify $Q_r(q_{n+1})$—a set of *related* queries from Q, such that for each query q_i in this set, its field set contains f.

Step 2: Create combinations of related queries. Create all possible subsets of $Q_r(q_{n+1})$ and add to them q_{n+1}, including an empty subset. For each such subset S=\{q_{i1}, ... , q_{ib} \} perform *cross-query checking* described next.

Step 3: Cross-query checking. Cross query checking is done by rerunning each query q_s from set S on dataset $\mathcal{D}$, and producing output data points $d(q_s)$. For simplicity, assume that each record has only one identity related to it. In case of multiple identities, the calculations described next would be performed for each role separately (e.g., in a case of network traces we would perform separate checks for sets of source IP addresses, and separate for sets of destination IP addresses).

We process the associated contributing identity set (CIS) for each output data point, and we calculate the following sets: (1) set V containing values for each field f that contribute to the x value of each output data point, and (2) CIS for each output data point. We then identify a set of output data points in the past queries, $ODPs$, that have common values in their set V with some data points for q_{n+1}. For these data points we calculate (1) set differences of the CIS of each of the queries in ODP and of CIS of q_{n+1}; (2) set differences of the sets A and B, where A is the union of the CISs in ODP and B is the CIS of query q_{n+1}; (3) set differences of C and D, where C is the union of the CISs in ODP and the CIS of q_{n+1}, and D is the CIS for the whole dataset. Each set difference calculation is performed twice—once for the original placement of minuend and subtrahend, and once when their places are switched. For each resulting set, we perform a k-blending check on each data point. If any of these checks fails, the query q_{n+1} is either rejected or the data point that did not pass the k-blending check is fuzzed. This is further illustrated in Figure 3, with data point for port 443 failing the check and being omitted or merged into a larger range of ports.

The downside of query introspection is that the number of checks required for each new query grows as a power of 2, and may be prohibitive as users pose more queries. However, the number of checks grows with the number of *related* queries and not with the number of *all* queries, and

Operation	Transformation
Group	Each group's CIS is the union of CISs of items that were grouped
Keep	Some groups/records are dropped and their CISs are dropped too
Arithmetic op.	The left hand side's CIS becomes the union of CIS of items on the right hand side
Conditions	If condition is met, for each statement within TRUE branch, the CISs of the variables from the conditional statement are added to the left hand side's CIS. Similarly when condition is not met, CISs within FALSE branch are extended to include the CISs from the condition.

Table 2: Transformations of the contributing identity set (CIS) for sample statements

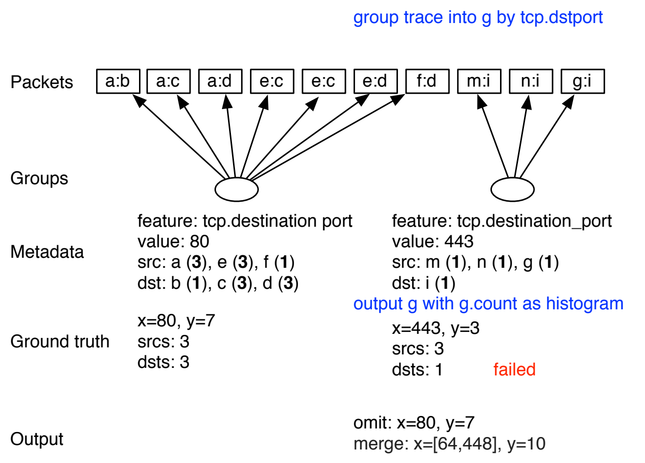


Figure 3: Illustration of per-query accounting and k-anonymity check for $k = 2$. Trol queries are shown in blue.

checks are only performed on some output data points that have common values from the set V. A few optimizations are possible to keep the size of the related query set small. First, data providers could provide synthetic datasets that are accessible without commoner privacy restriction. This would allow researchers to practice query writing until they "get it right," without affecting their future queries. Second, we can store CISs of the queries along with them so that queries do not need to be rerun. This trades the processing cost for the storage cost. We implement this approach and evaluate it in Section 5. The third possible optimization would reduce the number of tracker checks performed. When new queries are identical to old queries that were run on the same dataset tracker, checks can be skipped without loss of privacy.

4.1 Proof of Tracker Detection

Here we formally prove that query introspection protects against known tracker attacks. For completeness, we repeat tracker definitions from [8].

Let $n = ||Id(\mathcal{D})||$ be the count of individuals contributing to the dataset, and let k be the privacy parameter for the chosen privacy definition (k-anonymity, crowd-blending privacy or commoner privacy). A query is said to be *answerable* if it satisfies the chosen privacy definition.

THEOREM 4.1 (INDIVIDUAL TRACKER [8]). *Let $C = A \cap B$ be a formula identifying individual I, and suppose $T = A \cap \bar{B}$ is I's tracker where A, T are both answerable. With three answerable queries, calculate: (eq1) $COUNT(C) = COUNT(A) - COUNT(T)$, (eq2) $COUNT(C \cap a) = COUNT((T \cup A) \cap a) - COUNT(T)$. If $COUNT(C \cap a) = 0$, I does not have a characteristic a (negative compromise). If $COUNT(C \cap a) = COUNT(C)$, I has a characteristic a (positive compromise). If $COUNT(C) = 1$, arbitrary statistics about I can be computed from $q(C) = q(A) - q(T)$.*

THEOREM 4.2. *Query introspection protects against individual tracker.*

PROOF. Individual tracker relies on queries A and T to be answerable. When the second query is posed, our cross-query checking will test if combinations (union, set difference, intersection) of A and T are also answerable. It will thus detect the individual tracker from equation (eq1), which relies on set difference, and it will refuse to answer the second query. Similarly, we can prove that if both queries from equation (eq2) are posed, in any order, the second query will be rejected by our cross-query checking. □

THEOREM 4.3 (GENERAL TRACKER [8]). *Let T be a characteristic formula whose query size is in the restricted subrange $[2k, n-2k]$, that is: $2k \leq COUNT(T) \leq n-2k$, where $q(T)$ is always answerable due to its range is within the allowed query range $[k, n-k]$, and $k \leq n/4$. The value of any unanswerable query $q(C)$ can be computed as follows using any general tracker T. First calculate: $Q = q(T) + q(\bar{T})$. If $COUNT(C) < k$, the queries on the right-hand side of the following equation are answerable: (eq3) $q(C) = q(C \cup T) + q(C \cup \bar{T}) - Q$. Otherwise $COUNT(C) > n-k$ and the queries on the right-hand side of the following equation are answerable: (eq4) $q(C) = 2Q - q(\bar{C} \cup T) - q(\bar{C} \cup \bar{T})$.*

THEOREM 4.4. *Query introspection protects against general tracker.*

PROOF. We have $q(T) + q(\bar{T}) = Q = q(C) + q(\bar{C})$. Therefore, for equation (eq3): $q(C \cup T) + q(C \cup \bar{T}) = q(C) + Q$ Let $COUNT(combined) = COUNT(C \cup T) + COUNT(C \cup \bar{T})$. Cross-query checking will detect that $COUNT(C \cup T) + COUNT(C \cup \bar{T}) > COUNT(Q)$, and will continue checking $COUNT(combined) - COUNT(Q) = COUNT(C) < k$. This last check will result in the last query posed by the user being refused. Similarly, we can find a combination of queries in equation (eq4) that will be rejected by cross-query checking. □

THEOREM 4.5 (DOUBLE TRACKER [8]). *A double tracker is a pair of characteristic formulas (T, U) for which $X_T \subseteq X_U$, $k \leq COUNT(T) \leq n-2k$, and $2k \leq COUNT(U) \leq n-k$. Obviously $k \leq n/3$ if these conditions are to be met. The value of any unanswerable query $q(C)$ can be computed as following using any double tracker (T, U). If $COUNT(C) < k$, all queries on the right-hand side of the following equation are answerable: (eq5) $q(C) = q(U) + q(C \cup T) - q(T) - q(\overline{C \cap T} \cap U)$. Otherwise $COUNT(C) > n-k$ and all queries on the right-hand side of the* *following equation are answerable: (eq6)* $q(C) = q(\bar{U}) - q(\bar{C} \cup T) + q(T) + q(\overline{\bar{C} \cap T} \cap U)$.

THEOREM 4.6. *Query introspection protects against double tracker.*

PROOF. We can assume $C \cap T \neq \emptyset, C \cap U \neq \emptyset, T \cap U \neq \emptyset$, otherwise T and U are not useful for double tracker. For equation (eq5) consider the queries needed to calculate $q(C)$. Considering $q(U)$ and $q(\overline{C \cap T} \cap U)$, we have $q(U) - q(\overline{C \cap T} \cap U) = q(C \cap T) \subset q(C)$. Therefore, $COUNT(U) - COUNT(\overline{C \cap T} \cap U) < COUNT(C) = k$. This combination will be tested by cross-query checking and the last query posed by the user will be refused. Similarly, we can find a combination of queries in equation (eq6) that will be rejected by cross-query checking. □

5 EVALUATING COMMONER PRIVACY

It is difficult to compare utility and privacy of commoner privacy against those of crowd-blending and differential privacy because these measures depend on the exact queries asked and on the dataset's composition. In this section we first qualitatively discuss these measures using examples of *counting queries*, which are powerful primitives for many standard data-mining tasks [11]. We then quantitatively illustrate the utility gain of commoner privacy over crowd-blending and differential privacy using sample queries over network traces. Finally, we quantify the overhead of query introspection for sample compositions of queries on the same dataset.

5.1 Qualitative Discussion

We first consider *counting queries over identities that meet some condition C.* Utility-wise, differential privacy will release a noisy count, but the amplitude of the noise will be low, as the global sensitivity of the query is 1. Commoner and crowd-blending privacy will both release a true count if there are k or more identities that meet the condition C, and will release noisy count or zero otherwise. Privacy-wise, differential privacy will not allow an attacker to learn if an individual I meets the condition C. If the attacker knows that there are $m \geq k - 1$ identities in the dataset that meet condition C, prior to participation of an individual I, commoner and crowd-blending privacy will allow him to learn if I also meets C or not. *Thus, for counting queries over identities, commoner and crowd-blending privacy yield the same outcomes, and differential privacy has a slightly lower utility and a much stronger privacy guarantee.*

We next consider *counting queries over records that meet condition C.* Utility-wise, differential privacy will release a noisy count; in the case of long-tailed datasets, the noise's amplitude may be large. Assume that for a given data point there are n identities with records $rec(I_1), .., rec(I_n)$ that meet condition C. Crowd-blending privacy will release a true count if $\exists m | n \geq m \geq k$ and $||rec(I_1)|| = ||rec(I_2)|| = ... = ||rec(I_m)||$, and it will release a noisy count (or zero) otherwise. Commoner privacy will release a true count if $\exists m | n \geq m \geq k$ and there are no outliers among the counts $||rec(I_1)||, .., ||rec(I_m)||$, and it will release noisy count (or zero) otherwise. Privacy-wise, differential privacy will not allow the attacker to learn any specific record or the value of a data field for any individual I. If the attacker knows that there are $m, m \geq k - 1$ identities in the dataset and knows $||rec(I_1)||, ..., ||rec(I_m)||$, crowd-blending and commoner privacy will allow him to learn $||rec(I)||$ for another individual I that blends with $I_1, ..., I_m$. *Thus, for counting queries over records, commoner privacy yields higher utility than crowd-blending privacy, and both have higher utility than differential privacy. This higher utility comes at the cost of lower privacy guarantees as the all-but-one attacker can learn information about an individual.*

Similarly, if we considered *sum queries over data fields*, the utility and privacy of commoner, crowd-blending, and differential privacy would have the same relationship as counting queries over records.

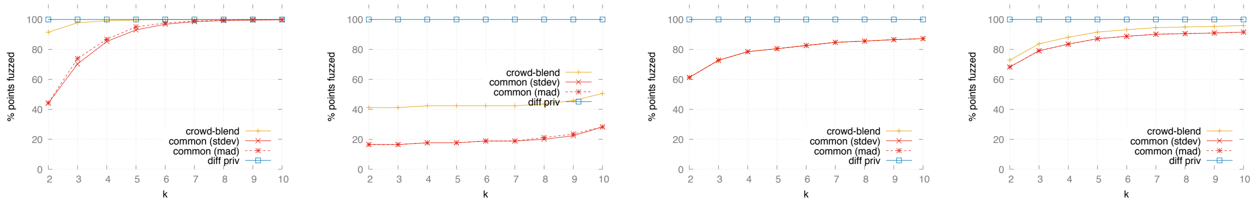
An individual I may be justly concerned that an all-but-one attacker can learn his exact contribution c_I to some aggregate output—either the count of records that meet some condition C or a value of a data field in a record. Specifically, c_I could be unique and thus a quasi-identifier; learning it may reveal presence or absence of I in the dataset. Note, however, that the all-but-one attacker is unrealistic, and we should discuss what a more realistic, interactive attacker can learn. Crowd-blending privacy will allow an interactive attacker to learn c_I only if c_I is shared by at least $k - 1$ other identities; i.e., it is not a quasi-identifier. Thus the attacker cannot learn if I is in the dataset. Commoner privacy will allow an interactive attacker to learn some approximation of c_I, only if c_I is similar to contributions of $k - 1$ other individuals. For example, an interactive attacker may learn the average salary in the company, calculated to include I's salary, only if I's salary is not an outlier. If I's salary is an outlier, the attacker only learns the average salary of other employees and cannot infer which of the following is true: (1) I is in the dataset and has a salary within the expected range, (2) I is in the dataset and his salary was omitted from calculations, (3) I is not in the dataset.

5.2 Quantifying Utility Gains

We now illustrate one use case for commoner privacy—queries on network traces—and compare utility afforded by differential privacy, crowd-blending privacy, and commoner privacy. We use a trace provided by the MAWI project [1]. The trace was collected on August 1st, 2016 and contains 15 minutes of traffic on a large US-Japan Trans-Pacific link. We use the first two million TCP packets in our evaluation. We run four common network processing queries on this data:

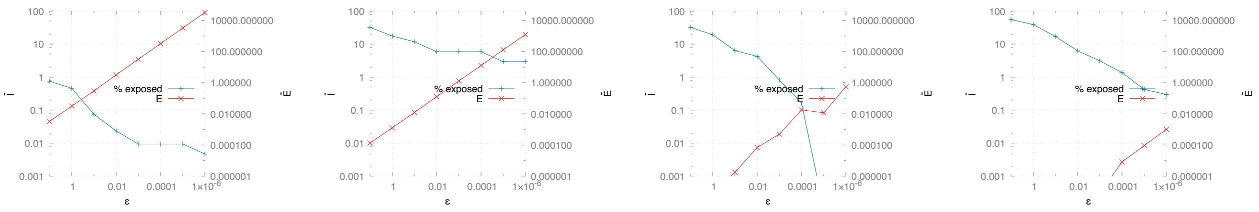
q1. Histogram of packet counts sent per source port
q2. Histogram of packet counts received per destination *service* port
q3. Total connection count in the trace
q4. Total traffic volume in the trace

All queries have high global sensitivity, as a large sender or receiver can dominate the count of packets/connections/bytes.



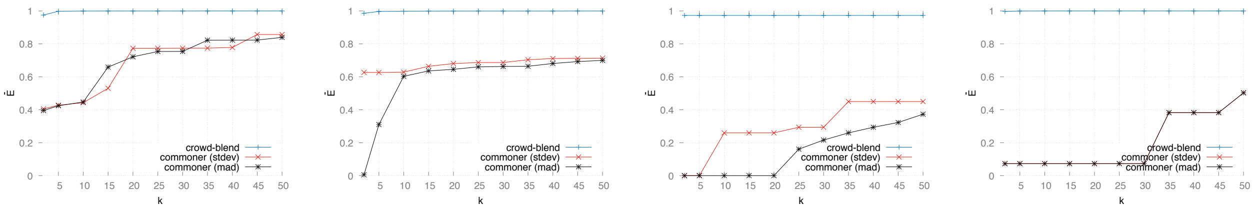
(a) #pkts per source port (b) #pkts per dst service port (c) # connections in trace (d) traffic volume in trace

Figure 4: % of output points fuzzed for four select queries on a network trace.



(a) #pkts per source port (b) #pkts per dst service port (c) # connections in trace (d) traffic volume in trace

Figure 5: Utility loss and privacy risk for differential privacy, as we vary $\Delta f/\epsilon$.



(a) #pkts per source port (b) #pkts per dst service port (c) #connections in trace (d) traffic volume in trace

Figure 6: Utility loss for differential privacy, crowd-blending privacy, and commoner privacy

As there are many ways to fuzz data points—add noise, omit or aggregate them—we first compare only the *number* of data points that will be fuzzed. If we assume that the same approach will be used for fuzzing, then the mechanism with lowest number of points fuzzed will yield the highest utility.

Figure 4 shows the percentage of output points fuzzed by differential privacy, crowd-blending privacy, and commoner privacy (with STDEV and MAD approaches for outlier detection) for *q1*–*q4*. We vary the parameter k from 2 to 10. In all cases, differential privacy fuzzes all output points, while crowd-blending and commoner privacy fuzz fewer. The actual utility gain depends on the dataset as well as the queries being asked.

For query *q1* (Fig. 4(a))—histogram of packets sent by a source port—we do not expect much similarity in hosts' behaviors, as source ports are often chosen by client applications at random, and hosts may send different amount of packets per port. For this query, crowd-blending performs just slightly better than differential privacy at $k = 2$, fuzzing 90% of points. Commoner privacy fuzzes much fewer points up until $k = 7$, with STDEV and MAD approaches having comparable performance. At $k = 2$ only 45% of points are fuzzed, at $k = 3$ 75% are fuzzed.

For query *q2* (Fig. 4(b))—histogram of packets received by a destination service port—we expect more similarity in hosts' behaviors, as destination ports are often service ports, and most hosts will have popular ports open, like port 80 and 443. Here, crowd-blending fuzzes only 40% of all points, and commoner privacy fuzzes only around 20%.

For query *q3* (Fig. 4(c))—connection count—crowd-blending and commoner privacy perform the same here, and fuzz 60–80% of all points, still providing a significant advantage over differential privacy.

Similarly, for query *q4* (Fig. 4(d))—traffic volume—-crowd-blending and commoner privacy perform similarly, and fuzz 65-95% of all points, with commoner privacy fuzzing around 5% fewer points than crowd-blending, and STDEV and MAD algorithms performing the same.

5.3 Quantifying Utility Loss and Privacy Risk

We now quantify utility loss and privacy risk of each privacy mechanism.

Utility loss. We adopt the measure of utility loss using a normalized error defined as:

$$\bar{E} = \frac{\sum_{i=1}^{N} |f_i - t_i|}{\sum_{i=1}^{N} |t_i|} \tag{4}$$

where t_i are the true outputs of a query (with no privacy protections), and f_i are the outputs generated by the privacy mechanism. This measure is a relative, cumulative difference between the true and the fuzzed data points, with lower value denoting higher research utility. It represents the fraction of the change in the outputs due to fuzzing.

Privacy risk. We adopt the measure of privacy risk by estimating the number of output data points, which can be used to identify individuals (hosts in our case). We assume that any data about an individual can become a quasi-identifier if it is unique to that individual and can be detected from the mechanism's output. We say that original data about an individual can be detected from the mechanism's output if we can reverse the fuzzing, and if the data point contains contributions of individuals that are both unique and outliers. We identify the output data points where we can reverse the fuzzing as follows. We assume that we have two sets of outputs—the true one and the fuzzed one (outputted by the privacy mechanism). We compare the values of each data point in the fuzzed output with all data points in the true output. If the fuzzed data point is closest to its true data point, we say that we can reverse the fuzzing. Our measure of privacy risk is then defined as:

$$\bar{I} = \frac{\sum_{i=1}^{N} \mathit{reverse\_and\_identifiable}(f_i)}{N} \tag{5}$$

where function *reverse_and_identifiable* returns 1 if we can reverse fuzzing for the data point and it contains contributions of individuals that are both unique and outliers. Because crowd-blending and commoner privacy both remove outlier contributions (and crowd-blending also removes all unique contributions), their privacy risk measure is always zero. Differential privacy, however, releases data with noise. If noise is not large enough, we will be able to reverse the fuzzing and in some cases identify individuals.

Figure 5 shows both the utility loss ($\bar{E}$) and privacy risk ($\bar{I}$) for *q1*–*q4* outputs protected by differential privacy, as we vary $\Delta f/\epsilon$. As expected, high values lead to high privacy risk and low utility loss, while low values eliminate privacy risk but increase utility loss.

Figure 6 shows the utility loss for differential privacy, crowd-blending, and commoner privacy (STDEV and MAD) assuming the value of $\Delta f/\epsilon$, which results in zero privacy risk. We vary the parameter k for crowd-blending and commoner privacy. Crowd-blending and commoner privacy both have utility loss under 1 for all queries, while differential privacy has it in the [100, 10,000] range. Further, for *q1*, commoner privacy with STDEV outlier detection loses only 47–82% of utility, while crowd-blending consistently loses 98–99%. Similarly, for *q2*, commoner privacy loses consistently less than 70–71% of utility, while crowd-blending consistently loses 98–99%. Similarly, for queries *q3* and *q4*, crowd-blending has around 99% utility cost, while commoner privacy significantly outperforms crowd-blending and achieves smaller than 51% utility loss even with $K = 50$ in *q3* and *q4*.

To summarize, with comparable privacy protections, crowd-blending and commoner privacy provide many orders of magnitude higher utility than differential privacy. In many cases, commoner privacy outperforms crowd-blending, halving its relative utility cost, and providing 9–49 times higher utility.

5.4 Scalability of Query Introspection

We implemented query introspection in our Patrol system and we now evaluate its scalability as the number of related queries increases. We run three select queries 100 times, without optimizations proposed in Section 4.1: (*qi1*) histogram of all TCP packets, grouped by source port; (*qi2*) same as *qi1* but excluding packets smaller than 100 bytes; (*qi3*) same as *qi1* but excluding packets outside a certain, small time range. The overhead of query introspection grew linearly from 1% of the total query processing time for the second round to 35–60% of the total runtime for the 100th round. Note that our implementation has no optimizations and runs as a single thread. We believe that further optimization and multi-threading could significantly reduce query introspection's overhead.

6 RELATED WORK

In this section we discuss work closely related to commoner privacy. De Montjoye et al. [6] discuss the re-identifiability of common user data, and highlight the contradicting requirements for utility and privacy.

Privacy protections for network trace data have been studied by many [4, 5, 18, 19, 25, 29, 30, 32, 35], mostly via sanitization and release of sanitized data. However, we explore a different approach where data remains with its provider.

Secure queries on network traces were sketched by us in [3], and explored by others using differential privacy [24] or manual vetting of queries [27]. Mittal et al. [26] further propose mediated trace analysis. Researchers submit their annotated black-box program and data providers repeatedly apply it to network traces with cleverly modified IP addresses, and compare outputs to detect if they depend on IP address data. Our focus is on preventing host re-identification through any quasi-identifier, not limited to IP addresses.

Sweeney et al.'s k-anonymity [33] is followed by work that utilizes those ideas. In more recent work, LeFevre et al. propose Incognito [20] to provide full-domain generalization using the k-anonymity model. Extensions are developed including (α, k)-anonymity [34], l-diversity [23] and t-closeness [22], and further applied to clustering [2], location privacy [13], etc. The main difference between k-anonymity and interactive k-anonymity is that the first is applied to original data, which is then released, while the second is applied to individual's contributions to query outputs. While k-anonymity is susceptible to a range of tracker attacks [8, 33], we prove in Section 4.1 that commoner privacy is not.

With regard to differential privacy and its challenges, Haeberlen et al. [16] found that a system for differential privacy called Airavat may be manipulated to mark a whole query "not differentially private" and abort, which can be used as proof of presence of an individual in a dataset. Further, researchers have struggled with how to best set the values of ϵ and the privacy budget to achieve strong privacy guarantees [16]. He et al. [17] propose Blowfish to automatically set those values to balance privacy and utility. Our work achieves better utility, while it provides sufficient privacy protection for network trace analysis.

7 CONCLUSIONS

Data privacy is a topic of increasing importance, as many of our daily interactions generate rich datasets, which may be analyzed and shared by data providers. Differential privacy, and lately crowd-blending, have been proposed as mechanisms that support complex queries over datasets, while guaranteeing strong privacy protections to individuals. Differential privacy performs well in many scenarios but has high utility cost on long-tailed datasets. Crowd-blending reduces this utility cost, but only when a sufficient number of individuals have the *same* records in the dataset; this situation is rare on datasets where some features have a large range of values.

We have proposed commoner privacy, and a mechanism to achieve it—interactive k-anonymity. We have further shown how commoner privacy can hold under query composition with careful recording and checking of queries. Commoner privacy improves the utility of query outputs on long-tailed and large-value-range datasets, compared to differential and crowd-blending privacy. It does so at the cost of lower privacy guarantees. Specifically, commoner privacy cannot defend against an all-but-one adversary. But it defends against an interactive adversary, who cannot learn anything specific about an individual, nor whether an individual is present in the dataset. While commoner privacy may not be the best match for some applications, we believe it is a competitive approach, with a realistic adversary model, that preserves much more research utility than state-of-the-art approaches, and at a modest privacy cost.

8 ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation, grant number 0914780. Authors are grateful to anonymous reviewers for their helpful comments.

REFERENCES

[1] MAWI Working Group Traffic Archive. http://tracer.csl.sony.co.jp/mawi/.

[2] Gagan Aggarwal, Tomás Feder, Krishnaram Kenthapadi, Samir Khuller, Rina Panigrahy, Dilys Thomas, and An Zhu. Achieving anonymity via clustering. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 153–162. ACM, 2006.

[3] Anonymized. Anonymized. In *Anonymized*, pages 3–10. ACM, 2008.

[4] S. Coull, M.P. Collins, C.V. Wright, F. Monrose, and M. Reiter. On Web Browsing Privacy in Anonymized NetFlows. In *Proceedings of the USENIX Security Symposium*, August 2007.

[5] S. Coull, C. Wright, F. Monrose, M. Collins, and M. Reiter. Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces. In *Proceedings of the Network and Distributed System Security Symposium*, February 2007.

[6] Yves-Alexandre de Montjoye, Laura Radaelli, Vivek Kumar Singh, and Alex "Sandy" Pentland. Unique in the shopping mall: On the re-identifiability of credit card metadata. *High Impact Journal*, 2014.

[7] Dorothy E Denning. A security model for the statistical database problem. In *Proceedings of the 2nd international workshop on Proceedings of the Second International Workshop on Statistical Database Management*, pages 368–390. Lawrence Berkeley Laboratory, 1983.

[8] Dorothy E Denning, Peter J Denning, and Mayer D Schwartz. The tracker: A threat to statistical database security. *ACM Transactions on Database Systems (TODS)*, 4(1):76–96, 1979.

[9] Dorothy E Denning and Jan Schlörer. A fast procedure for finding a tracker in a statistical database. *ACM Transactions on Database Systems (TODS)*, 5(1):88–102, 1980.

[10] Cynthia Dwork. Differential Privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming*, 2006.

[11] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[12] Srivatsava Ranjit Ganta, Shiva Prasad Kasiviswanathan, and Adam Smith. Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 265–273. ACM, 2008.

[13] Bugra Gedik and Ling Liu. A customizable k-anonymity model for protecting location privacy. 2004.

[14] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. Crowd-blending privacy. In *Advances in Cryptology–CRYPTO 2012*, pages 479–496. Springer, 2012.

[15] Irwin Guttman and Dennis E Smith. Investigation of rules for dealing with outliers in small samples from the normal distribution: I: Estimation of the mean. *Technometrics*, 11(3):527–550, 1969.

[16] Andreas Haeberlen, Benjamin C Pierce, and Arjun Narayan. Differential privacy under fire. In *USENIX Security Symposium*, 2011.

[17] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1447–1458. ACM, 2014.

[18] Eddie Kohler. Ipaggregate tool. http://www.cs.ucla.edu/~kohler/ipsumdump/aggcreateman.html.

[19] Eddie Kohler. Ipsumdump tool. http://www.cs.ucla.edu/~kohler/ipsumdump/.

[20] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60. ACM, 2005.

[21] Christophe Leys, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49(4):764–766, 2013.

[22] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, volume 7, pages 106–115, 2007.

[23] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[24] Frank McSherry and Ratul Mahajan. Differentially-private network trace analysis. *ACM SIGCOMM Computer Communication Review*, 41(4):123–134, 2011.

[25] Greg Minshall. tcpdpriv tool. http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html.

[26] Prateek Mittal, Vern Paxson, Robin Sommer, and Mark Winterrowd. Securing mediated trace access using black-box permutation analysis. In *HotNets*. Citeseer, 2009.

[27] J C Mogul and M Arlitt. Sc2d: An alternative to trace anonymization. In *Proceedings of the SIGCOMM 2006 Workshop on Mining Network Data*, 2006.

[28] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84. ACM, 2007.

[29] Ruoming Pang, Mark Allman, Vern Paxson, and Jason Lee. The devil and packet trace anonymization. *ACM SIGCOMM Computer Communications Review*, 36(1):29—38, 2006.

[30] Ruoming Pang and Vern Paxson. A High-level Programming Environment for Packet Trace Anonymization and Transformation. In *Proceedings of ACM SIGCOMM*, 2003.

[31] J Schlörer. Disclosure from statistical databases: quantitative aspects of trackers. *ACM Trans. Database Sys.*, 5(4):467–492, 1980.

[32] Q. Sun, D. R. Simon, Y. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu. Statistical Identification of Encrypted Web Browsing Traffic. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.

[33] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[34] Raymond Chi-Wing Wong, Jiuyong Li, Ada Wai-Chee Fu, and Ke Wang. (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 754–759. ACM, 2006.

[35] J. Xu, J. Fan, M. H. Ammar, and S. B. Moon. Prefix-Preserving IP Address Anonymization: Measurement-Based Security Evaluation and a New Cryptography-Based Scheme. In *Proceedings of the IEEE International Conference on Network Protocols*, 2002.