Mohammad Alaggan, Mathieu Cunche, and Sébastien Gambs

Privacy-preserving Wi-Fi Analytics

Abstract: As communications-enabled devices are becoming more ubiquitous, it becomes easier to track the movements of individuals through the radio signals broadcasted by their devices. Thus, while there is a strong interest for physical analytics platforms to leverage this information for many purposes, this tracking also threatens the privacy of individuals. To solve this issue, we propose a privacy-preserving solution for collecting *aggregate* mobility patterns while satisfying the strong guarantee of ε-differential privacy. More precisely, we introduce a sanitization mechanism for efficient, privacy-preserving and non-interactive approximate distinct counting for physical analytics based on perturbed Bloom filters called Pan-Private BLIP. We also extend and generalize previous approaches for estimating distinct count of events and joint events (*i.e.*, intersection and more generally t-out-of-n cardinalities). Finally, we evaluate expirementally our approach and compare it to previous ones on real datasets.

Keywords: Physical Analytics, Differential Privacy, Pan Privacy, Randomized Response, Cardinality Set Intersection.

DOI 10.1515/popets-2018-0010
Received 2017-08-31; revised 2017-12-15; accepted 2017-12-16.

1 Introduction

The possibility of detecting and tracking the movements of users through their Wi-Fi enabled devices [41] has lead to the emergence of the field of *physical analytics* [2, 49] that collect information on the human activity in the physical world. The main objective of this type of system is to be able to analyze human mobility at a large scale and to use the findings of this analysis for task such as urban planning or transportation optimization. For instance, mobility tracking through physical analytics has recently arrived in the London tube [37]. The system has been deployed over 54 stations and is collecting Wi-Fi probe requests emitted by mobile devices. It has already been used to gather the mobility data of tube users over a trial period of four weeks.

However, Wi-Fi tracking also raised important concerns, in particular with respect to the location privacy of individuals that are recorded [18] leading regulators and data protection authorities to take repressive measures. For example in the USA, the Federal Trade Commission has fined a Wi-Fi analytics company on the ground that users were not sufficiently informed and that no opt-out mechanism was available on site [28]. Similarly, the CNIL, the French data protection authority, has denied an authorization to a Wi-Fi analytics system on the ground that the data were not properly anonymized [13]. As a result, tracking companies have started to deploy some efforts to mitigate the impact on privacy, such as proposing an MLA (Mobile Location Analytics) Code of Conduct [29]. However, the current adopted solutions are often not enough to ensure a strong level of privacy as noted by the Electronic Frontier Foundation [31]. In particular, the obfuscation of MAC addresses (*e.g.*, through hashing or truncation), which has become a popular approach in the industry, does not offer strong privacy guarantees [20].

In this paper, our main contribution is to propose a novel privacy-preserving method for Wi-Fi analytics based on perturbed Bloom filters, called *Pan-Private BLIP*, that enables to estimate the number of devices, identified by their MAC address, that have been seen at a particular location (*i.e.*, access point) or that are in common between several locations. In fact, our method is even stronger as it can be used to estimate as well t-out-of-n distinct counts, which can be defined as the number of distinct MAC addresses that have been seen at exactly (*i.e.*, no more and no less) t locations out of n possible ones. Moreover, Pan-Private BLIP is generic in the sense that it actually works on sets of MAC addresses, agnostic to the interpretation of these sets. A set may represent a particular location (all MAC addresses observed at that location), or a particular time duration (all MAC addresses observed during that duration), or both (all MAC addresses observed at a particular location during a particular time duration) Thus, the sets may actually represent arbitrary spatio-temporal collec-

Mohammad Alaggan: Univ Lyon, Inria, INSA Lyon, CITI, Villeurbanne, France, E-mail: mohammad.nabil.h@gmail.com
Mathieu Cunche: Univ Lyon, INSA Lyon, Inria, CITI, Villeurbanne, France, E-mail: mathieu.cunche@insa-lyon.fr
Sébastien Gambs: Université du Québec à Montréal (UQAM), Canada, E-mail: gambs.sebastien@uqam.ca

tions of identifiers and the operations that can be performed on these sets enable for a richer class of analysis than simply counting the number of distinct devices at a particular location. Furthermore, our method is agnostic to the type of identifiers and could also be used to count and analyze other types of events such as web connections events.

Finally, another strength of Pan-Private BLIP is that it passively collects information in a privacy-preserving manner. More precisely, the entire internal state itself is as privacy-preserving as the final output of the system, a property known as *pan privacy* [25]. This means that the privacy guarantees of our scheme apply even against legal or illegal intrusions into the system. This property is directly in line with recommendation of data protection authorities such as the CNIL that require the anonymization to be perform on the fly (rather than storing everything and performing the sanitization at the end) in order to increase the level of protection against data stealing and leaking by against an external attacker but also an internal one.

The outline of the paper is as follows. First in Section 2, we review the elements of background necessary to the understanding of our work. Afterwards in Section 3, we introduce a novel differentially pan-private Bloom filter construction before explaining in Section 4 the techniques developed for event counting based on the data structure presented in the previous section. Then, we evaluate our methods in Section 5 followed by a discussion in Section 6. Finally, we review the related work in Section 7 before concluding in Section 8.

2 Background

In this section, we review the privacy notions that we are considering, namely differential privacy (Section 2.1) and pan privacy (Section 2.2) before introducing Bloom filters, the data structure we use (Section 2.3).

2.1 Differential Privacy

The concept of differential privacy was first introduced in 2006 by Dwork [22] in the context of statistical databases. Differential privacy is motivated by the standard objective of data analysis, namely that an analyst usually aims at learning *aggregate* information about the population rather than the data of specific individuals. In differential privacy, this translates to requiring that the statistical distribution of the output of an algorithm run on database (*e.g.*, a sanitization mechanism) should not depend too much on the data related to one individual. Notice that this property relates to the *distribution* of outputs, and not on the outputs themselves, and for this reason any mechanism satisfying differential privacy must be randomized. More precisely, the definition states that the probability p that the mechanism outputs a particular value $\boldsymbol{t}$ does not change much if an individual is added or removed from the input dataset. The amount of change allowed depends on a public privacy parameter ε as shown in Definition 2.1.

Definition 2.1 (ε-Differential Privacy [22])**.** A sanitization mechanism $\mathcal{M} : \{0,1\}^n \to \{0,1\}^n$ satisfies ε-differential privacy if for all inputs $\boldsymbol{a}, \boldsymbol{b} \in \{0,1\}^n$ and all outputs $\boldsymbol{t} \in \{0,1\}^n$:

$$\left| \ln \frac{\Pr[\mathcal{M}(\boldsymbol{a}) = \boldsymbol{t}]}{\Pr[\mathcal{M}(\boldsymbol{b}) = \boldsymbol{t}]} \right| \leqslant \varepsilon \cdot \|\boldsymbol{a} - \boldsymbol{b}\|_H \ ,$$

in which $\|\boldsymbol{a} - \boldsymbol{b}\|_H$ is the *Hamming distance* between $\boldsymbol{a}$ and $\boldsymbol{b}$, or equivalently, the number of positions at which they differ. The probability is taken over the coin tosses of $\mathcal{M}$.

If ε is equal to zero, the two distributions become identical and thus no information is revealed at all about the input. However in this case, no utility can be obtained from the output, which makes the sanitization mechanism private but also useless for analysis. In contrast, if ε is large, the mechanism is allowed to extract a significant amount of information about any individual and may use this information in producing its output. This is detrimental not only for privacy but also from the point of view of robust statistics. For instance, too much dependence on an outlier is likely to unreasonably skew the analysis results and the conclusions drawn.

One of the most appealing properties of differential privacy is that no assumptions need to be made about the adversary knowledge. In particular, differential privacy holds against a computationally unbounded adversary, even if they have arbitrary side knowledge (*i.e.*, auxiliary information). In addition, differential privacy also offers strong *composition* properties in the sense that applying an ε_1 and an ε_2-differentially private mechanisms on the *same*dataset is at least $(\varepsilon_1+\varepsilon_2)$-differentially private [39]. More precisely, applying two mechanisms on the same dataset is called *sequential* composition, while applying each mechanism on a disjoint dataset is called *parallel* composition. It has been proven that the parallel composition of an ε_1 and ε_2

mechanism is $\max(\varepsilon_1, \varepsilon_2)$-differentially private. This notion of compositionality stresses the fact that with every additional ε_i-differentially private release causes ε_i amount of privacy to be lost. The total amount of privacy lost across all released outputs of the *same* dataset is called the *privacy budget*. In practice, if the privacy budget is not bounded by a carefully chosen threshold, a complete loss of privacy may occur (we further discuss this issue in Section 6).

In general, ε is chosen to be small, typical values for it being 0.5, 1 or 3. In practice, the choice of the actual value may depend on a trade-off between the privacy level and the resulting utility (*e.g.*, measured in terms of global properties of the data or by an application-specific metric). Note there is currently no clear consensus in the community on a general method to measure privacy as a function of ε. Indeed, although ε itself serves as a quantification of privacy in the sense that the privacy gets stronger for lower values of ε, the practical guarantees ensured against various inference attacks may not be fully characterized only from the value of ε as the adversary's auxiliary information must be taken into account. For instance, Dwork shows that the amount of information leaked, as measured by information entropy, does not depend on the attack [22]. Nonetheless, for the purpose of analyzing this trade-off, some authors suggest choosing a value for ε that thwarts relevant *predetermined classes of inference attacks* [3]. In particular, they consider some attacks that do not require auxiliary knowledge, as a form of protecting against blatant non-privacy [19, 21].

2.2 Pan Privacy

Pan privacy was introduced by Dwork, Naor, Pitassi, Rothblum, and Yekhanin [25] in 2010. Their initial objective was to study and design algorithms that can maintain their privacy guarantees even if their entire internal state becomes visible to the adversary at some point. By "entire internal state becoming visible", we mean that the adversary get exactly one instantaneous snapshot of the internal state, as opposed to having continuous access to the evolving internal state. For example, this situation could occur following an intrusion by the adversary, whether it was legal (*e.g.*, following a subpoena order) or illegal. This model is particularly relevant for systems in which data is being continuously collected as is the case with a Wi-Fi tracking system such as ours[1]. In this model, the sequence of inputs is called a *stream*, which in our scenario would be the MAC addresses as they are observed in real-time by the system.

The original inventors of pan privacy have proposed to classify intrusions on the basis or whether or not they were announced. More precisely, an *announced* intrusion is one in which the system becomes aware of the intrusion immediately after it takes place and before any state change is performed. In the case of an announced intrusion, the system may take measures to ensure the continuity of the service in a privacy-preserving manner, such as the regeneration of the used hash function or the re-initialization of some of the randomness used in its data structures. In contrast, an *unannounced intrusion* is one in which the system does not realize that an intrusion took place. We later discuss that it has been proven impossible to achieve security against unannounced intrusions for systems similar to ours. Similarly to differential privacy algorithms, which ensure that the output is randomized, a pan-private algorithm ensures that both the output *and* the internal state are randomized in a differentially-private manner. Thus, pan-privacy can be seen as an extension of differential privacy. If more than one intrusion takes place, then all versions of the internal state the adversary has managed to observe, along with the output, must be *jointly* differentially private[2]. This notion is formalized in Definition 2.2.

Definition 2.2 (**Differential Pan Privacy** [25])**.** Let **Alg** be an algorithm, I the set of internal states of **Alg** and σ the set of possible outputs (an output is produced only at the end of the stream).

Then for all integers $d \geqslant 1$, the algorithm **Alg** mapping input prefixes to the range $I^d \times \sigma$ is *ε-differentially pan-private against d intrusions* if for all sets $I'_1, \ldots, I'_d \subseteq I$ and $\sigma' \subseteq \sigma$ and for all pairs of data stream prefixes S and S', we have that

$$\left| \ln \frac{\Pr\left[\mathbf{Alg}(S) \in (I'_1, \ldots, I'_d, \sigma')\right]}{\Pr\left[\mathbf{Alg}(S') \in (I'_1, \ldots, I'_d, \sigma')\right]} \right| \leqslant \varepsilon \cdot \|S - S'\|_X \ ,$$

in which the probability is taken over the coin flips of **Alg**, and $\|S - S'\|_X$ is the cardinality of the smallest set X such that $S \setminus X = S' \setminus X$. This means that S and S'

1 Note that the continuous collection of data is not the same as *continuous release* of output. The latter is called *continual observer* and its privacy-preserving implications are studied in [24]. We discuss a related issue in Section 6.

2 Here "jointly"refers to the sequential composition property of differential privacy [39].

differ only on the number of occurrences of elements of X and their positions.

Remark 2.1 (User-level versus event-level privacy). The original paper by Dwork, Naor, Pitassi, Rothblum, and Yekhanin [25] introduced the notion of *user-level* pan-privacy, in which *all* the events caused by a particular user are protected, and *event-level* pan-privacy, a weaker notion in which the protection only applies for individual events. Within the context of this paper, a user is a distinct MAC address while an event is a particular probe request issued by this MAC address. Consequently, a user may produce several events. In Wi-Fi tracking, the distinction between user-level pan-privacy and event-level pan-privacy is important since the user's device typically sends the user's MAC address in a probe request several times in a row, called a *burst*. The burst's number of probe requests and their fine-grained temporal pattern may enable device fingerprinting [44, 47] and is thus a threat to privacy. The definition we have adopted Definition 2.2, when X is a set of MAC addresses, amounts to *user-level* pan-privacy (assuming that a user does not have more than one MAC address), which unlike event-level pan-privacy provide a stronger guarantee that is suitable for Wi-Fi analytics.

2.3 Bloom Filters

Standard Bloom filters. A Bloom filter [11] is a data structure designed to answer set membership queries. The representation of a Bloom filter is simply a vector of bits, in which initially all the bits are set to zero. A specific Bloom filter is also associated with a set of k hash functions $h_1, h_2, \ldots, h_k$, the domain of which is the universe of all items and codomain is the set of positions in the bit vector associated with the Bloom filter. For example, if the bit vector is composed of m bits, then the hash functions map to the set $\{1, 2, \ldots, m\}$. When an item is *added* to the Bloom filter, a subset of those bits will be set to one. The choice of this particular subset depends on the item considered and on the hash functions. More precisely when an item i is added, the bits at the corresponding positions $\{h_1(i), h_2(i), \ldots, h_k(i)\}$ will be set to one. Similarly, these same positions will be queried when *querying for the presence* of this item in the Bloom filter. In this situation, an item will be considered to belong to the set if the bit values of those positions are all equal to one.

Due to their design, there may be hashing collisions leading to false positives (*i.e.*, falsely believing than an item is in the Bloom filter while it was not). The false-positive probability can be set arbitrary low at a trade-off of the size (*i.e.*, the number of bits) of the bit vector [11, 12]. In addition due to the potential collisions, it is not possible to remove items from a Bloom filter since it is not possible to know whether a bit was set to one (1) because of the item to be removed or (2) due to another unknown item. In a sense, a Bloom filter represents a set, and it has been known for a long time that it is possible to estimate the size of this set from its Bloom filter representation or to compute the size of the intersection or the union of two Bloom filters [14, 48].

Privacy-preserving Bloom filters. In 2012, Alaggan, Gambs, and Kermarrec [3] introduced a privacy-preserving version of Bloom filters, which they called BLIP (for *Bloom-and-fLIP*). In more details, they have shown that it is possible to randomly and independently flip each bit in a Bloom filter, with a probability strictly less than half, such that the resulting structure is ε-differentially private. They have also shown that even after a Bloom filter is flipped in such a way, it is still possible to extract some utility from it such as approximating the similarity between two profiles represented as sets. In a similar line of work Balu, Furon, and Gambs [8], and later Alaggan, Gambs, Matwin, and Tuhin [4] created techniques to estimate the set size and the size of intersection between two sets, given only their corresponding BLIPs. The application considered in [4] was the analysis of mobility patterns using mobile phone usage data (such as *Call Detail Records* collected by telecom operators). In this work, we expand this range of applications by considering arbitrary set counting operations on any number of BLIPs, allowing significantly more complex types of mobility pattern analysis. Our technique can also be applied directly to unperturbed Bloom filters as well and thus it might be of interest to other types of online big data analytics in which privacy is not a concern. In the next section, we discuss an extension to BLIP endowing it with the stronger privacy guarantee of differential pan-privacy.

3 Pan-Private Bloom Filters

In this section, we propose a pan-private version of Bloom filters satisfying differential privacy and that can withstand any number of *announced* intrusions at the cost of a graceful degradation in utility for each intrusion

(we address *unannounced* intrusions in Section 3.4). We call the proposed scheme Pan-Private BLIP. It is worth mentioning that the final released structure is entirely identical to ordinary BLIP for all intents and purposes. Thus, any algorithm that use BLIPs may be immediately applied with identical utility guarantees. The main difference lies in the internal workings of the algorithm that guarantees privacy protection to the internal state *while* the BLIP is being built. We believe that this feature is really a key property that analytics systems continously recording personal data should possess.

3.1 Pan-Private BLIP

Pan-Private BLIP algorithm takes as input a stream of items from some universe (*e.g.*, the universe of all MAC addresses). The set of internal states I and the set of outputs σ are both in $\{0,1\}^m$. The algorithm is composed of three main subroutines: (1) Algorithm 1: *initialization*, (2) Algorithm 2: *item addition* and (3) Algorithm 3: *intrusion recovery*. Furthermore before the final release, the number of intrusions d is incremented since the output is basically the internal state itself and revealing it shall be counted as an intrusion (however Algorithm 3 will not be invoked). The three phases are collectively referred to as Pan-Private BLIP.

In the initialization phase, the bits of the Bloom filter are set identically and independently at random according to the distribution Bernoulli(μ_0), in which $\mu_0 < 1/2$. In line with [5], on which later sections depend, we will define $\mu_0 = 1/2 - \eta/2$ and $\mu_1 = 1/2 + \eta/2$ for η a parameter in the range $(0,1)$. For all intents and purposes, η is completely interchangeable with the differential privacy parameter, ε, via the relation $\eta = \frac{\exp(\varepsilon)-1}{\exp(\varepsilon)+1}$ and will be treated as such hereafter. This relation between η and ε is a side effect of the relation between μ_0 and ε and is derived in the proof of Lemma 3.1.

When an item is added to the Bloom filter, the bits that are supposed to be set to one according to the hash functions applied to that item, will be set identically and independently at random according to a different distribution: Bernoulli(μ_1) for $\mu_1 > 1/2$.

Finally, in case of intrusion, pan privacy is guaranteed by re-initializing all the bits in the Bloom filter identically and independently at random according to Bernoulli(μ_0) or Bernoulli(μ_1) depending on the current value of the bits.

Since the algorithm does not store the true unperturbed values of these bits in its internal memory, there is no way to know what the bits actually were, and

Algorithm 1 Initializing an empty Bloom filter

1: **procedure** INITIALIZE(m, k, ε)
2: Set $d \leftarrow 1$ ▷ Number of intrusions so far
3: Set $\eta_0 \leftarrow \frac{\exp(\varepsilon)-1}{\exp(\varepsilon)+1}$
4: Set $\eta \leftarrow \eta_0$
5: Set $\mu_0 \leftarrow 1/2 - \eta/2$
6: Set $\mu_1 \leftarrow 1/2 + \eta/2$
7: **for** each `i in` $\{1,\ldots,m\}$ **do** ▷ For each bit
8: Initialize $\mathcal{B}[i] \leftarrow$ flip a coin with probability of 1 being μ_0
9: **end for**
10: **end procedure**

Algorithm 2 Adding an element to the Bloom filter

1: **procedure** ADD(x)
2: **for** each `i in` $\{1,\ldots,k\}$ **do**
3: Set $\mathcal{B}[h_i(x)] \leftarrow$ flip a coin with probability of 1 being μ_1
4: **end for**
5: **end procedure**

Algorithm 3 Recovering after an intrusion takes place

1: **procedure** AFTERINTRUSION
2: **for** each `i in` $\{1,\ldots,m\}$ **do**
3: **if** $\mathcal{B}[i] = 0$ **then**
4: Set $\mathcal{B}[i] \leftarrow$ flip a coin with probability of 1 being μ_0
5: **else**
6: Set $\mathcal{B}[i] \leftarrow$ flip a coin with probability of 1 being μ_1
7: **end if**
8: Set $d \leftarrow d + 1$ ▷ Increase the number of intrusions
9: Set $\eta \leftarrow \eta_0 \cdot \eta$
10: Set $\mu_0 \leftarrow 1/2 - \eta/2$
11: Set $\mu_1 \leftarrow 1/2 + \eta/2$
12: **end for**
13: **end procedure**

the re-initialization will depend solely on the noisy bits. Thus, such re-initialization is similar to flipping an already flipped bit, effectively compounding the flipping probability. In particular, the result is *as if* the Bloom filter had been initialized with a higher privacy parameter than it was initially given. Algorithm 3 takes this new augmented privacy parameter into account. At the end, the subsequent state and eventually the released output will have decreased utility with each intrusion, which is unavoidable according to [23].

An alternative strategy that could preserve utility, depending on the needs of application and available memory resources, is to freeze the old Bloom filter in its current state, leave it aside in a list of compromised Bloom filters, and start a new Bloom filter from scratch, using the initial and unaugmented flipping probability. All d Bloom filters will then be released at the end of the stream, in which techniques described later in the paper may be used to compute their union cardinality or other functions as necessary. Finally, note that the flipping probability (and ε) are part of the internal state but they are considered public information and will be eventually released as part of the output. As a consequence, there is no privacy violation occurring when the adversary observes them as part of an intrusion and there is no need to bestow them with pan-privacy guarantees.

3.2 Privacy Analysis

Lemma 3.1 (Differential privacy of Pan-Private BLIP)**.** *When no intrusion occurs, Pan-Private BLIP is ε-differentially private.*

The k factors in $(1+\exp(\varepsilon/k))^{-1}$ guarantees privacy for the items encoded in the Bloom filter, since each item may impact up to k different bits through the use of k different and *independent* hash functions [3]. However, for the rest of the paper we will ignore the k factors and assume it is equal to 1, for the sake of presentation. It was observed in [4] that $k = 1$ produces more utility that $k > 1$, which agrees with the intuition that even for equivalent privacy guarantees, using more hash functions increases the chances of hash collisions, and subsequently, loss of information. Our theorems do not lose generality by dropping k as they hold for $k > 1$ by using $\varphi' = \varphi^k$ in place of φ.

Theorem 3.2 (Pan privacy of Pan-Private BLIP)**.** *For every positive integer d, Pan-Private BLIP is at least $(d\varepsilon)$-differentially pan-private for $d-1$ announced intrusions, in which $\mu_0 = (1+\exp(\varepsilon))^{-1}$.*

Note that the bound given by Theorem 3.2 holds for both strategies: (1) the strategy in which d BLIPs are released (in which case the bound is tight) and (2) the strategy in which only one BLIP is released. However, in the latter case the bound is not tight and the effective privacy guarantee is stronger than the former strategy. In particular, using the moments accountant method [1], it is possible to numerically show for $\epsilon = 1$ and $d = 30$, that our algorithm is $(1.81, 10^{-5})$-differentially pan-private (*i.e.*, 1.81-differentially pan-private with probability at least $1 - 10^{-5}$) instead of being 30-differentially pan-private.

3.3 Utility Analysis

The quantification of utility depends what is done with the output of the algorithm. This paper (in particular Section 4), along with the unifying framework of [5], encompass the prior works of [3, 4, 8] and many others.

The previous bounds in [3, 4, 8] apply automatically for their respective applications once we substitute the correct value of the flipping probability $\mu_0 = 1/2 - \eta^d/2$, in which d is the number of intrusions including the output itself. In particular, for [5] all upper and lower bounds after d intrusions follow by replacing η by η^d. Such simple substitution is one of the reasons that the parameter η is more suited for our analyses than ε. Although it is straightforward, we refrain here from rewriting all these bounds with such a substitution due to space considerations.

Note that when considering the interaction of several BLIPs, such substitution is only applicable when all the BLIPs have undergone the same number of intrusions. This requirement can be ensured even without interactions between the BLIP holders (in case various BLIPs were held and managed by different parties), simply by releasing the number of intrusions a BLIP has endured along with its normal output. Moreover, this process should always be performed since otherwise if the final flipping probability, which depends on the number of intrusions, is not be publicly known this means that the output of BLIP will be useless. Afterwards the combination of several BLIPs together can be performed by emulating an intrusion on the BLIPs with the lowest number of intrusions until their flipping probability matches the BLIPs with the highest number of intrusions. The same technique also allows the combination

of two BLIPs with different privacy parameters, regardless of whether intrusions were involved.

3.4 Limitations

Unannounced Intrusions. Our algorithm can only tolerate *announced* intrusions. However, this is an inherent limitation as it has been proven to be impossible to tolerate any number of *unannounced* intrusions in the case in which the internal state is released at the end. Indeed, such final release is counted as an announced intrusion and Dwork, Naor, Pitassi, Rothblum, and Yekhanin [25] have shown that for all $\varepsilon > 0$, no ε-differentially pan-private algorithm with a state having a finite size for approximating stream density (*e.g.*, counting the number of distinct elements) can tolerate one unannounced intrusion followed by an announced intrusion [25, Corollary 6.2]. Since our algorithm always releases its internal state at the end, then any unannounced intrusion would have to have happened *before* such final release. Hence, the result of [25, Corollary 6.2] applies, precluding the possibility that BLIP or any other efficient methods can tolerate even as little as *one* unannounced intrusion.

In-core protection. Note that in the analysis of the algorithm, the space in which the item itself is temporarily stored before being included in the Bloom filter is not counted as part of the internal state to be protected. To be more precise, this means implicitly that the period of time in which an item exists in memory in clear is not protected by our scheme, which is unavoidable unless we make additional assumptions about the fact this information is protected by a highly trusted component such as a dedicated tamper-proof hardware. However, this is only a limitation if we consider a live intrusion, in which the adversary accesses the machine directly while it is actively operating on data. Alternatively, in case of a court-ordered subpoena, the most plausible scenario is that the system will be shut down and only cold storage will be accessed by the intruder. It is beyond the scope of this paper to consider further side-channel attacks.

4 Distinct Counting

An intuitive and well-known observation that the number of bit positions set to one in a Bloom filter is closely related to the size of the set that the Bloom filter encodes [48]. Furthermore, the cardinality of the intersection of two sets is also related to joint features of their corresponding Bloom filters [14, 46]. For instance, these relationships have been extensively used for event counting and stream analysis to estimate the number of distinct events taking place. As a concrete example, such as approach has been followed for estimating the number of unique IP addresses observed for the purpose of detecting denial-of-service attacks [14].

These relationships have been extended to privacy-preserving Bloom filters (such as BLIP [3]) in the case of one Bloom filter [8] and two Bloom filters [4]. More precisely, given a privacy-preserving Bloom filter, each of the previous works [3, 4, 8] adopt a *direct* approach by analyzing the direct relationship between the observed perturbed bits and the target set cardinality to be computed. These approaches are tailored specifically for one or two Bloom filters and are challenging to generalize even to as little as three Bloom filters.

In contrast to that direct approach, we propose a *two-step* method, in which we decompose the problem into two independent sub-problems that can be solved individually. The *un-flipping step* is to estimate, from the perturbed Bloom filters, the number of bit positions (either in a single Bloom filter or jointly in two or more), whose values were zero prior to perturbation. In particular, this quantity can be defined as the number of bit positions which, jointly across n Bloom filters, have *exactly* t ones, in any combination. This estimate is provided for all $t \in \{0, 1, \ldots, n\}$. The *un-hashing step* uses those estimates to compute the required set cardinalities.

The un-flipping step was already solved by Alaggan, Cunche, and Minier [5] and we use it here implicitly as a black box. In a nutshell, their method takes the perturbed t-out-of-n density vector (*cf.* Definition 4.2) and uses a linear program to find a candidate vector close to the original density vector with high probability. Their method is designed for values of ε approaching 0, and thus the utility of the estimated density vector may not be as optimal as possible for large values of ε. We refer the interested reader to [5] for further details on the technique and its error bounds.

For the un-hashing step, usually a model of hashing collisions must be assumed in order to be able to give meaningful cardinality estimates and error bounds. In particular, previous work exists in which the hashing model analyzed is that of 4-wise independent hash functions [25]. However, in this paper we only study the truly random hash function model for simplicity of presentation [17]. In this model, hash functions are assumed to map each input independently and identically at random to its hash value. While this model is unrealistic since representing such a hash function requires expo-

nential amount of memory, Chung, Mitzenmacher, and Vadhan have shown that provided that the input stream itself (*e.g.*, the stream of MAC addresses) has enough entropy, the resulting utility in practice when the truly random hash function is replaced by an $O(1)$-wise independent hash function will not differ significantly from the theoretical analysis that used truly random hash functions [17]. Note also that it is not pan-private to store an associative list of MAC addresses and their randomly chosen hash values, even if it is practical to do so for the small set of observed MAC addresses in most applications, since the keys of such associative list, even if hashed or truncated, may reveal information about the observed MAC addresses, beating the purpose of using a pan-private algorithm.

Note that the un-flipping step is independent of the semantics of the Bloom filter and the amount of hash collisions taking place and that the un-hashing step is agnostic to the sanitization mechanism used. Moreover, each of those two steps can be independently useful if integrated with a suitable substitute of the other and thus different sanitization mechanisms or different hashing models may be accommodated.

4.1 t-out-of-n Distinct Count

In the previous section, we reviewed the well-known relationship between the set size (quantity 1) and the corresponding number of bits set to one in the associated Bloom filter (quantity 2). In this section, we generalize these two quantities to multiple sets and describe a novel relationship between those two generalizations. This relationship will serve as the gist of the un-hashing step described previously and provides the theoretical framework for our distinct count estimation algorithm.

For n sets, the generalization of the set size (quantity 1) is the *t-out-of-n distinct count*, which we denote as the vector $\boldsymbol{T}$, while that of the number of bits set to 1 in the Bloom filter (quantity 2) is that of *t-out-of-n density*, denoted as the vector $\boldsymbol{D}$. The t-out-of-n density is precisely the output of un-flipping step described earlier. Hereafter, we formalize these definitions.

Definition 4.1 (t-out-of-n distinct count)**.** Given a multiset s of n sets: $s = \{\!\{s_1, s_2, \ldots, s_n\}\!\}$, let $\boldsymbol{T} = (T_1, \ldots, T_n)^\top$ denote the vector of t-out-of-n distinct counts, in which T_t is the number of elements belonging to exactly t sets of n.

Definition 4.2 (t-out-of-n density [5])**.** Given n Bloom filters: $b^1, b^2, \ldots, b^n$, each of size m, let $\boldsymbol{D} = (D_1, \ldots, D_n)^\top$ denote the vector of t-out-of-n densities, in which D_t is the density of bit positions which, jointly across the n Bloom filters, contain exactly t bits set to one out of n. In particular $D_t = \frac{1}{m} \cdot \left|\left\{i | i \in \{1, 2, \ldots, m\} \wedge t = \sum_{j=1}^{n} b_i^j\right\}\right|$.

As described in the previous section, $\boldsymbol{D}$ can be estimated up to additive error using the technique of [5] (the un-flipping step) when the Bloom filters are pan-private. One of the advantages of the technique of [5] is that the estimate of $(D_0, D_1, \ldots, D_n)^\top$, which contains one more component than $\boldsymbol{D}$, namely D_0 is guaranteed to sum to one as expected (since it accounts for the fact that the t-out-of-n density is a partition of all bit positions), unlike other methods that uses unbiased estimators (such as the inverse of the transition matrix). This consistency enhances the accuracy for larger n compared to the baseline introduced later (Section 5.1), which is an inclusion-exclusion based approach. By inclusion-exclusion based approach we mean to refer to the relationship, known under that name, between cardinality set intersection and cardinality set union of two sets, and its generalization for more than two sets. For example, the inclusion-exclusion principle for two sets states that $|A \cup B| = |A| + |B| - |A \cap B|$.

4.2 t-out-of-n Distinct Count Estimator: The un-hashing step

The goal of the un-hashing step is to convert densities ($\boldsymbol{D}$) to distinct counts ($\boldsymbol{T}$). In the hashing model studied in this section, we treat the distinct counts as an unknown given constant, while densities are a random variable over the probability space of the random choice of a hash function from a family of hash functions. The family of hash functions considered is the family of truly random functions (*i.e.*, the chosen hash function is picked uniformly at random from the set of all functions from a given domain to a given range). This model is characterized by the probability vector $\boldsymbol{P}$, indexed by t from 1 to n, such that P_i is the probability, over the choice of the hash function, that a particular bit position contains exactly t bits set to 1 across the n Bloom filters. It can be shown that $\boldsymbol{P} = \mathbb{E}[\boldsymbol{D}]$ (Proposition C.1).

4.2.1 The un-hashing step estimator

Hereafter, we described our proposed un-hashing step estimator and prove upper bounds on its additive error, along with a numeric example.

Definition 4.3 (t-out-of-n Distinct Count Estimator)**.** Given a multiset S of n sets: $S = \{\!\{s_1, s_2, \ldots, s_n\}\!\}$, and their corresponding BLIPs, each of size m, let $\boldsymbol{D}$ be the t-out-of-n density vector (*cf.* Definition 4.2) and $\widehat{\boldsymbol{D}}$ be its estimator (*i.e.* is the output of the un-flipping step when it is fed the BLIPs of each set in S). Our proposed estimator of the t-out-of-n distinct count vector $\boldsymbol{T}$ (*cf.* Definition 4.1) is:

$$\widehat{\boldsymbol{T}} \triangleq \widehat{\boldsymbol{D}} / \ln(1/\varphi) \ , \tag{1}$$

in which $\varphi = 1 - 1/m$.

When m is large, the estimator may be simplified to $\widehat{\boldsymbol{D}}/\ln(1/\varphi) \approx m\widehat{\boldsymbol{D}} + O(1/m)$. To compute the additive error of this estimator, which effectively is composed of three disjoint sources of errors, we need to first define a few symbols. The first source of error, the vector $\boldsymbol{e}_F$, is the additive error of the un-flipping step estimator[3]. Having the ability to isolate $\boldsymbol{e}_F$ from other sources of error makes it easier to analyze the impact on accuracy of using alternative un-flipping step estimators. However, we only use the additive error of the [5] estimator. In all cases, $\boldsymbol{e}_F \triangleq \widehat{\boldsymbol{D}} - \boldsymbol{D}$. The second source of error $\boldsymbol{e}_H$ is unrelated to the noise injected by the differential privacy mechanism, but is due to the variance resulting from the random choice of the hash function and the ensuing effect on the number of collisions[4]. In particular $\boldsymbol{e}_H = \boldsymbol{D} - \mathbb{E}[\boldsymbol{D}]$, in which the expectation is taken over the choice of hash function. While, $\boldsymbol{e}_H$ depends on the chosen family of hash functions, it is usually very small compared to other sources of error , such as often three orders of magnitude smaller. For instannce, in our experiments it is usually 10 or 100 when other sources of error are respectively 10000 or 100000.. As a consequence, this error may be omitted in most cases. Both sources of error described so far jointly contribute to the *variance* of our estimator. This means that the probabilistic amount of additive error that will differ for different runs, with different random coins, even for the exact same input multiset S and parameters m and ε.

The remaining source of error, $\boldsymbol{e}_T$ is the bias, in the sense that it is not affected by the random coins. It will depend solely on the input multiset S, m, ε and on the form of Equation (1). It is also perhaps the most intriguing component of the error since it gives rise to a rich set of alternative behaviors depending on the form of Equation (1), often related to number-theoretic notions and computationally-difficult problems. The general form of the total additive error as a function of these components is given in Theorem 4.4.

Theorem 4.4 (Estimator's Additive Error)**.** *Let Q^{-1} is the matrix whose i, j entry is $(-1)^{n+i+j-1}\binom{j}{n-i}$ for $1 \leqslant i, j \leqslant n$ and φ be equal to $1 - 1/m$, for m being the size of the Bloom filters. Then the additive error of $\widehat{\boldsymbol{T}}$ (cf. Equation* (1) *in Definition 4.3) is:*

$$\boldsymbol{T} - \widehat{\boldsymbol{T}} = Q^{-1}\boldsymbol{e}_T + (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi \ . \tag{2}$$

Note that this is an *exact* additive error as any sources of uncertainty is deferred to the underlying error components (the proof is provided in Appendix C). A worked numerical example is presented in Appendix E.

4.3 Upper Bound

We upper bound all terms except the hashing variance ($\boldsymbol{e}_H$) since it is negligible compared to other factors. The proof of Lemma 4.5 is in Appendix D. The upper bound in Figure 2 shows that although it can be loose when m is small, it is still useful for the choice of m, given that ξ has been estimated suitably, as described in [5].

Lemma 4.5 (Upper Bound)**.** *If the un-flipping step estimator used was that of [5], then the upper bound Γ_ξ on the error $\|\boldsymbol{T} - \widehat{\boldsymbol{T}}\|$, ignoring $\boldsymbol{e}_H$, is, for sufficiently large m:*

$$\Gamma_\xi = \frac{\|Q^{-1}\boldsymbol{K}(x \mapsto x^2)\|}{2m} + O(\eta^{-n})\frac{2\xi\sqrt{-\ln(\beta)\ln(n+1)}}{-\ln(\varphi)\sqrt{2m}},$$

with probability at least $1-\beta$, in which all the norms are the max norm; $\|\boldsymbol{x}\|_\infty \triangleq \max_i |x_i|$, or its induced norm for matrices, and $\xi \in (0,1)$ is the precomputed multiplier described in [5].

5 Experimental Evaluation

To evaluate the efficiency of our approach we tested it on a dataset provided by CISCO, coming from a Meraki lo-

3 The mnemonic "e" comes from "error" and "F" comes from "un-Flipping".

4 The mnemonic "H" is for "Hashing".

cation analytics plateform[5]. This dataset have been generated by a dozen access points (AP) installed around a busy roundabout in a large European city. The resulting data, collected over a period of three months in 2016, contains timestamped events corresponding to the detection of Wi-Fi devices by the APs composing the platform. For obvious privacy reasons, we did not have direct access to the raw MAC addresses. Rather, the MAC addresses provided were first pseudonymized using a secret one way function to which the authors do not have access (*e.g.*, using an HMAC with a secret key that is thrown away afterwards). The generated pseudonyms are consistent throughout the dataset.

Afterwards, the dataset was aggregated on a daily basis in the sense that all the MAC addresses observed by all APs during a particular day are encoded together in a single privacy-preserving Bloom filter (BLIP). From this BLIP, we use the tools developed in Section 4 to estimate the number of distinct MAC addresses observed that day. As mentioned previously, the roundabout at which the data was collected is very busy and thus the number of distinct devices seen in day can be as high as 50 000 unique MAC addresses.

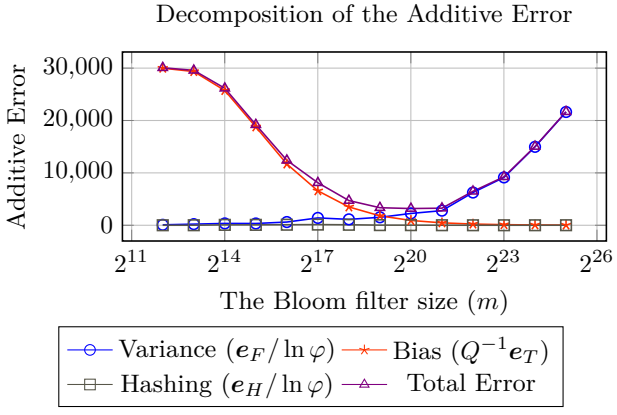

Fig. 1. Error decomposition for two randomly generated sets, A and B, such that $|A| = |B| = 25000$ and $|A \cap B| = 10000$. The additive error, when $\varepsilon = 1$ shown on the y-axis is $\|T - \widehat{T}\|_\infty$, the maximum of additive errors for all t.

5.1 Baseline

To the best of our knowledge, there is no method in the litterature that combines three or more flipped Bloom filters, or even *raw unsanitized* Bloom filters, in way that

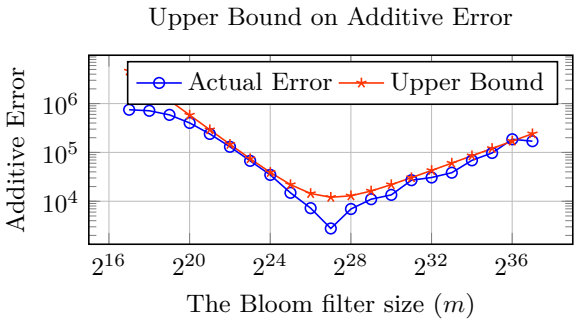

Fig. 2. Upper bound $\Gamma_{1/8}$ on the additive error for two randomly generated sets, A and B, such that $|A| = |B| = 625000$ and $|A \cap B| = 250000$. The additive error, when $\varepsilon = 1$, shown on the y-axis is $\|T - \widehat{T}\|_\infty$, the maximum of additive errors for all t.

is comparable to our work[6]. Thus, to evaluate the efficiency of the un-hashing step, which transforms the density estimate of the un-flipping step into t-out-of-n distinct counts, we propose as a baseline a straightforward generalization of the approach of [46] for computing the cardinality intersection of two Bloom filters, to compute instead the cardinality intersection of n Bloom filters (equivalently, n-out-of-n distinct counts). Both our method and the proposed baseline take as input the output of the un-flipping step [5]. The baseline relies on the 0-out-of-n densities and the technique of [48], which convert the density of zeros into an estimate of the cardinality set union, to compute the union of all 2^n subsets of the n Bloom filters. In particular, it will invoke the un-flipping step $\Theta(2^n)$ times. Given the values of all these unions, the baseline will employ the inclusion-exclusion principle $\Theta(2^n)$ times to *recursively* compute the intersection of the given n Bloom filters. As shown later in the results obtained, our method provides better utility and runtime performance for higher n, while being at least as good as the baseline for small n. Furthermore, the baseline is not scalable to large n as it requires an exponential runtime in n.

5.2 Utility versus Dataset Size

The strong guarantees provided by differential privacy come from the fact that it bounds the amount of information each user contributes to the output [6]. In fact, in [6], Alvim, Andrés, Chatzikokolakis, and Palamidessi showed that the utility increased with the number of

5 https://meraki.cisco.com/solutions/location-analytics

6 RAPPOR [26] is not comparable to our work as will be discussed later

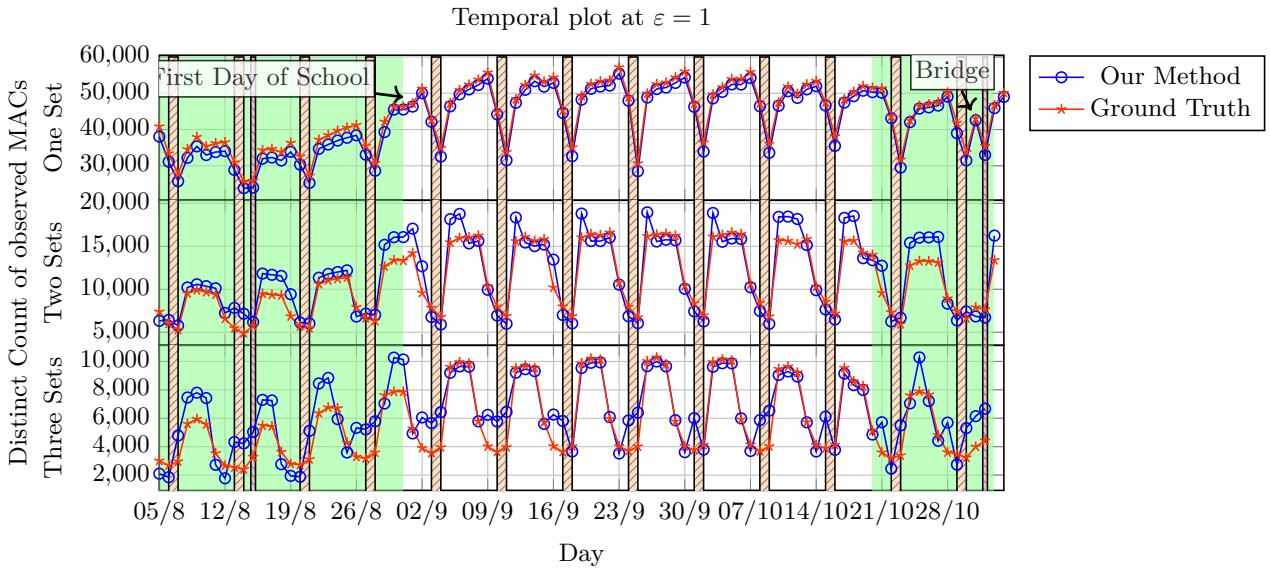

Fig. 3. Plotting the privacy-preserving estimates against the ground truth. The figure shows the distinct count of MAC addresses for each day, for the intersection of a rolling window of 2 consecutive days, and for the intersection of a rolling window of 3 consecutive days. For each situation, m, the size of the Bloom filters, was chosen as the one with the least error among possible values of m. The first day is August 5 2016, a Friday, and the last is November 3 2016, totaling 91 days. Weekends (Saturdays and Sundays) are highlighted in orange (north east pattern), official holidays in pink (north west pattern), and school holidays in green (no pattern). The count of October 31 is less than the average of a working day, indicating that many people took the day off as a bridge holiday between the weekend and the official holiday on November 1.

users in the dataset. Intuitively, the more users there are, the more information can be pieced together about their aggregated properties. In contrast, the fewer users, there are the less utility can be extracted. In particular, in the extreme situation in which there is only one user in the system, no utility can be extracted as any such utility will be considered as a privacy breach since it is ultimately information about that particular user. Therefore, differential privacy mechanisms provide higher utility when the number of users is large while behaving poorly when that number is small. This relationship depends on the privacy parameter ε and the mechanism itself. We explore the trade-off between the size of the dataset and the utility by evaluating our method on subsamples of the original dataset of various sizes. More precisely, in Figure 9 we compare the results obtained for the original dataset versus a subsample of one-half (respectively one-quarter and one-eighth) of its size, for different levels of the privacy parameter ε.

5.3 Results

Temporal patterns. In Figure 3, we plot in the first row the estimated count of distinct MACs (y-axis) seen each day (x-axis) along with the ground truth. The second and third rows are similar except that they show the estimated count of distinct MACs appearing in a window of respectively two and three consecutive days. The figure shows that most temporal patterns are preserved, even with a privacy level as strict as $\varepsilon = 1$. For each row, m, the size of the Bloom filters, was chosen as the one with the least multiplicative error among possible values of m. Figure 4 complements this choice by showing the median of the multiplicative error for varying values of m and ε. The observed existence of a value of m in the middle of the spectrum that is optimal in terms of utility agrees with Appendix E and Figure 1.

Spatial patterns. To complement the previous experiment, in Figure 5 (left panel) we evaluate an application-oriented utility metric. More precisely, we consider that the task at hand is the computation of an origin-destination matrix [9]. In a nutshell, an origin-destination matrix is a matrix showing the number of people who were at a particular origin location at one point in time and then moved to the destination location at another point in time. When the time component is taken into account they it is known as time-dependent or dynamic origin-destination matrix, while otherwise it is called time-independent or static. Time-independent origin-destination matrices are generally used when data

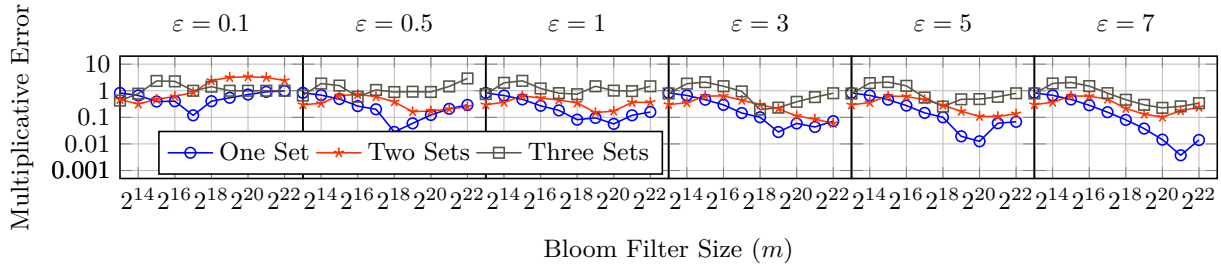

Fig. 4. Median of the multiplicative error for one, two and three sets.

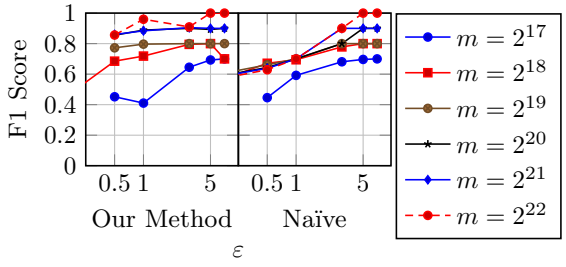

Fig. 5. F1-score obtained for the top-10 most common origin-destination pairs of access points (*i.e.*, higher values are better). The lines correspond to different sizes m of the Bloom filter with results averaging over 80 runs.

about a large period of time is available, such as in our case, and long-term planning is desired. We constructed the time-independent origin-destination matrix between the 14 different APs in the dataset for the entire period of three months. The value in a cell of the matrix is simply the number of distinct MAC address in the intersection of the two access points. To realize this, we consider the application of finding the top-k most common origin-destination pairs. The quality of our estimation is measured by the F1-score, whose value ranges between 0 (no match) and 1 (perfect match), and is equivalent to the scaled harmonic mean of recall (r) and precision (p): $F1 = 2/(1/r + 1/p) = 2pr/(p + r)$ when $p + r > 0$ and ∞ otherwise. More precisely, the F1-score measures the extent to which two sets are similar. In particular, the *precision* is the fraction, out of the inferred set, of the true top-k set of origin-destination pairs, while the *recall* is the fraction, out of the true top-k set of origin-destination pairs, which were inferred. The results show that our scheme reached as high as an F1-score of 0.8 for some values of m, even for the strict privacy level of $\varepsilon = 0.5$.

Baseline. Figure 8 shows how our method is comparable, in terms of utility, against the less computationally efficient and less flexible alternative. The figure demonstrates that our method always performs better when ε is strict, but also that the utility is often comparable with, if not better than the baseline when ε is not strict.

Utility versus dataset size. We also assess the sensitivity of our method with respect to the dataset size in Figure 9. To realize this, we sequentially remove half of the distinct MAC addresses in the dataset three times in a row before evaluating our method each time. On average, a day has roughly 44000 (respectively 22000, 11000 and 55000) distinct MAC addresses in the dataset with 0% (respectively 50%, 75% and 87.5%) of distinct MAC addresses randomly removed. An intersection of two (respectively three) consecutive days has roughly 11000 (respectively 6000) distinct MAC addresses on average. The figure shows that the utility decreases as the size of the dataset get smaller, as expected from a differentially-private algorithm which limits the amount of information released by an individual, thus resulting as the total amount of released information diminishing as the number of individuals decreases. The figure also shows that the intersection of three days (respectively two days) is more sensitive to dataset size reduction than the intersection of two days (respectively one day). This observation suggests that for higher-level mobility analytics requiring the involvement of many spatio-temporal BLIPs, either a large dataset should be available or a large privacy budget should be allowed.

Naïve approach. In Figure 5 (both panels) we study an assumption made by the un-flipping step. In particular, the un-flipping step is designed for strict values of ε (*i.e.*, values of ε approaching zero). This fact has two consequences: (1) when ε is large the assumption is violated and the utility may not be as optimal as possible, and (2) the un-flipping step makes our method particularly well-suited for strict value of ε. The plot confirms both consequences. In particular, we compare our method to the naïve approach of considering the perturbed Bloom filter directly. Using the perturbed Bloom filter directly makes sense for high value of ε since in that case it will

approach the true Bloom filter. While such high value of ε is not recommended for real applications, if it was utilized in practice, it may be easier and sometimes more accurate to rely on the naïve approach rather than our method.

5.4 World Cup 1998 Website Access Logs

To demonstrate its applicability to other contexts, we have also applied our method to a non-WiFi dataset. The dataset considered corresponds to the access logs of the website of FIFA World Cup championship that have taken place in France 1998 [7]. This dataset records all the HTTP requests sent to and served by the various servers hosting the website, from 30 April 1998 to 26 July 1998. Instead of using MAC addresses as identifiers for individual persons, we use the source IP address of an HTTP request. There is 1,352,804,107 HTTP requests in the dataset, sent by 2,769,901 unique IPs and served by 33 servers. In Figure 6, we show the intersection of n *consecutive* days, starting from 30 April 1998, for n ranging up to 30 days. The figure contains one subplot for each n, ordered left-to-right, and spanning multiple rows, in order of increasing n. The parameters for each n, namely the choice of the Bloom filter size m and the privacy parameter ϵ was computed as the parameters minimizing the error, among $m \in \{2^8, 2^9 \ldots, 2^{24}\}$ and $\varepsilon \in \{0.1, 0.5, 1, 3\}$. The values chosen for m, from $n = 1$ up to $n = 30$, in that order, are: $\{2^{22}, 2^{23}, 2^{22}, 2^{21}, 2^{22}, 2^{20}, 2^{19}, 2^{22}, 2^{22}, 2^{21}, 2^{21}, 2^{21}, 2^{20}, 2^{21}, 2^{21}, 2^{20}, 2^{20}, 2^{20}, 2^{20}, 2^{21}, 2^{18}, 2^{20}, 2^{20}, 2^{20}, 2^{19}, 2^{19}, 2^{19}, 2^{19}, 2^{18}\}$, and the values for ε are {3, 3, 3, 3, 3, 3, 3, 1, 1, 3, 3, 3, 3, 1, 1, 0.5, 0.5, 0.5, 0.5, 1, 1, 0.5, 0.5, 0.5, 3, 3, 1, 1, 1, 3}. We can see from the figure that the temporal patterns are preserved even as n grows so large, albeit as clearly some almost-constant bias is introduced. The apparent irregularities for $n \in \{6, 7, 21\}$ are most likely due to the use of power-of-two values for the size of the Bloom filter, which may not correspond to the optimal value. In particular, these irregularities are not unique to $n \in \{6, 7, 21\}$. Indeed, they also appear – to a lesser degree – in $n \in \{2, 3, 4, 25, 26\}$, and to an even lesser degree in others.

6 Discussion

Consider the scenario in which three BLIPs are released, and in which each BLIP contains information about a different day and each was flipped using the privacy level ε. If the three days are guaranteed not to share any users at all, then the composition of the three BLIPs is also ε-differentially private, by the parallel composition lemma [39]. However, if all users appear in all three days then the total privacy budget used by the system is in fact 3ε, due the sequential composition lemma [39]. Consequently, the users who appear in more than one day are given weaker guarantees about their privacy than users who appear in at most one day. In the extreme case in which they appear in many days it could even be the case that they can be identified with high probability [26], showing that the privacy risk increases with the number of days in which a user appears in the data.

However, in practice most people appear in a small number of days, according to the power-law principle [51]. For example, Figure 7 shows that 90% people do not appear in more than 6 days in the dataset considered in this paper. Nonetheless, it remains unsustainable for the privacy of the minority of users who happen to appear in many days to keep releasing BLIPs publicly for an unbounded number of days.

Countermeasures to specific attacks such as re-identifying the most frequent users by means of techniques like RAPPOR [26] can be deployed. For instance, one possibility is to change the hash functions used by the Bloom filters on a regular basis (*e.g.*, every week) while keeping those hash functions secret, which is a *compartmentalization* technique. While RAPPOR can handle perturbed Bloom filters that use different hash functions (called cohorts), their method requires the knowledge of the hash function used.

Unfortunately, a generic countermeasure that would work against any attack is much more challenging to accomplish, especially without making assumptions about the auxiliary knowledge the adversary has. For example, even if no BLIPs were released, but rather locally computed statistical information involving the same users were released many times, the privacy budget would still be exhausted. Indeed, the query response itself carries information and is also composable, which means that solutions such as compartmentalization, which assumes that the culprit of the composable privacy loss is the ability to link different BLIPs, would not work.

As mentioned previously, the above problem is known as *continual observation* problem and has been studied before [23]). In our case in which the internal state itself (the BLIP) is released, it might be called *continual intrusion* as well. Dwork, Naor, Pitassi, and Rothblum show that a pan-private algorithm releasing a counter (*i.e.*, the count estimation from a BLIP falls

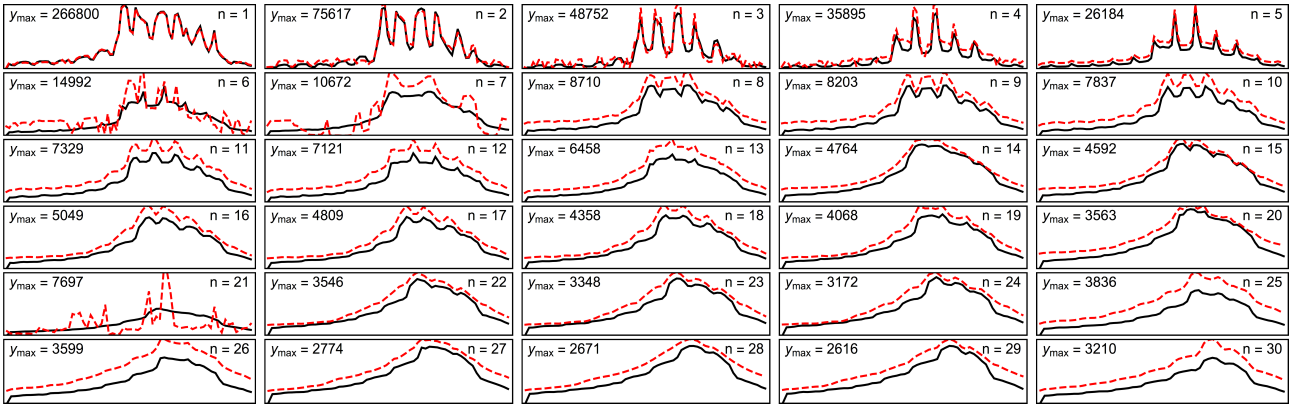

Fig. 6. The number of the daily distinct IP addresses sending an HTTP request to the FIFA Worldcup 1998 championship website. The x-axes is the day, and the y axis is the distinct count of the IP addresses ($y_{\min}$ is always 0). The dark solid black line represents the ground truth, while the dashed red line represents the estimate by our method.

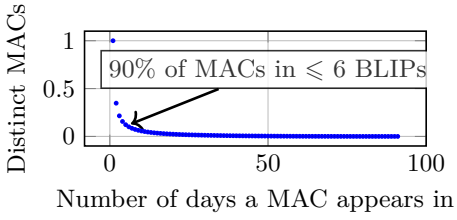

Fig. 7. The privacy budget consumed by each user. The y-axis is cumulative and shows the ratio of distinct MAC addresses appearing in at least x BLIPs.

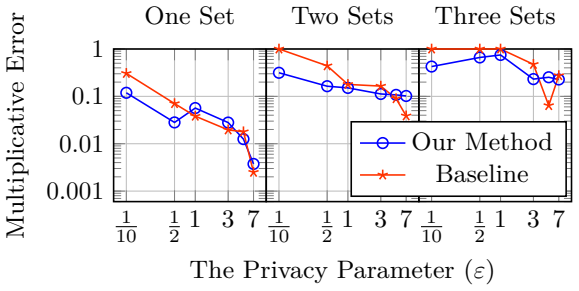

Fig. 8. Median of the multiplicative error.

under this category) under continual intrusion must be performing some form of randomized response [23]. Additionally, they prove that to maintain privacy in this situation, the error must grow as the number of releases increases [23, Theorem 4.3]. As a consequence when the number of days increases, the error will eventually become too large, effectively enforcing an upper bound on the total number of usable BLIPs. More precisely, an unbounded number of days may be released, but all except the first few will be totally random and convey no information. This can be implemented by triggering the re-initialization of the privacy parameters after a release as is the case after an intrusion (*i.e.*, Algorithm 3).

7 Related Work

The privacy guarantees provided by plain Bloom filters has been formally studied by Bianchi, Bracciale, and Loreti [10]. Their study highlights their inherent limitation, when the universe can be enumerated, which is the case with network identifiers like MAC addresses. Focusing on the false-positive rate and the plausible deniability features provided by Bloom filters, they demonstrate that depending on the filter parameters, not all elements may benefit from the same level of protection and even that some of them can be denied this protection. For example, if a particular bit in the Bloom filter can only be set to 1 by only one item in the universe. One of their suggestions for improvement was to randomly choose the values of some bits in the array, which is similar in spirit to the approach that we have taken.

In [30], Gonçalves, José, and Baquero considered the problem of estimating the number of persons in a privacy-preserving manner using Bloom filters. The privacy guarantees of this system is based solely on the intrinsic false-positive probability of Bloom filters, and thus it provides a weak protection while our approach provides stronger guarantees by combining differential and pan-privacy. More recently, Lim, Zimmerling, and Thiele have introduced DEVCNT [35], a system for counting Wi-Fi devices that only relies on the detection and counting of active Wi-Fi scan events to estimate the number of devices, thus disregarding privacy issues associated with the collection of identifiers. In contrast to

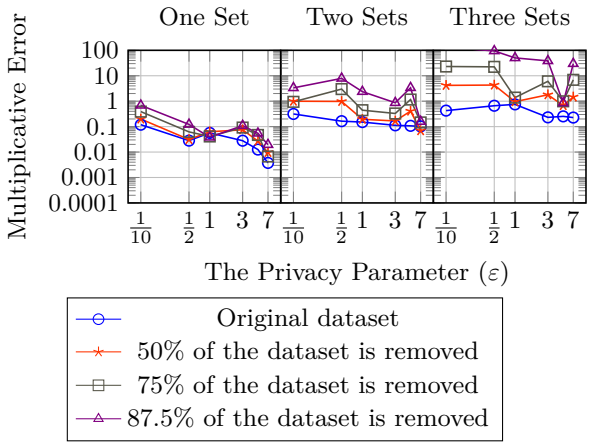

Fig. 9. Experiment to see the effect of sampling on the dataset size.

our approach, DEVCNT is limited to counting devices at a single location and does not support complex set operations as our protocol.

Fawaz, Kim, and Shin investigated [27] the tradeoff between privacy and rewards in location-based services in which users are willingly sharing their whereabouts in exchange of rewards. The proposed system ensure that the user will get a fair trade-off between the level of information provided to the system and the reward received. In contrast to our approach, the user has to actively interact with the tracking service, meaning that he can control the level of information shared and that he has given explicit and informed consent, which is not the case in our work. Furthermore, the tracking entity considered by Fawaz et al. has for objective to profile each user individually while in our setting the tracker is interested in global properties of the dataset.

In [26], Erlingsson, Pihur, and Korolova introduced RAPPOR for *Randomized Aggregatable Privacy-Preserving Ordinal Response*, a system that allows to extract, in a privacy-preserving manner, the most-frequently visited websites for a very large group of users. While RAPPOR also relies on Bloom Filter to encode data and on the addition of noise to make them differentially private, their objective is different from ours and incomparable. More precisely, they compute the exact "heavy-hitters" set from the intersection of *millions* of sanitized Bloom filters, while we compute only the *number* of *distinct* elements in the intersection (and other combinations, like union and symmetric difference) of a handful of sanitized Bloom filters (usually less than 10).

In a similar way, Melis, Danezis, and Cristofaro presented an approach based on sketches to crowdsource statistics in a privacy-preserving way [40]. This scheme works by combining counting sketches with homomorphic encryption for private aggregation to which Laplacian noise can be added to ensure differential privacy. This last work has then been applied by Pyrgelis, De Cristofaro, and Ross to the case of crowdsourced mobility analytics [43]. Using real world datasets, they validate the efficiency of their approach in the context of predictive analytic tasks and anomaly detection.

Haze, a privacy-preserving traffic monitoring system Brown et al. based on threshold cryptography and differential privacy techniques has been introduced by [15]. Popa, Blumberg, Balakrishnan, and Li proposed PrivStats [42], a solution for the aggregation of statistics on location data featuring provable guarantees on location privacy and privacy-preserving accountability. A system for the distributed collection of visit quantities has been proposed [32] by Kopp, Mock, and May. In [45], Shi, Chan, Rieffel, Chow, and Song proposed a framework allowing an untrusted aggregator to learn statistics on time-series data produced by users.

All those works assume that the user is cooperating by locally performing computation. However, this approach is only valid in a crowdsourcing scenario in which users willingly cooperate which is not the case of our setting. Indeed, physical analytics based on Wi-Fi typically rely on the passive collection of signal emitted by personal devices [41] and is particularly pervasive as it does not require any interactions with those devices.

In [36], Liyue Fan and Li Xiong introduced FAST, an adaptive system for releasing time-series while providing differential privacy. In FAST, a central entity is in charge of periodically releasing statistics over private data. FAST minimize the overall privacy budget by adaptively sampling the data to be tailored to the dynamics of the data. Although FAST allows the release of differentialy private time-series without the cooperation of the subjects, it does not protect against an intruder that can access the internal state of the system in contrast to our approach. Cao, Carminati, Ferrari, and Tan introduced CASTLE [16] a framework for aggregating data stream under time constraints. The time-constrained publication of datastream has also been investigated by Zhou, Han, Pei, Jiang, Tao, and Jia [50] and Li, Ooi, and Wang [34]. As our approach, those works address the issue of on the fly sanitization but they only provide k-anonymity while our scheme provides differential and pan privacy.

In the seminal paper on which they introduced the concept of pan-privacy [24], Dwork, Naor, Pitassi, and Rothblum also presented several algorithms to compute statistics over a stream of events while enforcing user-

level pan-privacy. Namely they presented algorithms to estimate the count of distinct events, but also other statistics such as t-Cropped Mean, k-Heavy Hitters, and t-incidence [25]. For the t-incidence case which is closest to our application, they use a data structure composed of a collection of arrays, in which each array acts as a noisy histogram modulo a different prime. Our work provides a practical implementation of pan-private algorithms relying on another type of data structures, namely Bloom filters. Furthermore, our approach enable the computation of complex set operations, after the data collection phase, while the approach in Dwork et al. restricts the analysis to a unique stream of events.

MAC randomization is a client side solution that has been proposed and put forward by vendors to protect users from tracking. In this technique, instead of using the real MAC address, the device uses an address that is periodically renewed. The main drawback of MAC randomization is that its applicability is limited to scanning mode. Indeed, when connected to a network, the device falls back to a stable MAC address [38], thus exposing the user to tracking.

8 Conclusion

In this paper, we have proposed Pan-Private BLIP a novel privacy-preserving sanitization mechanism for physical analytics that can be used to estimate the number of Wi-Fi enabled devices that have been seen at a particular access point. Our method can also be used to compte more complex correlations such as t-out-of-n distinct counts. These operations pave the way for a richer class of spatio-temporal analysis tasks than simply counting the number of distinct devices that have been seen at a particular location. In addition, our method is agnostic to the type of identifiers and therefore that it could be used for performing analytics in other contexts and for other types of data as demonstrated in our experiments. In addition, Pan-Private BLIP offers a strong level of privacy as it is ensures pan privacy, which is an extension of differential privacy providing strong guarantees even against intrusions into the system by ensuring that the internal state of the algorithm is as private as the output itself. Finally, Pan-Private BLIP is efficient both in terms of computation time and memory used due to the use of Bloom filters but also due to the fact that the data structure can be built in an online and privacy-preserving manner.

As future work, we are aiming at designing and testing practical inference attacks targeted at Wi-Fi data to be able to evaluate the privacy provided by Pan-Private BLIP for different values of ε. Once a meaningful set of inference attacks has been developed, it can also be used by a practitioner to tailor the value of ε when deploying our system in real-life. Another avenue of research concerns the investigation of using our approach to perform more complex physical analysis tasks, such for road traffic application including traffic forecast and anomaly detection [43], point-to-point travel time [35] or urban network characterization [33].

Finally, we want to underlined that those results can find may applications beyond Wi-Fi tracking. Basically, any domain in which users are associated to a unique identifier could benefit from our contribution such as for instance public transportation systems in which users authentify via smartcards or traffic monitoring applications based on RFID tags or plate-number recognition.

9 Acknowledgments

This work is supported by the Cisco grant CG# 593780 and the French *Programme d'Investissement d'Avenir -FSN-AAP 1 Protection des Données Personnelles projet ADAGE n° P128356-2659748PIA ADAGE* as well as an NSERC Discovery Grant and Discovery Accelerator Supplement Grant for Sébastien Gambs. The authors would also like to thank the PETS reviewers for their helpful comments and feedback.

References

[1] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, Vienna, Austria, October 2016. ACM.

[2] U. G. Acer, G. Vanderhulst, A. Masshadi, A. Boran, C. Forlivesi, P. M. Scholl, and F. Kawsar. Capturing Personal and Crowd Behavior with Wi-Fi Analytics. In *Proceedings of the 3rd International on Workshop on Physical Analytics*, WPA '16, pages 43–48, New York, NY, USA, 2016. ACM.

[3] M. Alaggan, S. Gambs, and A.-M. Kermarrec. BLIP: Non-Interactive Differentially-Private Similarity Computation on Bloom Filters. In *Proceedings of the 14th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS'12)*, Toronto, Canada, October, 2012.

[4] M. Alaggan, S. Gambs, S. Matwin, and M. Tuhin. Sanitization of Call Detail Records via Differentially-Private Bloom Filters. In P. Samarati, editor, *Data and Applications Security and Privacy XXIX - 29th Annual IFIP WG 11.3 Working Conference, DBSec 2015, Fairfax, VA, USA, July 13-15, 2015, Proceedings*, volume 9149 of *Lecture Notes in Computer Science*, pages 223–230. Springer, 2015.

[5] M. Alaggan, M. Cunche, and M. Minier. Non-interactive (t, n)-Incidence Counting from Differentially Private Indicator Vectors. In *Proceedings of the 2017 ACM on International Workshop on Security And Privacy Analytics, IWSPA@CODASPY 2017*, Scottsdale, AZ, USA, March 2017. ACM.

[6] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, and C. Palamidessi. On the relation between differential privacy and quantitative information flow. In L. Aceto, M. Henzinger, and J. Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2011.

[7] M. Arlitt and T. Jin. 1998 World Cup Web Site Access Logs, August 1998. URL http://www.acm.org/sigcomm/ITA/.

[8] R. Balu, T. Furon, and S. Gambs. Challenging Differential Privacy: The Case of Non-Interactive Mechanisms. In *ESORICS*, pages 146–164, 2014.

[9] S. Bera and K. Rao. Estimation of origin-destination matrix from traffic counts: the state of the art. *European Transport/Trasporti Europei*, 49:3–23, 2011.

[10] G. Bianchi, L. Bracciale, and P. Loreti. " Better Than Nothing" Privacy with Bloom Filters: To What Extent? In *International Conference on Privacy in Statistical Databases*, pages 348–363. Springer, 2012.

[11] B. H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Commun. ACM*, 13(7):422–426, July 1970. ISSN 0001-0782.

[12] P. Bose, H. Guo, E. Kranakis, A. Maheshwari, P. Morin, J. Morrison, M. H. M. Smid, and Y. Tang. On the false-positive rate of bloom filters. *Inf. Process. Lett.*, 108(4): 210–213, 2008.

[13] C. Bouchenard. JC Decaux's pedestrian tracking system blocked by french data regulator. *Marketinglaw*, 2015. URL http://marketinglaw.osborneclarke.com/advertising-regulation/jc-decauxs-pedestrian-tracking-system-blocked-by-french-data-regulator/.

[14] A. Z. Broder and M. Mitzenmacher. Survey: Network Applications of Bloom Filters: A Survey. *Internet Mathematics*, 1(4): 485–509, 2003.

[15] J. W. S. Brown, O. Ohrimenko, and R. Tamassia. Haze: Privacy-preserving real-time traffic statistics. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL'13, pages 540–543, New York, NY, USA, 2013. ACM.

[16] J. Cao, B. Carminati, E. Ferrari, and K. L. Tan. CASTLE: A delay-constrained scheme for ks-anonymizing data streams. In *2008 IEEE 24th International Conference on Data Engineering*, pages 1376–1378, Apr. 2008.

[17] K. Chung, M. Mitzenmacher, and S. P. Vadhan. Why simple hash functions work: Exploiting the entropy in a data stream. *Theory of Computing*, 9:897–945, 2013.

[18] S. Clifford and Q. Hardy. Attention, Shoppers: Store Is Tracking Your Cell. *The New York Times*, 2013. URL http://www.nytimes.com/2013/07/15/business/attention-shopper-stores-are-tracking-your-cell.html?pagewanted=all.

[19] A. De. Lower bounds in differential privacy. In R. Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 321–338. Springer, 2012.

[20] L. Demir, M. Cunche, and C. Lauradoux. Analysing the privacy policies of Wi-Fi trackers. pages 39–44. ACM Press, 2014.

[21] I. Dinur and K. Nissim. Revealing information while preserving privacy. In F. Neven, C. Beeri, and T. Milo, editors, *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 9-12, 2003, San Diego, CA, USA*, pages 202–210. ACM, 2003.

[22] C. Dwork. Differential Privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP'06), Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12, Venice, Italy, 2006. Springer.

[23] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In L. J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 715–724. ACM, 2010.

[24] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724. ACM, 2010.

[25] C. Dwork, M. Naor, T. Pitassi, G. N. Rothblum, and S. Yekhanin. Pan-Private Streaming Algorithms. In A. C. Yao, editor, *Proceedings of the 1st Symposium on Innovations in Computer Science (ICS'10)*, pages 66–80, Tsinghua University, Beijing, China, 2010. Tsinghua University Press.

[26] U. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. pages 1054–1067. ACM Press, 2014.

[27] K. Fawaz, K.-H. Kim, and K. G. Shin. Privacy vs. Reward in Indoor Location-Based Services. *Proceedings on Privacy Enhancing Technologies*, 2016(4):102–122, 2016. ISSN 2299-0984. 00000.

[28] Federal Trade Commisioin. Retail tracking firm settles ftc charges it misled consumers about opt out choices, 2015. URL https://www.ftc.gov/news-events/press-releases/2015/04/retail-tracking-firm-settles-ftc-charges-it-misled-consumers.

[29] Future of Privacy Forum. Mobile location analytics code of conduct, 2013. URL https://fpf.org/wp-content/uploads/10.22.13-FINAL-MLA-Code.pdf.

[30] N. Gonçalves, R. José, and C. Baquero. Privacy Preserving Gate Counting with Collaborative Bluetooth Scanners. In R. Meersman, T. Dillon, and P. Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2011 Workshops*, number 7046 in Lecture Notes in Computer Science, pages 534–543. Springer Berlin Heidelberg, Oct. 2011.

[31] P. Higgins and L. Tien. Mobile tracking code of conduct falls short of protecting consumers. *Electronic Frontier Foundation*, 2013. URL https://www.eff.org/fr/deeplinks/2013/10/mobile-tracking-code-conduct-falls-short-protecting-consumers.

[32] C. Kopp, M. Mock, and M. May. Privacy-preserving distributed monitoring of visit quantities. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, pages 438–441, New York, NY, USA, 2012. ACM.

[33] P. A. Laharotte, R. Billot, E. Come, L. Oukhellou, A. Nantes, and N. E. E. Faouzi. Spatiotemporal Analysis of Bluetooth Data: Application to a Large Urban Network. *IEEE Transactions on Intelligent Transportation Systems*, 16 (3):1439–1448, June 2015. ISSN 1524-9050.

[34] J. Li, B. C. Ooi, and W. Wang. Anonymizing streaming data for privacy protection. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 1367–1369. IEEE, 2008.

[35] R. Lim, M. Zimmerling, and L. Thiele. Passive, Privacy-Preserving Real-Time Counting of Unmodified Smartphones via ZigBee Interference. In *2015 International Conference on Distributed Computing in Sensor Systems*, pages 115–126, June 2015.

[36] Liyue Fan and Li Xiong. Adaptively Sharing Time-Series with Differential Privacy. Technical report, Jan. 2013.

[37] J. O. Malley. Here's what tfl learned from tracking your phone on the tube. *Gizmodo UK*, 2017. URL http://www.gizmodo.co.uk/2017/02/heres-what-tfl-learned-from-tracking-your-phone-on-the-tube/.

[38] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown. A Study of MAC Address Randomization in Mobile Devices and When it Fails. *Proceedings on Privacy Enhancing Technologies*, 2017(4): 268–286, 2017.

[39] F. McSherry. Privacy Integrated Queries: an Extensible Platform for Privacy-Preserving Data Analysis. *Commun. ACM*, 53(9):89–97, 2010.

[40] L. Melis, G. Danezis, and E. D. Cristofaro. Efficient private statistics with succinct sketches. *CoRR*, abs/1508.06110, 2015.

[41] A. Musa and J. Eriksson. Tracking unmodified smartphones using wi-fi monitors. In *Proceedings of the 10th ACM conference on embedded network sensor systems*, pages 281–294. ACM, 2012.

[42] R. A. Popa, A. J. Blumberg, H. Balakrishnan, and F. H. Li. Privacy and accountability for location-based aggregate statistics. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 653–666, New York, NY, USA, 2011. ACM.

[43] A. Pyrgelis, E. De Cristofaro, and G. J. Ross. Privacy-friendly mobility analytics using aggregate location data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 34. ACM, 2016.

[44] A. E. C. Redondi, D. Sanvito, and M. Cesana. Passive Classification of Wi-Fi Enabled Devices. pages 51–58. ACM Press, 2016.

[45] E. Shi, H. T. H. Chan, E. Rieffel, R. Chow, and D. Song. Privacy-preserving aggregation of time-series data. In *Annual Network & Distributed System Security Symposium (NDSS)*. Internet Society., 2011.

[46] S. J. Swamidass and P. Baldi. Mathematical correction for fingerprint similarity measures to improve chemical retrieval. *Journal of Chemical Information and Modeling*, 47(3):952–964, 2007.

[47] O. Waltari and J. Kangasharju. The Wireless Shark: Identifying WiFi Devices Based on Probe Fingerprints. In *Proceedings of the First Workshop on Mobile Data*, MobiData '16, pages 1–6, New York, NY, USA, 2016. ACM. 00000.

[48] K. Whang, B. T. V. Zanden, and H. M. Taylor. A linear-time probabilistic counting algorithm for database applications. *ACM Trans. Database Syst.*, 15(2):208–229, 1990.

[49] Y. Zeng, P. H. Pathak, and P. Mohapatra. Analyzing Shopper's Behavior Through WiFi Signals. In *Proceedings of the 2Nd Workshop on Workshop on Physical Analytics*, WPA '15, pages 13–18, New York, NY, USA, 2015. ACM.

[50] B. Zhou, Y. Han, J. Pei, B. Jiang, Y. Tao, and Y. Jia. Continuous Privacy Preserving Publishing of Data Streams. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, pages 648–659, New York, NY, USA, 2009. ACM.

[51] G. Zipf. *Human behavior and the principle of least effort: an introduction to human ecology*. Addison-Wesley Press, 1949.

A Proofs for Pan-Private BLIP

Lemma 3.1 (Differential privacy of Pan-Private BLIP)**.** *When no intrusion occurs, Pan-Private BLIP is ε-differentially private.*

Proof of Lemma 3.1. Consider a raw unflipped, Bloom filter. The probabilistic map taking the bit 0 to Bernoulli(μ_0) and the bit 1 to Bernoulli(μ_1), in which $\mu_1 = 1 - \mu_0$ is essentially a bit flipping operation in which a bit in the raw Bloom filter is flipped with probability μ_0. Hence if μ_0 equals the flipping probability $(1 + \exp(\varepsilon/k))^{-1}$ used in [3], we establish an equivalence between our Pan-Private BLIP and the original BLIP in [3]. In particular, the distributions of the output of Pan-Private BLIP and the BLIP produced in [3] are identical. Hence, by reduction, since the latter is ε-differentially private [3, Theorem 1] so is the former. □

Theorem 3.2 (Pan privacy of Pan-Private BLIP)**.** *For every positive integer d, Pan-Private BLIP is at least $(d\varepsilon)$-differentially pan-private for $d-1$ announced intrusions, in which $\mu_0 = (1 + \exp(\varepsilon))^{-1}$.*

Proof of Theorem 3.2. The final output is counted in d even though it is not technically an intrusion. In particular, $d = 1$ means that no intrusions occurred and the

adversary has obtained only one version of the internal state, which corresponds to the output willingly released at the termination of the algorithm. Therefore, if there was $d-1$ intrusions, the intruder will have at most d different versions of the flipped Bloom filter. According to Lemma 3.1, for $i \in \{1, 2, \ldots, d\}$, the ith version will be ε_i-differentially private for $\varepsilon_i = \ln\left(\frac{1+\eta^i}{1-\eta^i}\right)$. In particular, we note that $\varepsilon_i \geqslant \varepsilon_{i+1}$. Therefore, by the sequential composition property of differential privacy [39], the composition of all such versions is $(\sum_i \varepsilon_i)$-differentially private. Finally, since ε is equivalent to ε_1 – the case in which no intrusions take place – we end up with $\sum_i \varepsilon_i \leqslant \sum_i \varepsilon_1 = d\varepsilon$. □

B Proof of Bias' Relation to T and P

Lemma B.1 (Bias relation to $\boldsymbol{T}$ and $\boldsymbol{P}$)**.**

$$\boldsymbol{e}_T = Q\boldsymbol{T} + Q\boldsymbol{P}/\ln\varphi \ .$$

B.1 Nomenclature

If m is the size of the Bloom filters used, then let $\varphi \triangleq 1 - 1/m$.

B.1.1 Sets

We consider n sets, $s_1, s_2, \ldots, s_n$, the members of the multiset $S \triangleq \{\!\{s_1, s_2, \ldots, s_n\}\!\}$. For a subset $s \subseteq S$ of S, we define the set $(\bigcup s)$ to be the union of the sets in s, that is $\bigcup s \triangleq \bigcup_{s' \in s} s'$. The we denote the multiset $\mathcal{P}'(S) = 2^S \setminus \varnothing$ as the powerset of S without the empty set. Therefore, $\mathcal{P}'(S)$ contains $2^n - 1$ sets. Then, the *Venn decomposition* $\mathcal{V}$ of S is a set with $2^n - 1$ elements. In this decomposition, each element x of $\mathcal{V}$ is bijectively associated with an element y of $\mathcal{P}'(S)$ such that x is the set containing the elements in each of the sets in y but not in any set of $S \setminus y$. That is $x = (\bigcup y) \setminus (\bigcup (S \setminus y))$.

B.1.2 Vectors

Let the function $K_t(f) = \sum_{s \subset S, |s|=t} f(|\bigcup s|)$, and the vector-valued function $\boldsymbol{K}(f) = (K_1(f), K_2(f), \ldots, K_n(f))$. Then define $\boldsymbol{\Phi} \triangleq K(x \mapsto x)$, and $\boldsymbol{\Psi} \triangleq \boldsymbol{K}(x \mapsto \varphi^x)$. For example, if $S = \{\!\{A, B, C\}\!\}$, then $\boldsymbol{\Phi} = (|A| + |B| + |C|, |A \cup B| + |A \cup C| + |C \cup A|, |A \cup B \cup C|)^\top$ and $\boldsymbol{\Psi} = (\varphi^{|A|} + \varphi^{|B|} + \varphi^{|C|}, \varphi^{|A \cup B|} + \varphi^{|A \cup C|} + \varphi^{|C \cup A|}, \varphi^{|A \cup B \cup C|})^\top$. Furthermore let $\boldsymbol{e}_T \triangleq \boldsymbol{\Phi} + (\boldsymbol{v} - \boldsymbol{\Psi})/\ln\varphi$. Note that by Lemma D.1, $\boldsymbol{e}_T$ asymptotically approaches $\boldsymbol{K}(x \mapsto x^2)$ for large m.

Let $\boldsymbol{v}$ be such that $v_i = \binom{n}{i}$ for $i \in \{1, 2, \ldots, n\}$. Let the vector $\boldsymbol{P}$ is the probability vector indexed by t from 1 to n, and similarly let the vector $\boldsymbol{P'}$ the probability vector indexed by t from 0 to $n-1$. One can be derived from the other by the fact that the sum $P_0 + P_1 + \cdots + P_n = 1$ or via the relationship defined in Lemma B.2.

Let $\boldsymbol{V}$ be the vector describing the number of elements in each element of the Venn decomposition $\mathcal{V}$ of S. That is, $v = |x|$, for v a component of $\boldsymbol{V}$, and x is the element of $\mathcal{V}$ corresponding to v. Finally, let $\boldsymbol{C}$ be the vector describing the sizes of all the possible $2^n - 1$ combinations of set unions among the n sets in S; that is, and element c of $\boldsymbol{C}$ corresponding to an element y of $\mathcal{P}'(S)$ is simply $|\bigcup y|$.

Finally, let $\boldsymbol{e}_i$ be the standard basis vectors, which is $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$ and $\boldsymbol{e}_2 = (0, 1, 0, \ldots, 0)^\top$, and so on, until $\boldsymbol{e}_n = (0, 0, \ldots, 0, 1)^\top$.

B.1.3 Matrices

Let U be the upper triangular $n \times n$ Pascal matrix, which means the matrix whose i, j entry is the binomial coefficient $\binom{j}{i}$, in which i, j range from 1 to n. Define Q as the matrix whose i, j entry equals $\sum_{1 \leqslant k \leqslant j} \binom{n-k}{i-1} = \binom{n}{i} - \binom{n-j}{i}$ for $i, j \in \{1, 2, \ldots, n\}$. Then the i, j entry of its inverse, Q^{-1}, is $(-1)^{n+i+j-1}\binom{j}{n-i}$. The matrix Q is derived later in Lemma B.7.

There exist two linear operators: Z, the linear operator mapping $\boldsymbol{V}$ to $\boldsymbol{T}$ (the vector defined in Definition 4.1); and R, mapping $\boldsymbol{V}$ to $\boldsymbol{C}$. Interestingly, Z also maps $\boldsymbol{C}$ to $\boldsymbol{\Phi}$. If we define $\chi(i)$ for a positive integer $i < 2^n$ to be the n-component vector representing its binary expansion and $|\chi(i)|$ to be the number of non-zero components of that vector (*i.e.*, $\|\chi(i)\|_0$), then we can define the linear operators Z and R as follows: Element i, j of Z is 1 if $|\chi(j)| = i$ and 0 otherwise, for $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, 2^n - 1\}$. Element i, j of R is 1 if $\chi(i) \cdot \chi(j) > 0$ and 0 otherwise, for $i, j \in \{1, 2, \ldots, 2^n - 1\}$.

Finally, let J denote the row-reversed identity matrix, which means the matrix whose *anti-diagonal* is all ones and the off-diagonals are all zeros. For example, $J\boldsymbol{e}_0 = \boldsymbol{e}_n$.

B.2 From P to P'

Lemma B.2.

$$\boldsymbol{P} = Q^{-1}\left(\boldsymbol{v} - UJ\boldsymbol{P'}\right) \ .$$

Let $\mathcal{D}$ be the linear projection operator from $\mathbb{R}^{n+1}$ to $\mathbb{R}^n$ which drops the first component when applied to a vector. Similarly, let $\mathcal{P}$ be the reverse operator of $\mathcal{D}$, in the sense that it maps vectors from $\mathbb{R}^n$ to $\mathbb{R}^{n+1}$ by prepending a zero component. Let $\underline{J}$ and $\underline{U}$ be the $(n+1)\times(n+1)$ dimensional counterparts of the $n\times n$ dimensional operators J and U ($\underline{U}$ counts from 0 to n instead of 1 to n). Note that $\mathcal{D}\mathcal{P} = I$, but $\mathcal{P}\mathcal{D} = I - \boldsymbol{e}_1\boldsymbol{e}_1^\top$.

Proposition B.3.

$$Q^{-1} = -\mathcal{D}\underline{JU}^{-1}\mathcal{P} \ .$$

Proof. We are going to prove this form $I = -Q\mathcal{D}\underline{JU}^{-1}\mathcal{P}$. For each $i,j\in[n]$ we have

$$\begin{aligned}
&\langle i|-Q\mathcal{D}\underline{JU}^{-1}\mathcal{P}|j\rangle\\
&= \sum_{\substack{k,\ell,m\in[n+1]\\ q\in[n]}} \langle i|-Q|q\rangle\ \langle q|\mathcal{D}|k\rangle\ \langle k|\underline{J}|\ell\rangle\ \langle \ell|\underline{U}^{-1}|m\rangle\ \langle m|\mathcal{P}|j\rangle\\
&= (-1)^j\left[\sum_{q\in[n]}(-1)^q\binom{n-q}{i}\binom{j}{n-q} - \binom{n}{i}\sum_{q\in[n]}(-1)^q\binom{j}{n-q}\right]\\
&= (-1)^j\left[\sum_{0\leqslant q\leqslant n-1}(-1)^q\binom{q}{i}\binom{j}{q} - \binom{n}{i}\sum_{0\leqslant q\leqslant n-1}(-1)^q\binom{j}{q}\right]\\
&= (-1)^j\left[\sum_{0\leqslant q\leqslant n-1}(-1)^q\binom{q}{i}\binom{j}{q} - (-1)^{n-1}\binom{n}{i}\binom{j-1}{n-1}\right]\\
&= (-1)^j\left[(-1)^j\binom{0}{i-j} - (-1)^n\binom{n}{i}\binom{j}{n} - (-1)^{n-1}\binom{n}{i}\binom{j-1}{n-1}\right]\\
&= (-1)^j\left[(-1)^j\binom{0}{i-j} - (-1)^n\binom{n}{i}\binom{j-1}{n}\right]\\
&= \binom{0}{i-j}
\end{aligned}$$

In which we used in the last equality the fact that $\binom{j-1}{n} = 0$ that holds since $j-1<n$. □

Lemma B.2.

$$\boldsymbol{P} = Q^{-1}\left(\boldsymbol{v} - UJ\boldsymbol{P'}\right) \ .$$

Proof. First, we show that $Q^{-1}\boldsymbol{v} = \boldsymbol{e}_n$, in other words $\boldsymbol{v} = Q\boldsymbol{e}_n$, which is easy to see since the ith element of $Q\boldsymbol{e}_n$ is simply $\binom{n}{i} - \binom{n-j}{i} = \binom{n}{i} - \binom{n-n}{i} = \binom{n}{i} - \binom{0}{i} = \binom{n}{i}$ since i is never 0. Then, we use $Q^{-1} = -\mathcal{D}\underline{JU}^{-1}\mathcal{P}$ shown in Proposition B.3 to show that

$$Q^{-1}UJ\begin{pmatrix}a_1\\ a_2\\ \vdots\\ a_n\end{pmatrix} = \begin{pmatrix}-a_2\\ -a_3\\ \vdots\\ -a_n\\ a_1+a_2+\cdots+a_n\end{pmatrix} \ ,$$

or equivalently that

$$\langle i|Q^{-1}UJ|j\rangle = \begin{pmatrix}0\\ n-i\end{pmatrix} - \begin{pmatrix}0\\ i-j+1\end{pmatrix} \ .$$

Note that $U = \mathcal{D}(\underline{U} - \boldsymbol{e}_1\mathbf{1}^\top)\mathcal{D}^\top$. Therefore, $\mathcal{P}U = (I - \boldsymbol{e}_1\boldsymbol{e}_1^\top)(\underline{U} - \boldsymbol{e}_1\mathbf{1}^\top)\mathcal{D}^\top$. Hence

$$\begin{aligned}
Q^{-1}UJ =& -\mathcal{D}\underline{JU}^{-1}\mathcal{P}UJ\\
&= \mathcal{D}\underline{JU}^{-1}(\boldsymbol{e}_1\boldsymbol{e}_1^\top - I)(\underline{U} - \boldsymbol{e}_1\mathbf{1}^\top)\mathcal{D}^\top J\\
&= \mathcal{D}\underline{JU}^{-1}\boldsymbol{e}_1\boldsymbol{e}_1^\top\underline{U}\mathcal{D}^\top J - \mathcal{D}\underline{J}\mathcal{D}^\top J \ .
\end{aligned}$$

Remarking that $\boldsymbol{e}_1\boldsymbol{e}_1^\top\boldsymbol{e}_1 = \boldsymbol{e}_1$, it is easy to show that the i,j element of $\mathcal{D}\underline{J}\mathcal{D}^\top J$ is 1 if $i = j-1$ and 0 otherwise. Afterwards, by observing that $\underline{U}^{-1}\boldsymbol{e}_1 = \boldsymbol{e}_1$, that is $\boldsymbol{e}_1$ is an eigenvector of $\underline{U}$, and $\boldsymbol{e}_1^\top\underline{U} = \mathbf{1}^\top$, then $\mathcal{D}\underline{JU}^{-1}\boldsymbol{e}_1\boldsymbol{e}_1^\top\underline{U}\mathcal{D}^\top J = \mathcal{D}\underline{J}\boldsymbol{e}_1\mathbf{1}^\top\mathcal{D}^\top J$, it is therefore straightforward to show that this expression evaluates to 1 when $i = n$ and 0 otherwise. □

B.3 From P' to Ψ

Lemma B.4.

$$\boldsymbol{\Psi} = UJ\boldsymbol{P'} \ .$$

Theorem B.5. *Let $S, F, \overline{F}$ be multisets. In particular, the multiset cardinalities $|S|, |F|, |\overline{F}|$ represent the sum of the multiplicities of their elements. Furthermore, the multiset complement operation $S \setminus \overline{F}$ subtracts the element multiplicities, such that the resulting multiset has no elements of nonpositive multiplicity. We use the notation $\{\!\{\cdots\}\!\}$ to denote multisets. Finally, let $\bigcup\overline{F}$ denote the* set $\{i | f\in\overline{F}\wedge i\in f\}$.

Let S be the multiset of n sets, $\{\!\{s_1, s_2, \ldots, s_n\}\!\}$. Then, given a multiset $\overline{F}\subseteq S$, define $F = \{\!\{s' | s\in S\setminus\overline{F}\wedge s' = s\setminus\bigcup\overline{F}\}\!\}$. Then the probability $P_t \overset{def}{=} P(S, n=|S|, t)$ that a bit position contains exactly t 1-bits across n Bloom filters, with fully uniform hashing is: $P(S,n,t) = 0$ when $t\notin\{0,1,\ldots,n\}$; $P(S,n,t) = \varphi^{|\bigcup S|}$ when $t=0$; $P(S,n,t) = 1 - \sum_{t'\neq n} P(S,n,t')$ when $t=n$; and $P(S,n,t) = \sum_{\substack{\overline{F}\subset S\\ |\overline{F}|=n-t}} P(\overline{F}, n-t, 0)P(F,t,t)$ otherwise;

in which $\varphi = 1 - 1/m$, and m is the size of each Bloom filter.

Proof. We prove the recurrence relation by proving each case separately.

1. The case in which $t \notin \{0, 1, \ldots, n\}$ is 0 by the definition of the problem.

2. The case in which $t = 0$ occurs when all elements in $\bigcup S$ fail to hash to a particular position. The probability that an element fails to hash to a particular position is the complement of the probability that it *does* hash to this position. For the case of fully uniform hashing, the latter probability is $1/m$, and the former is $\varphi = 1 - 1/m$. By fully uniform hashing, all these events are also independent and thus their conjunction is simply the product of individual probabilities, which is $\varphi^{k|\bigcup S|}$, in which k is the number of the hash functions used by the Bloom filter. We assume that $k = 1$ for the entire paper for simplicity of presentation. The interested reader can substitute a different value for k in this formula and the result will be valid.

3. The case in which $t = n$ is simply defined as the complement of all the other cases in which $t < n$ since the probability vector $\boldsymbol{P}$ sums to 1. Its components are mutually exclusive since a bit position be associated with exactly one value of t at a time.

4. The case in which $0 < t < n$ is the most interesting case. We first note that since S is a *multiset*, it is guaranteed that $|F| = |S \setminus \overline{F}|$ will always equal t, and thus the second parameter in the expression $P(F, t, t)$ is valid (since it should be equal to $|F|$). The event which we wish to compute the probability for is

$$\bigcup_{\substack{\overline{F} \subset S \\ |\overline{F}| = n-t}} \left\{ \left(\bigcap_{s \in \overline{F}} \text{Fails}(s) \right) \bigcap \left(\bigcap_{s \in F} \text{Succeeds}(s) \right) \right\} \ ,$$

in which $\text{Fails}(s)$ indicates the event that no item in the set s is hashed to the bit position under consideration, while $\text{Succeeds}(s)$ indicates the event that at least one element in s is hashed to it. Both $\left(\bigcap_{s \in \overline{F}} \text{Fails}(s)\right)$ and $\left(\bigcap_{s \in F} \text{Succeeds}(s)\right)$ are the intersection of non-independent events since the sets in F and $\overline{F}$ may share elements. However, by construction, F and $\overline{F}$ do not share any elements. Adding that to the fact that the hash function used is fully uniform and thus does not introduce dependencies between otherwise unrelated elements, we conclude that $\left(\bigcap_{s \in \overline{F}} \text{Fails}(s)\right)$ and $\left(\bigcap_{s \in F} \text{Succeeds}(s)\right)$ are independent events and consequently that the probability of their intersection is equivalent the product of their respective probabilities. Their probabilities have been shown in the beginning of this proof (*i.e.*, the cases in which $t = 0$ and $t = n$) to be equivalent to $P(\overline{F}, n - t, 0)$ and $P(F, t, t)$. It remains to show that the union is over mutually exclusive events and thus its probability is simply the sum of the probabilities of individual events. The case in which the union is over at most one event is trivial. Consequently, we assume that the union is over at least two events. Take any two distinct events in the domain of the union, an event indexed by $\overline{F}_1$ and the other indexed by $\overline{F}_2$. Since $\overline{F}_1$ and $\overline{F}_2$ are distinct multisets and are of the same positive cardinality (since $t < n$), then there exists a set s with positive cardinality c in $\overline{F}_1$ which has cardinality $c' < c$ in $\overline{F}_2$. Since F_2 is a function of the complement of $\overline{F}_2$ then, there is a subset s' of s which has positive cardinality $c'' \geqslant c - c' \geqslant 1$ in F_2. The event indexed by $\overline{F}_1$ then asserts that no element in s (and consequently in s') hashes to a particular bit position, while the event indexed by $\overline{F}_2$ asserts that s' contains at least one element which hashes to that particular bit position. Hence, the events are mutually exclusive (*i.e.*, if one of them occurred, the other does not). □

Conjecture B.6.

$$P_t = \sum_{0 \leqslant i \leqslant n} (-1)^{i+n-t} \binom{i}{n-t} \Psi_i \ . \tag{3}$$

Proof. This is a conjectured form of the formally-proven recurrence relation provided in Theorem B.5. □

Lemma B.4.

$$\boldsymbol{\Psi} = UJ\boldsymbol{P}' \ .$$

Proof. Follows directly from Conjecture B.6. □

B.4 From T to Φ

Lemma B.7. *Let the vector $\boldsymbol{T}$ be the "t-out-of-n distinct count" vector as defined in Definition 4.1. Then $\boldsymbol{T} = Q^{-1}\boldsymbol{\Phi}$. That is $T_i = \sum_{1 \leqslant j \leqslant n} (-1)^{n+i+j-1} \binom{j}{n-i} \Phi_j$.*

Proof. We proceed by showing that $\boldsymbol{\Phi} = Q\boldsymbol{T}$, *i.e.*,

$$\Phi_i = \sum_{1 \leqslant j \leqslant n} \left[\binom{n}{i} - \binom{n-j}{i} \right] T_j \ . \tag{4}$$

Uniqueness: In this proof, we will show the existence of a third linear operator Q mapping $\boldsymbol{T}$ to $\boldsymbol{\Phi}$, such that the following diagram commutes.

$$\begin{array}{ccc} \boldsymbol{V} & \xrightarrow{Z} & \boldsymbol{T} \\ {\scriptstyle R}\downarrow & & \downarrow{\scriptstyle Q} \\ \boldsymbol{C} & \xrightarrow{Z} & \boldsymbol{\Phi} \end{array}$$

From the diagram observe that $\boldsymbol{T} = Z\boldsymbol{V}$ and $\boldsymbol{\Phi} = ZR\boldsymbol{V}$, we solve for Q that satisfies $ZR = QZ$. The rank of the left-hand side (LHS) is $\mathrm{Rank}(ZR) = \min\{\mathrm{Rank}(Z), \mathrm{Rank}(R)\} = n$ and thus the we have n^2 linearly-independent equations to solve. Since Q has n^2 unknowns, then the system has a unique solution.

Existence: Thus $ZR = QZ$ reduces that for each $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, 2^n - 1\}$ the following must hold $Q_{i,|\chi(j)|} = \sum_{1 \leqslant k \leqslant n} [|\chi(k)| = i][\chi(k) \cdot \chi(j) > 0]$, in which $[p]$ is 1 if the predicate p is true, and 0 otherwise. From all $2^n - 1$ possible values of $\chi(k)$, only $\binom{n}{i}$ of them satisfy $|\chi(k)| = i$. Of of them, those which do *not* satisfy $\chi(k) \cdot \chi(j) > 0$ have zeros *jointly* in $|\chi(j)|$ positions, therefore the i ones that $\chi(k)$ have must be chosen among $n - |\chi(j)|$ specific positions, and there are $\binom{n-j}{i}$ ways of doing that. Consequently, the right-hand side equals $\binom{n}{i} - \binom{n-|\chi(j)|}{i}$, and crucially, does not depend on j itself, or $\chi(j)$, but solely on $|\chi(j)|$, which means the left-hand side is well-defined. □

B.5 From Φ and Ψ to Bias

Lemma B.8.

$$\boldsymbol{\Psi} = (\boldsymbol{\Phi} - \boldsymbol{e}_T) \ln \varphi + \boldsymbol{v} \ .$$

Proof. For all $t \in \{1, 2, \ldots, n\}$ we know by definition that $\Psi_t \triangleq \sum_j \varphi^{a_j}$ for some sequence $\boldsymbol{a}$ of length $\binom{n}{t}$. Rewriting φ^x as $\exp(x \ln \varphi)$ we get $\Psi_t = \sum_j \exp(a_j \ln \varphi)$, which, by Taylor expansion around 0, turns to $\Psi_t = \sum_{i \geqslant 0} (\ln \varphi)^i (\sum_j a_j^i)/i!$. Then by expanding the first two terms in the series $\Psi_t = (\sum_j a_j^0) + (\ln \varphi)(\sum_j a_j) + \sum_{i \geqslant 2} (\ln \varphi)^i (\sum_j a_j^i)/i!$. Therefore, with the convention that $0^0 = 1$ and the fact that by definition $\Phi_t \triangleq \sum_j a_j$ and $(\boldsymbol{e}_T)_t \triangleq -\sum_{i \geqslant 2} (\ln \varphi)^{i-1} (\sum_j a_j^i)/i!$

$$\begin{aligned}
\Psi_t &= \binom{n}{t} + (\ln \varphi)\Phi_t + \sum_{i \geqslant 2} (\ln \varphi)^i (\sum_j a_j^i)/i! \\
&= v_t + (\ln \varphi) \left[\Phi_t + \sum_{i \geqslant 2} (\ln \varphi)^{i-1} (\sum_j a_j^i)/i! \right] \\
&= v_t + (\ln \varphi) \left[\Phi_t - (\boldsymbol{e}_T)_t \right] \ .
\end{aligned}$$ □

B.6 From Bias to T and P

Lemma B.1 (Bias relation to $\boldsymbol{T}$ and $\boldsymbol{P}$)**.**

$$\boldsymbol{e}_T = Q\boldsymbol{T} + Q\boldsymbol{P} / \ln \varphi \ .$$

Proof.

$$\begin{aligned}
Q\boldsymbol{T} + Q\boldsymbol{P}/\ln\varphi &= Q\boldsymbol{T} + (\boldsymbol{v} - UJ\boldsymbol{P}')/\ln\varphi && \text{Lem B.2} \\
&= Q\boldsymbol{T} + (\boldsymbol{v} - \boldsymbol{\Psi})/\ln\varphi && \text{Lem B.4} \\
&= \boldsymbol{\Phi} + (\boldsymbol{v} - \boldsymbol{\Psi})/\ln\varphi \ . && \text{Lem B.7}
\end{aligned}$$

which simplifies to $\boldsymbol{e}_T$ when we substitute for $\boldsymbol{\Psi}$ using Lemma B.8. □

C Proof of Additive Error

Theorem 4.4 (Estimator's Additive Error)**.** *Let Q^{-1} is the matrix whose i, j entry is $(-1)^{n+i+j-1}\binom{j}{n-i}$ for $1 \leqslant i, j \leqslant n$ and φ be equal to $1 - 1/m$, for m being the size of the Bloom filters. Then the additive error of $\widehat{\boldsymbol{T}}$ (cf. Equation (1) in Definition 4.3) is:*

$$\boldsymbol{T} - \widehat{\boldsymbol{T}} = Q^{-1}\boldsymbol{e}_T + (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi \ . \qquad (2)$$

Proposition C.1.

$$\boldsymbol{P} = \mathbb{E}[\boldsymbol{D}] \ .$$

Proof. Let $[B_i = t]$ equal 1 if $t = \sum_{j=1}^n b_i^j$ (*cf.* Definition 4.2), and 0 otherwise. Then, for each t:

$$\mathbb{E}[D_t] = \mathbb{E}\left[\frac{1}{m}\sum_i [B_i = t]\right] = \frac{1}{m}\sum_i \mathbb{E}\left[[B_i = t]\right] = P_t \ . \square$$

Proof of Theorem 4.4.

$$\begin{aligned}
\widehat{\boldsymbol{T}} &= \widehat{\boldsymbol{D}}/\ln(1/\varphi) \\
&= (\boldsymbol{D} + \boldsymbol{e}_F)/\ln(1/\varphi) \\
&= (\mathbb{E}[\boldsymbol{D}] + \boldsymbol{e}_H + \boldsymbol{e}_F)/\ln(1/\varphi) \\
&= (\boldsymbol{P} + \boldsymbol{e}_H + \boldsymbol{e}_F)/\ln(1/\varphi) && \text{Prop C.1} \\
&= \boldsymbol{P}/\ln(1/\varphi) + (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln(1/\varphi) \\
&= -\boldsymbol{P}/\ln\varphi - (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi \\
&= \boldsymbol{T} - \boldsymbol{T} - \boldsymbol{P}/\ln\varphi - (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi \\
&= \boldsymbol{T} - Q^{-1}\boldsymbol{e}_T - (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi \ . && \text{Lem B.1}
\end{aligned}$$

D Proof of Upper Bound

Lemma 4.5 (Upper Bound)**.** *If the un-flipping step estimator used was that of [5], then the upper bound Γ_ξ on the error $\|\boldsymbol{T} - \widehat{\boldsymbol{T}}\|$, ignoring $\boldsymbol{e}_H$, is, for sufficiently large m:*

$$\Gamma_\xi = \frac{\|Q^{-1}\boldsymbol{K}(x \mapsto x^2)\|}{2m} + O(\eta^{-n}) \frac{2\xi\sqrt{-\ln(\beta)\ln(n+1)}}{-\ln(\varphi)\sqrt{2m}},$$

with probability at least $1-\beta$, in which all the norms are the max norm; $\|\boldsymbol{x}\|_\infty \triangleq \max_i |x_i|$, or its induced norm for matrices, and $\xi \in (0,1)$ is the precomputed multiplier described in [5].

Lemma D.1.

$$(\boldsymbol{e}_T)_t \sim \frac{1}{2m}\|\boldsymbol{a}\|_2^2 = O(1/m) \ .$$

Proof. We will proceed by showing that $\lim_{m\to\infty}(\boldsymbol{e}_T)_t m = \frac{1}{2}\|\boldsymbol{a}\|_2^2$.

$$\begin{aligned}
(\boldsymbol{e}_T)_t &= \Phi_t + \frac{v_t - \Psi_t}{\ln\varphi} \\
(\boldsymbol{e}_T)_t m &= \Phi_t m + \frac{v_t - \Psi_t}{\ln\varphi} m \\
&= (\sum_j a_j)m + \frac{v_t - (\sum_j \varphi^{a_j})}{\ln\varphi} m \\
&= (\sum_j a_j)m + \frac{v_t - (\sum_j (1-1/m)^{a_j})}{\ln(1-1/m)} m
\end{aligned}$$

Replace variables: $m' = 1/m$:

$$\begin{aligned}
&= (\sum_j a_j)/m' + \frac{v_t - (\sum_j (1-m')^{a_j})}{\ln(1-m')}/m' \\
&= \frac{(\sum_j a_j)\ln(1-m') + v_t - (\sum_j (1-m')^{a_j})}{m'\ln(1-m')}
\end{aligned}$$

We use L'hôpital's rule twice to find the limit at $m' \to 0$, since the limit exists for the numerator and denominator separately.

$$\frac{-(\sum_j a_j)/(1-m')^2 - (\sum_j a_j(a_j-1)(1-m')^{a_j-2})}{2/(m'-1) - m'/(1-m')^2}$$

which at $m' = 0$ (direct substitution) yields:

$$\frac{1}{2}((\sum_j a_j) + (\sum_j a_j(a_j-1))) = \frac{1}{2}\sum_j a_j^2 \ . \qquad \square$$

Proof of Lemma 4.5.

$$\begin{aligned}
\|\boldsymbol{T} - \widehat{\boldsymbol{T}}\| &= \|Q^{-1}\boldsymbol{e}_T + (\boldsymbol{e}_H + \boldsymbol{e}_F)/\ln\varphi\| \\
&\approx \|Q^{-1}\boldsymbol{e}_T + \boldsymbol{e}_F/\ln\varphi\| \\
&\leqslant \|Q^{-1}\boldsymbol{e}_T\| + \|\boldsymbol{e}_F\|/\ln(1/\varphi) \ .
\end{aligned}$$

Then, the upper bound on $\|n_F\|$ from [5], which holds with probability at least $1-\beta$, asserts that $\|\boldsymbol{e}_F\| \leqslant O(\eta^{-n})\sqrt{2\ln(n+1)\ln(1/\beta)/m}$. Finally, Lemma D.1 shows that for large m, $\boldsymbol{e}_T$ asymptotically approaches $\boldsymbol{K}(x \mapsto x^2)/2m$. $\square$

E Example

An example of how the error decomposes as described in Theorem 4.4 appears later in Figure 1. The figure shows that the error due to deviation of hashing from its expectation is negligible. Moreover, it also shows that as expected the error is dominated by bias when m is small, and by variance (due to the differentially-private noise), when m is large. As one source of error increases and another decreases when we vary the Bloom filter size, m, the choice of m should be taken with care to set a good trade-off between both sources of error and minimize the overall error. As a numerical example for the case of Figure 1, and using the formula for $\boldsymbol{e}_F$ from [5] we have, using Theorem 4.4: $(\boldsymbol{e}_T)_1 = \Phi_1 + (v_1 - \Psi_1)/\ln\varphi = |A| + |B| + (n - \varphi^{|A|} - \varphi^{|B|})/\ln\varphi = 50000 + (2 - 2\varphi^{25000})/\ln\varphi$, and $(\boldsymbol{e}_T)_2 = \Phi_2 + (v_2 - \Psi_2)/\ln\varphi = |A \cup B| + (n - \varphi^{|A\cup|})/\ln\varphi = 40000 + (1 - \varphi^{40000})/\ln\varphi$, We are leaving m unspecified so we can optimize for it. Then $(Q^{-1}\boldsymbol{e}_T)_2 = (-\varphi^{40000} + 2\varphi^{25000} - 10000\ln\varphi - 1)/\ln\varphi$. Next, from [5], letting the probability β that the bound does not hold being 0.1, $|(\boldsymbol{e}_F)_2| \leqslant \|A^{-1}\|_\infty\sqrt{2\ln(n+1)\ln(1/\beta)/m} = (3\eta^{-2} - 1)\sqrt{\ln(3)\ln(10)/2m}$, in which we are also letting η be free. Lastly, ignoring $\boldsymbol{e}_H$ as insignificant for our purpose, we have the additive error for the intersection be: $T_2 - \widehat{T}_2 \leqslant (Q^{-1}\boldsymbol{e}_T)_2 + (\boldsymbol{e}_F)_2/\ln\varphi = \frac{2\varphi^{25000} - \varphi^{40000} - 1 + (3\eta^{-2}-1)\sqrt{\frac{\ln 3 \ln 10}{2m}}}{\ln\varphi} - 10000$. Minimizing this formula directly, for $\varepsilon = 1$ yields the optimal m as 2^{24}, which does not agree with Figure 1. This is because the expression we used for $\boldsymbol{e}_F$ is a very *loose* upper bound, which while gives an indication for an upper bound on the error, is not helpful when we balance a trade-off. For this purpose, we need a tighter estimate of the error. Luckily, [5] describes an empirical way to obtain such a tighter method, by effectively using $T_2 - \widehat{T}_2 \approx (Q^{-1}\boldsymbol{e}_T)_2 + \xi(\boldsymbol{e}_F)_2/\ln\varphi$, for ξ empirically estimated to be approximately 1/4. Using this formula, we obtain the optimal m as 2^{19}, which agrees with Figure 1. Remark that the knowledge of $|A|, |B|$, and $|A \cup B|$ is needed to apply this formula, which are not available in real-life scenarios because these values are private data for whose estimation BLIPs are used. However, if we have a rough estimate of the order of magnitude of these values, the formula may still be used, bearing in mind that bets are off if the original estimate was grossly mistaken.