Distributed Privacy-Aware User Counting

Florian Tschorsch   Björn Scheuermann

Telematics Group
University of Würzburg, Germany
{tschorsch, scheuermann}@informatik.uni-wuerzburg.de

1 Introduction

In this work, we consider the question of how to determine—in a privacy-preserving way—the number of distinct users that have contacted an Internet service. This question is particularly challenging in a distributed setting. In order to protect the users' privacy, the service operators should obviously not exchange information about user identities. Then, however, it becomes challenging to determine the total number of distinct users in the overall system without counting users more than once.

One key motivation for our work is the anonymization network Tor [3], and the problems the Tor project is currently facing when it comes to estimating the number of Tor users [6]. Tor node operators definitely shouldn't exchange (or even record) explicit information about locally observed IP addresses of Tor users. Nevertheless, it would still be very interesting to obtain statistics about the total number of distinct users of the system.

From a more general perspective, we are looking for a way to obtain the number of distinct elements in a multiset of user IDs (e. g., in case of Tor, the IP addresses of users). These user identifiers can occur multiple times at the same service entry point, and they can occur at multiple entry points. No single point, however, will in general be able to see the "whole" multiset.

In order to tackle this challenge, we start from an existing algorithm for probabilistic counting [5]. This algorithm allows for estimating the total number of distinct elements in a sequence of user IDs without keeping track of the IDs that have already been seen. It also provides an interesting basis for obtaining such estimates in a distributed fashion at multiple observation points. What is more, this approach already mitigates the privacy problems to some extent. However, it does not fully eliminate the problems: in the worst case, it is still possible to conclude with arbitrarily high probability that a specific user was present. We thus proceed with modifications which avoid that an attacker can gain an arbitrary amount of knowledge. Here, the knowledge gain of an attacker can be considered as a privacy metric.

2 Naive Distributed Counting

In this section, we start by introducing the algorithmic basis of our approach—FM sketches—, and how they could be used to determine the total number of users of a distributed Internet service like Tor. In this first step, the application of FM sketches will be quite naive, and we will see that it comes with severe privacy problems.

2.1 FM sketches

FM sketches, introduced in [5], are an algorithmic mean to estimate the cardinality of a multiset M of n elements. Their low computational effort was the key original motivation behind them. However, as we will see, they also exhibit other very interesting properties, which make them a promising basis for solving the problem considered here. We will now outline the key ideas behind FM sketches; more details can be found in [5].

An FM sketch is based on a bit vector $S = (s_1, \ldots, s_w)$, $w \geq 1$, which is initialized to zero. We also need a hash function h_1 with geometrically distributed positive integer output, where the probability that $h_1(x) = j$ (with $j \geq 1$) for any randomly picked element x equals $P(h_1(x) = j) = 2^{-j}$. Each element $x \in M$ is hashed using h_1. The hash value is interpreted as an index in S, and the corresponding bit $s_{h_1(x)}$ is set to one. This leads to a bit pattern in S with many 1-bits on the left and many 0-bits on the right.

Flajolet and Martin found that a good estimate for the number of distinct elements can be obtained from the length of the uninterrupted, initial sequence of ones in S, i. e., from $Z := \min\left\{ j \in \mathbb{N}_0 \mid s_{j+1} = 0 \right\}$. There is a constant factor $\varphi \approx 0.77351$ such that $n \approx 2^Z/\varphi$, so that estimates can be obtained on this basis.

The accuracy can be improved by using multiple sketches in parallel. The respective technique is called Probabilistic Counting with Stochastic Averaging (PCSA) in [5]. Each element is first mapped to one of the sketches by using a uniformly distributed hash function h_2, and is then added to this (and only this) sketch. In the following, we will use this variant. If m sketches are used with PCSA, then the estimate for the total number of distinct items added is given by $C = m \cdot 2^{\sum_{i=1}^{m} Z_i/m}/\varphi$, where Z_i is the number of leading 1-bits in the i-th sketch. One can identify a PCSA set with an $m \times w$ matrix, where each row is a standard FM sketch. For a sufficiently large number of elements, PCSA yields a standard error of approximately $0.78/\sqrt{m}$ [5]. Increasing m thus results in a higher estimation accuracy.

Multiple FM sketches (and likewise PCSA matrices) can be merged to obtain the total number of distinct elements added to at least one of them by a simple bit-wise OR. Observe that combining the FM sketch with all elements of a multiset A and the FM sketch with all elements of another, possibly overlapping multiset B using bit-wise OR produces an FM sketch that is identical to the sketch of multiset $A \cup B$. Elements present in both A and B will not be counted twice, since the respective bit will always have value 1 in both sketches. This duplicate insensitivity trait allows us to perform distributed user counting.

2.2 Applying FM sketches for user counting

Now let us look at how we could apply FM sketches to distributed user counting. Each service entry point might maintain a PCSA matrix with pre-configured dimensions $m \times w$. m and w as well as the hash functions h_1, h_2 are agreed on by the service operators in advance. When a user with ID u contacts the service at one of the entry points, u is hashed into the sketch matrix and the respective bit is set locally. Clearly, if the same user contacts the entry point more than once, the user will not be counted again, since the respective bit is already set. By evaluating the sketch generated by a single entry point, we therefore obtain an estimate for the number of distinct users who contacted the respective mirror.

If the sketch matrices from multiple service entry points are collected and merged by a bit-wise logical OR, this results in a sketch for the total number of *distinct* users contacting *at least one* of the respective service entry points. The users are therefore counted in a duplicate insensitive way.

In the specific use case of Tor, the counting operation could be performed at the directory mirrors. As already argued in [6], this is a reasonable design, because each Tor user contacts at least one mirror during bootstrapping.

Since the user IDs are not exchanged directly, and typically many different user IDs are mapped to the same bit, one might expect that the sketch does not reveal much information about which specific users contacted the service. This conclusion is treacherous, though. Recall that hash function h_1, which selects the column in the sketch, is geometrically distributed. Consequently, there are relatively few user IDs mapped to bits more towards the right hand side of the sketch. If an attacker observes *such* a bit being set, then it becomes suddenly very likely that a specific user has indeed been present in the system. In the extreme case, an attacker knows for sure that only one single out of all possible user IDs maps to a specific location in the sketch; if this bit is set to one, the attacker can be sure that the user has contacted the service.

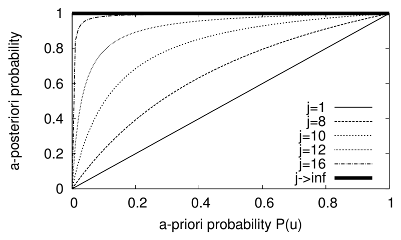
3 Privacy-Aware User Counting

In order to improve on the worst case behavior, we propose a perturbation technique. Specifically, a service entry point will proceed just as discussed above and enable bits according to the hash coordinates of the locally observed user IDs. In addition, though, each bit in the sketch matrix will be set to one with a fixed, configured probability r. While randomness has been used before in privacy-aware data bases, e. g. in [1, 4, 8], existing approaches do not allow to obtain the statistics we are interested in.

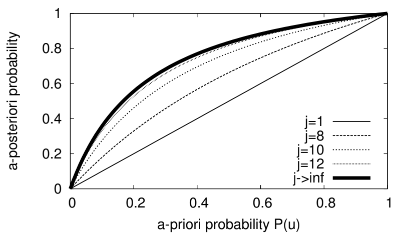
By adding these randomly switched on bits, we add an additional source of "vagueness": The attacker cannot make a definite decision whether a set bit has been set by a corresponding user or whether it has been switched on at random. The fact that we switch additional bits on uniformly at random has two important implications. First, we set bits on the right hand side in our sketch with non-negligible probability, where, as discussed above, an attacker is otherwise able to gain very substantial knowledge. Second, when it comes to extracting estimates for the number of distinct users, the uniform distribution of the randomly set bits will allow us to separate their effects from the geometrically distributed bits introduced by "real" users.

We now analyze this intuitive explanation of our approach and contrast it with the naive user counting. Subsequently, we give a short note on how accurate estimates can still be obtained. The question that we consider is how "sure" can an attacker be that a specific user has contacted the service. This is expressed by the *a-posteriori probability* of the user being there, after the information from the sketch is known. A probability of one means that the user has contacted the service for sure, a probability of, e. g., 0.5 means that it is equally likely that the user has or hasn't been there.

Clearly, this a-posteriori probability for the presence of a given user depends on the attacker's *a-priori knowledge*: how certain has the attacker been about the user being active *before* taking the information in the sketch into account? If the attacker, for



(a) Without perturbation ($r = 0$)



(b) With perturbation ($r = 0.2$).

Fig. 1. Privacy evaluation: knowledge gain ($m = 8$, $n = 1000$).

whatever reason, has been 99.9% sure that the user has been active, then the a-posteriori probability will be at least 99.9%, as the attacker will not "lose" knowledge by looking at the sketch. We therefore always look at the a-posteriori knowledge *depending on the attackers a-priori knowledge*. The smaller the difference between the attacker's a-priori and a-posteriori knowledge, the less information an attacker can gain.

We are now interested in the probability $P(u \mid (i,j), r)$ that the user has used the service after the attacker has learned that bit (i,j) is set. Here, (i,j) denotes u's position in the sketch. r is our perturbation probability; consequently, $r = 0$ corresponds to the naive case without perturbation. In order to determine the a-posteriori probability, we start from the probability that a specific bit is set given that there was a total of n users and given u's a-priori probability $P(u)$ of being present. Following the definition of FM sketches in Sec. 2.1, this is $P((i,j) \mid n, r) = 1 - \left(1 - 1/(m \cdot 2^j)\right)^n \cdot (1 - P(u)) \cdot (1 - r)$. We then apply Bayes' theorem [2] and obtain

$$P(u \mid (i,j), r) = \frac{P(u)}{1 - \left(1 - \frac{1}{m \cdot 2^j}\right)^n \cdot (1 - P(u)) \cdot (1 - r)}. \tag{1}$$

One can see that the a-posteriori probability depends on several parameters, including j, n, m, r, and $P(u)$. In particular, the equation supports our intuition that bits to the right, to which a lower number of users are mapped, are more problematic: the higher the column index j, the more information an attacker gains if the bit is actually set.

Since our aim must be to protect the privacy of *all* users, we have to take the worst case into account. This can be done by taking the limit of the a-posteriori probability for $j \to \infty$. This limit is given by

$$\lim_{j \to \infty} P(u \mid (i,j), r) = \frac{P(u)}{P(u) + r - r \cdot P(u)}. \tag{2}$$

Observe that for $r = 0$ the limit turns out to be equal one for $P(u) > 0$. However, for $r > 0$ the a-posteriori probability does no longer converge to 1; there is a significant remaining uncertainty for an attacker even for large j. Consequently, we achieved our aim of mitigating the worst case. This relation is shown in Figure 1 with and without perturbation for varying j.

Now, the question arises how to still calculate accurate results despite the randomly added bits. Using the standard FM sketch evaluation formula above would lead to massive estimation errors, since the additional bits may increase the length of the initial sequence of ones in a sketch.

In [7], a related problem occurred. Even though the reasons for the perturbations and the problem setting are in fact quite different, the problem can be tackled with similar means. In short, the constant φ in Flajolet and Martin's estimation formula can be adapted according to the probability r. Along similar lines as in [7], the new constant φ_r can be obtained from $\varphi_r = \lim_{n\to\infty}(2^{E[Z|n,r]}/n)$, where $E[Z \mid n, r]$ is the (easily derived) expected value of Z for n distinct elements and additional bits switched on uniformly at random with probability r. If φ_r is used in the role of φ, the estimation error is compensated. For a deeper understanding we refer to [7].

4 Conclusion

We presented a methodology to count users based on their observed user IDs in a distributed and privacy-preserving manner. The algorithmic properties of FM sketches provide a way to deal with duplicate occurrences of user IDs, due to users contacting multiple times the same and/or different service entry points.

However this naive approach has severe shortcomings with respect to user privacy, at least in the worst case. We showed that this can be overcome by a perturbation technique that sets additional bits to one uniformly at random. In order to still calculate decent estimations by our modified FM sketches, we showed how the evaluation methodology can be adjusted. With respect to the specific use case of Tor user counting, our proposed method could be applied to estimate the number of Tor users at high accuracy, without compromising anonymity.

Acknowledgments

The authors thank the German Research Foundation (DFG) for the support.

References

1. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: SIGMOD '00. pp. 439–450 (May 2000)
2. Bayes, T.: An essay towards solving a problem in the doctrine of chances. Phil. Trans. of the Royal Soc. of London 53, 370–418 (1763)
3. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: USENIX Security '04. pp. 303–320 (Aug 2004)
4. Du, W., Zhan, Z.: Using randomized response techniques for privacy-preserving data mining. In: KDD '03. pp. 505–510 (2003)
5. Flajolet, P., Martin, G.N.: Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences 31(2), 182–209 (Oct 1985)
6. Hahn, S., Loesing, K.: Privacy-preserving ways to estimate the number of tor users. Tech. rep. (Nov 2010)
7. Lieven, P., Scheuermann, B.: High-speed per-flow traffic measurement with probabilistic multiplicity counting. In: INFOCOM '10 (Mar 2010)
8. Mishra, N., Sandler, M.: Privacy via pseudorandom sketches. In: PODS '06. pp. 143–152 (Jun 2006)