How (Not) to Apply Differential Privacy in Anonymity Networks

Scott E. Coull
RedJack, LLC.
scott.coull@redjack.com

Differential privacy is a particularly appealing way of defining privacy due to its ability to handle adversaries with arbitrary auxiliary information at their disposal [1]. In essence, using differential privacy should imply that we do not have to worry about the external databases or specialized knowledge that an attacker has access to. Meanwhile, one of the biggest challenges for anonymity networks, like Tor, lies in trying to account for the variety of traffic features that may be used to attack the unlinkability of sessions or the anonymity of users. Therefore, developing anonymity networks that meet some variant of the differential privacy definition would appear to be an ideal way to address all of the potential sources of information leakage in a single security model.

In this paper, however, I will discuss why I believe that differential privacy and its standard variants are unsuitable for computer networks in general, and anonymity networks in particular. The crux of the argument rests on an implicit assumption made within the differential privacy definition; namely, that each row in the database being protected is independent of all other rows (*i.e.,* there are no correlations or other semantic relationships in the data). This non-obvious assumption interacts with the inherent structure induced by network protocols and user activities to render differential privacy ineffective on the vast majority of network traffic. This is true whether the data is released non-interactively via packet traces, or interactively as a means of modeling the security of anonymity networks.

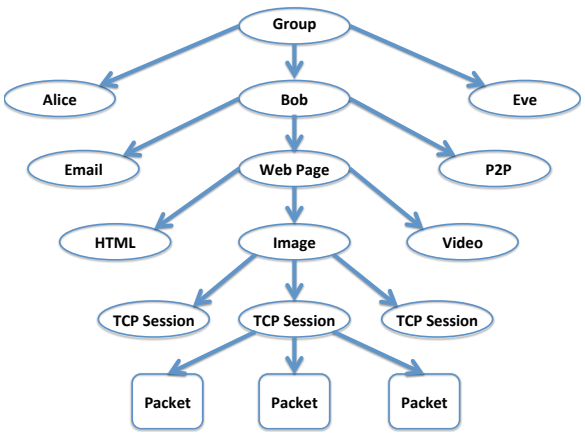

Figure 1: Example network traffic ontology.

The following sections outline some unique properties of network traffic and discuss their implications on achieving differential privacy. To conclude, I present a relatively new privacy definition framework proposed by Kifer and Machanavajjhala, called Pufferfish [3], and briefly sketch how it might be parameterized for the setting of anonymity networks. Pufferfish naturally subsumes the notion of differential privacy and offers a similar level of robustness to arbitrary auxiliary information, albeit under explicitly-defined assumptions about the protected data (*e.g.,* the distribution it is drawn from). As such, it would seem that Pufferfish may give us the opportunity to customize a definition similar to differential privacy, yet capable of accommodating the structure found in network traffic.

Correlations in Network Traffic. The complexity of computer network traffic is quite unlike any other type of privacy-sensitive data. Traffic on most networks is driven by convoluted interactions among users, applications, and ephemeral network conditions. Intuitively, we can think about the network data as a fine-grained record of the interactions among various *objects* whose representations are dictated by the underlying network protocols. These objects may include individual packets, TCP sessions, application objects (*e.g.,* web pages), computers, users, and even groups of users.

The relationships among most of these objects may be represented in an *ontological hierarchy* based on our understanding of the data, which may be derived from well-defined network protocol specifications or, due to their complexity, inferred through observation (*e.g.,* behavior of users). Figure 1 shows one such ontology where the influence of groups or users is less quantifiable than the clear relationship between packets and TCP sessions. In some sense, the ontology represents an observer's *view* of the network traffic based on their knowledge and assumptions.

The ontology is important in considering the privacy of the data because *each object carries some information about both its ancestors and descendants in the hierarchy.* The presence of a particular user, for instance, is implicitly encoded in each packet created due to her influence. Traffic analysis attacks exploit this ontology to take information from apparently innocuous packets, work back up the hierarchy using known relationships, and infer the presence of application objects or even the user's identity.

Moreover, the ontology underscores the idea that *there are many different ways to define the same objects*. The failure of traffic analysis countermeasures can often be traced to the defender having a weaker (or less accurate) ontology than the attacker. A weaker ontology means that the defender is incapable of protecting the object relationships she does not know about. One example of this asymmetry is when an attacker uses a more complex statistical model (*e.g.,* larger n-gram size) than the defender, and as a result that attacker has a more accurate understanding of the underlying relationships. To successfully achieve any comprehensive form of privacy in this space we must (1) have a more accurate ontology for the network traffic than our adversary, and (2) appropriately hide all of the known objects and relationships in that ontology.

Implications for Differential Privacy. Having established some of the basic structural properties of network traffic, we now examine how those properties interact with the notion of differential privacy. It is important to recognize that differential privacy was developed with databases of health or census records in mind. In these scenarios, each record is associated with only a single person and, therefore, clearly uncorrelated to all other records. This allowed for differentially private data release mechanisms whose outputs did not change much due to the presence or absence of any one person.

Kifer and Machanavajjhala explored the implications of the implicit independence assumption made by differential privacy and demonstrated several situations where correlations in the data lead to violations of privacy [2]. One of the simpler cases considers a differentially private database containing edges from a social network graph. Kifer *et al.* showed that the presence or absence of a single edge in the graph is amplified because it can significantly alter the growth of the social network. Specifically, the presence of a single edge would determine the degree to which two independent communities in the graph merged, and so the differentially private count of cross-community edges would signal the existence of the edge.

Unfortunately, the relationships found within network data can be used in exactly the same way to infer the presence of various objects despite the use of differential privacy. To see how this might lead to problems, let us consider the relationship between TCP sessions and packets. Every TCP session must complete a three-way handshake before any data is transmitted, and therefore SYN, SYN/ACK, and ACK packets are tightly correlated with all other packets in the session. A simple test for the presence or absence of the handshake packets would be to issue a differentially private query for the sum of the bytes in two packet databases. If the TCP session associated with the handshake packet transferred a sufficient number of bytes, the difference in the results of the two queries would be enough to distinguish the databases.

This same line of reasoning can be applied throughout the ontology with similar effect: a TCP session downloading an HTML page influences all other HTTP queries for that same web page, and the absence of the web page would impact DNS queries and other user behaviors. These problems can be mitigated by adjusting the ϵ parameter to be sufficiently small, however this would translate to significant overhead in the case of anonymity networks since, ostensibly, the only "noise" that could be added in that scenario would be dummy traffic and padding. More to the point, it is not clear what setting of ϵ would be sufficient to protect the traffic (and associated objects) sent over the anonymity networks.

Moving Forward. While the standard formulations of differential privacy appear to be unsuitable for use in anonymity networks, a new privacy definition framework by Kifer and Machanavajjhala, called Pufferfish [3] offers us a potential path forward. The concept behind the proposed framework is that privacy definitions often need to be customized to the requirements and assumptions of their application domain, as illustrated in the examples from the previous section. The framework allows a domain expert (not necessarily a privacy expert) to specify the secrets in the data that they wish to protect and some assumptions about the way the data was generated, such as correlations or other structures that might exist.

More formally, the domain expert is tasked with defining a set of secrets $\mathbb{S}$, a set of mutually exclusive pairs of secrets $\mathbb{S}_{pairs} \subseteq \mathbb{S} \times \mathbb{S}$, and a set of probability distributions $\mathbb{D}$, called data evolution scenarios. The evolution scenarios essentially describe the ways the attacker might think that the data was generated, similarly to the ontology from the earlier section. As an example, we could define sender anonymity by setting the secrets to be the set of all possible senders $\mathbb{S} = \{u_i : i = 1 \dots N\}$ and the discriminative pairs to be the event that each sender is or is not in the data $\mathbb{S}_{pairs} = \{(u_i, \neg u_i) : i = 1 \dots N\}$. Setting the data evolution scenarios requires specific consideration for the anonymity network's protocol outputs and may be difficult to specify if the protocol was not designed with a specific distribution in mind. Assuming we are able to derive a set of distributions $\mathbb{D}$ for the network traffic, then the protocol (denoted as function $\mathbb{M}$ applied to messages) satisfies ϵ-SenderAnonymity$(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ if the following holds:

$$P(\mathbb{M}(Msgs) = \omega | u_i, \Theta) \leq e^{\epsilon} P(\mathbb{M}(Msgs) = \omega | \neg u_i, \Theta) \quad (1)$$

$$P(\mathbb{M}(Msgs) = \omega | \neg u_i, \Theta) \leq e^{\epsilon} P(\mathbb{M}(Msgs) = \omega | u_i, \Theta) \quad (2)$$

References

[1] Cynthia Dwork. Differential Privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming*, pages 1–12, 2006.

[2] Daniel Kifer and Ashwin Machanavajjhala. No Free Lunch in Data Privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, pages 193–204, 2011.

[3] Daniel Kifer and Ashwin Machanavajjhala. A Rigorous and Customizable Framework for Privacy. In *Proceedings of the 31st Symposium on Principles of Database Systems*, pages 77–88, 2012.