Task 1.3.2: Software for disclosure controls

The starting effort for this task falls under Task 1.3.2.1: Understand gaps between privacy-preservation techniques and network and security research needs.

There are several classes of disclosure controls, including differential privacy (DP), secure multi-party computation (SMPC), homomorphic encryption and synthetic datasets. Our initial focus has been DP because we believe that the issues around this method are the more challenging.

During the first year, we have followed two paths to make progress on our understanding of DP. The first is to develop a sufficient understanding of the fundamentals behind the variants of DP, and the other is to identify existing software packages that we will evaluate in year two of the project.

Background on DP

From a practical perspective, DP offers great promise and raises great challenges. We do not attempt to reproduce here the mathematical foundations of DP, which have been well-documented elsewhere. DP provides privacy protection by adding noise to the result of a query. (There are other modes, but we focus on this one, which involved a trusted curator that has access to the raw data and assigns the necessary amount of noise to the query result.) DP provides a mathematical link between the amount of noise added and a parameter ε, which abstractly characterizes the *privacy loss* from releasing the results of a query to the public.

However, there are four fundamental problems that we will encounter in applying DP to a problem:

1. The difficulty of setting ε, the privacy loss parameter.
2. The difficulty of disentangling all that we wish to learn about the dataset into a set of suitable queries. If we can pose queries that draw on distinct subpopulations of the data, the total privacy loss will be lower.
3. The difficulty of identifying the so-called *sensitivity* of the queries, which differential privacy defines as the contribution of one person's data to a given *query*, rather than the contribution of a measurement of one person's data to the database.
4. The relative immaturity in the DP literature in working with certain kinds of data, such as time-series data.

A principal challenge when using DP is that the literature provides minimal guidelines for how to translate the value of ε into a measure of potential harm from loss of privacy that a policy-maker can use in any practical way. The literature on DP states that setting ε is a matter of policy, but offers no help as to how to set it. Most papers on DP present results with a range of values for ε, leaving the decision of what value ε should have in practice to the policymaker. This is, after all, one of the core innovations of DP: it leaves the decision of privacy protection to

the policymaker, rather than to technologies. But the decision still needs to be made if data are to be released.

Because noise is added to the result of a query, the designer of the query must assess whether the returned values have enough precision—if they are a sufficiently close approximation to the actual value—that the result is actually useful in answering the question of interest. If ε=0, there is perfect privacy, but essentially infinite noise has been added to the result, so it is useless. As ε increases, the abstract privacy loss increases, as does the utility of the data. In practice, in order to achieve sufficient utility, the values of ε for actual working systems have been far higher than those that were anticipated by DP's inventors or by many members of the DP research community.

To better understand DP in practical terms, we implemented one of the commonly used DP algorithms for adding noise to a query result, the Laplace noise distribution. We plotted the actual noise plots for the simplest typical DP query, which is a *count* query: in a given set of records, how many of them match some selection criterion? We will use this implementation in the DP tutorial that will be part of our final deliverables for this project.

The foundational assumption of DP is that the adversary attempting to undo the protection has access to an arbitrary amount of external data. For a count query, the worst-case assumption is that the adversary knows, for all records in the data but one, whether they match the criterion. The records for which the adversary knows the answer can be ignored in deciding how much noise to add. There is only one record of interest, so removing the other records from the returned value, the answer will either be 0 or 1. DP requires us to add enough noise so that the adversary cannot be sure which the true answer is.

Figure 1 plots the actual distribution of noise that DP would add to the result of the count query, for different values of ε.

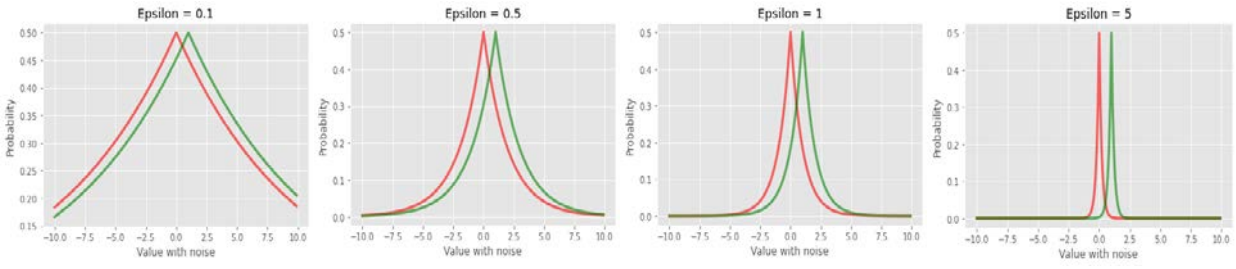

Figure 1: The actual distribution of the noise function, using the Laplace noise function commonly used in DP. For ε = .1, the returned answer might differ from the true value over a range of more than +/- 5. As we increase ε, the overlap between the curves decreases.

For this specific query, we can compute the probability that the returned value is the true value, by looking at areas under the curve. The result are as follows:

ε	Probability of returning the true value
.1	.52 (essentially random)
.5	.61
1	.7
5	.96

If the adversary knows ε (which is typically published with the data), for ε < 1 the odds of guessing correctly are low enough that the adversary would probably not guess. There is substantial privacy protection in these cases. By the time ε is 5 for this single query, the odds of guessing right are high enough that the adversary would probably try to exploit the result, with a resulting loss of privacy.

This example shows how noise can be mapped to a probability of privacy loss that could make sense to a policy-maker, but the challenge is that the value of ε as an intermediate step in deriving that understanding is not helpful. Further, the relationship between ε and some practical measure of potential harm is going to be different for each sort of query.

The plots also allow us to estimate the utility of the different values of ε. The impact of the noise on the result depends on the total number of values in the database. If there were 10 records, the noise for ε=.1, which might well be +/-5, would render the result essentially useless. But if there were 10,000 records in the database, getting an answer that might be off by as much as 5 would still be a small level of uncertainty. An important intuition about DP is that the amount of noise that has to be added is (in a simple version) independent of the number of records in the underlying database, but the utility does depend on that number.

The worst-case assumption about the external knowledge of the adversary will often be unrealistic. In practice, researchers may be able to formulate a more realistic model of what the adversary might know, but the challenge then is that there is no general way to incorporate that model into the mathematical formulations of DP. The actual potential for harm gets even further disconnected from the value of ε.

While this assessment of DP may seem somewhat pessimistic, we find several important insights in the DP approach.

- Adding noise (by whatever means the amount is computed) is a theoretically sound approach to privacy protection.
- Other forms of privacy protection such as k-anonymity can suffer low probability events that cause massive losses of privacy. If noise infusion is incorporated into these other schemes, the harm from these events can be reduced.
- While there are software packages that provide basic DP primitives in the context of a database query system, for best result it may be necessary to design a bespoke DP system for a given application, taking into account the specifics of the data and the nature of the desired query. This reality raises the barrier to adoption. It is necessary to

think of DP both as a software package and as a foundational concept that can underlie many embodiments.

- Combinations of noise, clustering, and aggregation may provide good protection and high utility. That is, it may be necessary to adopt a system that is "inspired" by DP or that has elements of DP, but incorporates traditional privacy protection approaches that lack the mathematical rigor of DP. Such a scheme will almost certainly have to be designed from scratch, which will both increase development costs and will cause concern among DP researchers and advocates, necessitating the need for independent review.
- Any query based on confidential data that is released will cause some loss of privacy and a resultant privacy risk. DP provides a structured way to think about this risk, even if the concept of ε is hard to translate into practical measures of loss, but DP does not create risk-free data for release. There is always risk.

Software for DP.

We have identified three open-source software packages that implement the basic DP primitives: εktelo[1][1], OpenDP (from Harvard)[2] and a package from Google[3]. In the next year, we will explore the utility of these packages for protecting various sorts of network data, and provide a tutorial on their use. In particular, we will try to reproduce some of the queries that are documented in [2], which describes the use of the PINQ DP system to make queries across network traces (see the discussion of Task 1.2.4.1). While the PINQ system has not been supported for some time, the various query strategies described there should be implementable in the software packages we found that are currently supported.

[1] https://ektelo.github.io
[2] https://privacytools.seas.harvard.edu/opendp
[3] https://github.com/google/differential-privacy