Differentially Private Password Frequency Lists

Or, How to release statistics from 70 million passwords (on purpose)

Jeremiah Blocki
Microsoft Research
Email: jblocki@microsoft.com

Anupam Datta
Carnegie Mellon University
Email: danupam@cmu.edu

Joseph Bonneau
Stanford University & EFF
Email: jbonneau@cs.stanford.edu

***Abstract*—Given a dataset of user-chosen passwords, the frequency list reveals the frequency of each unique password. We present a novel mechanism for releasing perturbed password frequency lists with rigorous security, efficiency, and distortion guarantees. Specifically, our mechanism is based on a novel algorithm for sampling that enables an efficient implementation of the exponential mechanism for differential privacy (naive sampling is exponential time). It provides the security guarantee that an adversary will not be able to use this perturbed frequency list to learn anything of significance about any individual user's password even if the adversary already possesses a wealth of background knowledge about the users in the dataset. We prove that our mechanism introduces minimal distortion, thus ensuring that the released frequency list is close to the actual list. Further, we empirically demonstrate, using the now-canonical password dataset leaked from RockYou, that the mechanism works well in practice: as the differential privacy parameter ϵ varies from 8 to 0.002 (smaller ϵ implies higher security), the normalized distortion coefficient (representing the distance between the released and actual password frequency list divided by the number of users N) varies from 8.8×10^{-7} to 1.9×10^{-3}. Given this appealing combination of security and distortion guarantees, our mechanism enables organizations to publish perturbed password frequency lists. This can facilitate new research comparing password security between populations and evaluating password improvement approaches. To this end, we have collaborated with Yahoo! to use our differentially private mechanism to publicly release a corpus of 50 password frequency lists representing approximately 70 million Yahoo! users. This dataset is now the largest password frequency corpus available. Using our perturbed dataset we are able to closely replicate the original published analysis of this data.**

I. Introduction

Passwords are the dominant method of authentication on the internet, and, despite many attempts to replace them [21], [8], [2], they are likely to remain entrenched as the dominant means of authentication [19]. Therefore, it is useful to understand and characterize the underlying distribution of user selected passwords. Given passwords from N users a password frequency list is a multiset (or bag) of non-negative integers $f_1 \geq f_2 \geq \ldots \geq 0$ such that $\sum f_i = N$. Here, f_i denotes the number of users who selected the i^{th} most popular password. For example, if '123456' is the most common password than f_1 would denote the number of users who selected the password '123456.' However, the password frequency list does not include the corresponding passwords.

Password frequency lists from empirical datasets have great value to security researchers who wish to understand the nature of an underlying password distribution so that they can accurately estimate security risks or evaluate various password defenses. For example, the sum $\lambda_\beta = \sum_{i=1}^{\beta} f_i$ is an approximate upper bound on the number of accounts that an untargeted adversary could compromise with β guesses per user. Because a password frequency list allows us to compute λ_β for every value of β this data can be useful for analyzing defenses against online attacks (small β) and offline attacks (larger β). An organization might use this data to help determine how much key-stretching to perform on its authentication servers to mitigate damages in the event of an offline attack. Similarly, consider an organization which implements a k-strikes policy to protect users against online password attackers. A smaller value of k can decrease the usability of the authentication experience, while selecting a larger value of k can reduce security. The empirical data from password frequency list(s) could help organizations to make a more informed decision when considering the trade-off between security and usability.

Despite their usefulness an organization may understandably be wary of publishing password frequency lists for its own users due to potential security and privacy risks. For example, Yahoo! allowed Bonneau [5] to collect anonymized password frequency data from a random sample of 70 million users and publish some aggregate statistics such as min-entropy. However, Yahoo! declined to publish these password frequency lists so that other researchers could use them. In the absence of provable security guarantees this was a reasonable decision. In the past researchers have been able to exploit background knowledge to re-identify individuals in seemingly 'anonymized' datasets [28], and it is not unreasonable to expect that an adversary will have some background knowledge about the password dataset. For example, an adversary would have access to data from previous breaches like RockYou and Adobe.

Consider a toy scenario in which 10 users create passwords at Yahoo!, with 8 selecting the password '123456' and the other 2 selecting the password 'abc123.' In this case the frequency list is $f = (8, 2)$. Now imagine that, after Yahoo! publishes f, the adversary learns the passwords of 9 of these users (e.g., from a server breach at another service like RockYou). In this case the adversary could combine f with his background knowledge to learn the password for the last

NDSS '16, 21-24 February 2016, San Diego, CA, USA
Copyright 2016 Internet Society, ISBN 1-891562-41-X
http://dx.doi.org/10.14722/ndss.2016.23328

user. While somewhat far-fetched, this scenario illustrates the potential risk of releasing password frequency lists when the adversary might have background knowledge.

Contributions: We present a differentially private algorithm for releasing password frequency lists with minimal cumulative distortion. Differential privacy provably provides strong security guarantees to users in the dataset. An adversary will not be able to use this perturbed frequency list to learn anything of significance about any individual user's password. These guarantees will continue to hold even if the adversary obtains background knowledge about the users in the future. Even in the extreme example discussed above the adversary will not be able to extract useful information about the user's password from the distorted password frequency list.

At a technical level our algorithm is based on the exponential mechanism of McSherry and Talwar [27]. The key algorithmic challenge is developing an efficient algorithm to sample from the exponential mechanism for integer partitions — a straightforward implementation would require exponential time and space and there is strong evidence that sampling from the exponential mechanism is computationally intractable in the most general case [36]. We use a novel dynamic programming algorithm to approximately sample from the exponential mechanism for integer partitions. We prove that our that algorithm is efficient and that it introduces minimal L1 distortion to the underlying password frequency list. The algorithm is of independent interest outside of its application to releasing password frequency lists. For example, our algorithm improves upon the differentially private algorithm of Hay et al. [18] for releasing degree distributions in a social network.

We empirically demonstrate, using the RockYou dataset of released passwords, that the mechanism works well in practice: as the differential privacy parameter ϵ varies from 8 to 0.002 (smaller ϵ implies higher security), the normalized distortion coefficient (representing the distance between the released and actual password frequency list divided by the number of users N) varies from 8.8×10^{-7} to 1.9×10^{-3}. This combination of security and distortion guarantees makes it practical for organizations to release perturbed password frequency lists to support public research on password security.

As a demonstration, we have collaborated with Yahoo! to perform perhaps the first *intentional* release of a large corpus of password data using our differential privacy approach. This rich dataset represents approximately 70 million Yahoo! users and contains frequency lists for more than 50 subsets of these users based on demographic and behavioral characteristics. It is our hope that other organizations might follow suite by collecting and publishing their own differentially private password frequency lists.

This represents a promising development both for password research, which is slowly transitioning from analyzing unintentionally leaked data from breached websites to working with legitimate data sources, as well as for differential privacy which is just beginning to gain practical adoption after nearly a decade of research.

II. Background

A. Integer Partitions

A partition of a non-negative integer $n \in \mathbb{N}$ is an ordered list of n integers $x_1 \geq x_2 \geq \ldots \geq x_n \geq 0$ such that

$$\sum_{i=1}^{n} x_i = n \ .$$

We use $\mathcal{P}(n)$ to denote the set of all partitions of the integer n. For example, $\mathcal{P}(3) = \{(1,1,1), (2,1,0), (3,0,0)\}$. Hardy and Ramanujan [17] showed that asymptotically

$$|\mathcal{P}(n)| \sim \frac{1}{4n\sqrt{3}} \exp\left(\pi\sqrt{\frac{2n}{3}}\right) \ ,$$

and Pribitkin [10] proved a very similar upper bound on $|\mathcal{P}(n)|$ for all $n \geq 1$. We let $\mathcal{P} \doteq \bigcup_{n=0}^{\infty} \mathcal{P}(n)$ denote the set of all integer partitions.

Given two partitions $x \in \mathcal{P}(n)$ and $x' \in \mathcal{P}(n')$ we use

$$\mathbf{dist}(x, x') \doteq \frac{1}{2} \sum_{i=1}^{\max\{n,n'\}} |x_i - x'_i| \ ,$$

to measure the distance between the two partitions — for partitions $x \in \mathcal{P}(n)$ we adopt the convention that $x_i \doteq 0$ whenever $i > n$. **dist** is essentially the L1 norm $\|x - x'\|_1$, also called the absolute value norm, of the vector $x - x'$. For convenience, we choose to normalize by $1/2$ because it leads to a natural interpretation of **dist** in our passwords context — if $f \in \mathcal{P}(n)$ represents the initial password frequency list and $f' \in \mathcal{P}(n)$ represents the frequency counts after a single user changes his password then $\mathbf{dist}(f, f') \leq 1$.

a) Measuring Utility: Given an input partition $x \in \mathcal{P}$ and an output partition $x' \in \mathcal{P}$ we use $\mathbf{dist}(x, x')$ to measure the quality of the output x'. We stress that our goal of minimizing cumulative error should not be confused with related goals like ensuring that the worst case error on a single entry (e.g., $\max_i |x_i - x'_i|$) is small or ensuring that the L2 norm $\|x - x'\|_2$ is small. These goals are strictly weaker than our goal. For example, if we set $x'_i = x_i + 1$ for $i \leq n$ then the worst case error would be 1 and the L2 norm would be $\sqrt{n}$, but we would have $\mathbf{dist}(x, x') = n/2$.

b) Password Applications: Given a large dataset D of passwords (or password hashes) we let f_1^D denote the number of users who selected the most popular password pwd_1^D (typically, pwd_1^D will be something like '123456' [20]). In general, for $i \geq 1$ we let f_i^D denote the number of users who selected the i^{th} most popular password. Formally,

$$f_i^D = \left|\left\{u \in \mathcal{U} : pwd_u = pwd_i^D\right\}\right| \ ,$$

where $\mathcal{U}$ denotes the set of users and pwd_u denotes the password of user $u \in \mathcal{U}$ and pwd_i^D denotes the i^{th} most popular password in the dataset. We will use $N_D \in \mathbb{N}$ to denote the number of passwords in a dataset D. Because the dataset D is usually clear from context we will typically drop the D and write pwd_i and f_i and N.

If there are N users in a password dataset D then the password frequency list $f = f_1, f_2, \ldots f_N$ is simply a partition of the integer N — when there are only $t < N$ unique passwords in the dataset we define $f_i = 0$ for $i > t$.

It is useful for password researchers and system administrators to know the values of $f = f_1, \ldots, f_N$. For example,

$$\lambda_\beta \doteq \sum_{i=1}^{\beta} f_i$$

is an upper bound on the number of accounts that an (untargeted) adversary could compromise with a budget of β guesses per user. If we let $\tilde{f} = \tilde{f}_1, \ldots, \tilde{f}_{N'}$ denote a perturbed password frequency list then the quantity $2 \cdot \mathbf{dist}\left(f, \tilde{f}\right)$ is an upper bound on the error term $\left|\lambda_\beta - \tilde{\lambda}_\beta\right|$ for *any* adversary budget β. By contrast, we note that a bound on the L2 norm (or a bound on the worst case error) does not yield a meaningful bound on this error term. For example, if $\tilde{f}_i = f_i + 1$ for all $i \leq N$ then the L2 norm is $\|f - \tilde{f}\|_2 = \sqrt{N}$, but the error term is $\left|\lambda_\beta - \tilde{\lambda}_\beta\right| = \beta$ so the estimate $\tilde{\lambda}_\beta$ is increasingly less accurate as β grows.

Password frequency lists (or accurate approximations) can guide the development of password policies and defenses. For example, suppose that an authentication server changes its k-strikes policy from $k = 3$ to the more lenient $k = 5$. A password frequency list would help us predict the security implications of this policy change — an adversary could compromise $\lambda_5 - \lambda_3$ more accounts in an offline attack. Similarly, the data from a password frequency list would allow us to estimate how many password guesses an offline attacker would be willing to try in order to crack a password hash. A rational adversary should stop guessing when his marginal guessing costs exceed his marginal utility, and we can use a password frequency list to predict when this will happen. In particular, if c_g (resp. v) denotes the cost of one password guess (resp. value of a cracked password) then an offline adversary's marginal guessing cost (resp. marginal benefit) is $c_g\left(1 - \lambda_\beta/N\right)$ (resp. $v \cdot f_{\beta+1}/N = v\big(\lambda_{\beta+1} - \lambda_\beta\big)/N)$ if he stops after $\beta + 1$ guesses instead of β guesses[1]. An authentication server might alter the adversary's marginal guessing costs by make its password hash function more expensive to compute (e.g., by adopting key stretching techniques like hash iteration [31] or memory hard functions [29]). However, this means that the server incurs greater costs every time a user authenticates. If we know $f_1, \ldots, f_n$ then we could precisely quantify the security gains from key stretching. As a final application suppose that $f, g \in \mathcal{P}$ encode the frequencies of passwords selected under two different password composition policies (e.g., must include numbers, capital letters etc...). If we had f and g then we could compare the security impact of these policies. Similarly, f and g might represent frequency lists for passwords chosen by two different populations of users, e.g. users from the U.S. and users from the U.K., allowing researchers to analyze factors impacting user's password choices.

B. Differential Privacy

We use the notion of differential privacy [12], [27] to ensure that the password statistics we release will not harm individual users. Differential privacy provides a strong information theoretic privacy guarantees to each individual in a dataset, and it has been an active area of research in the last decade (e.g., see [13]). Informally, a differentially private mechanism for releasing statistics about a dataset D ensures that an adversary cannot use the output to make inferences about any individual in the dataset except inferences that the adversary would have been able to make without that individual's data.

We first introduce the notion of neighboring partitions — see Definition 1. Intuitively, two password datasets D and D' are neighbors if we can obtain D' by adding (or removing) a single user to the dataset D. In this case we will have $\mathbf{dist}\left(f, f'\right) = \frac{1}{2}$ because we will have $f_i = f'_i$ for every index except one, and $f'_j = f_j + 1$ for one index (e.g., when we add a user with password pwd_j).

Definition 1. *(Neighboring Partitions) We say that two partitions $f, f' \in \mathcal{P}$ are neighbors if $\mathbf{dist}\left(f, f'\right) = \frac{1}{2}$, and we write $f \sim f'$ to indicate this.*

Observe that if two password datasets D and D' are neighbors then the corresponding frequency lists f and f' will also be neighbors. We do not require that the converse holds.

We are now ready to define differential privacy. To simplify our presentation we define differential privacy for mechanisms $\mathcal{A} : \mathcal{P} \rightarrow \mathcal{P}$ which output integer permutations — of course the general definition of differential privacy is applicable in other contexts.

Definition 2. *[12] A randomized mechanism $\mathcal{A} : \mathcal{P} \rightarrow \mathcal{P}$ preserves (ϵ, δ)-differential privacy if for* any *pair $f \sim f' \in \mathcal{P}$ and* any *subset of outcomes $S \subseteq \mathcal{P}$ we have*

$$\Pr\left[\mathcal{A}\left(f\right) \in S\right] \leq e^\epsilon \Pr\left[\mathcal{A}\left(f'\right) \in S\right] + \delta \ .$$

When $\delta = 0$ we say that $\mathcal{A}$ is ϵ-differentially private.

Smaller values of ϵ and δ yield stronger privacy guarantees. We will think of ϵ as being a moderately small constant (typically $\epsilon \leq 1$) and we will think of δ as being a negligibly small value (e.g., $\delta = 2^{-100}$). Note that we will allow our mechanism to output $\tilde{f} \in \mathcal{P}\left(\tilde{N}\right)$ given an input $f \in \mathcal{P}\left(N\right)$ even if $N \neq \tilde{N}$. We discuss the security implications of this definition below.

C. Security Implications

Differential privacy offers very strong security and privacy guarantees. A nice feature of differential privacy is that it is preserved under post-processing [13]. We will use the following fact in our discussion of security.

Fact 1. *Suppose that the randomized mechanism $\mathcal{A} : \mathcal{P} \rightarrow \mathcal{P}$ preserves (ϵ, δ)-differential privacy and let $g : \mathcal{P} \rightarrow Range\left(g\right)$ be an arbitrary post-processing function. Then the mechanism $g \circ \mathcal{A}$ also preserves (ϵ, δ)-differential privacy.*

Intuitively, we can take g_u to denote *any* attack that the adversary might mount against a user u after we publish our

[1] An offline adversary with a maximum budget of $\beta + 1$ guesses will only use $\big(\beta + 1\big)\big(1 - \lambda_\beta/N\big) + \sum_{i=1}^{\beta} i \cdot f_i/N$ guesses on average. Thus, the marginal guessing cost is $c_g\big(\beta + 1\big)\big(1 - \lambda_\beta/N\big) + c_g \sum_{i=1}^{\beta} i \cdot f_i/N - c_g\left(\beta\big(1 - \lambda_{\beta-1}/N\big) + \sum_{i=1}^{\beta-1} i \cdot f_i/N\right) = c_g\big(1 - \lambda_\beta/N\big) + c_g\beta\big(\lambda_{\beta-1}/N - \lambda_\beta/N\big) + c_g\beta f_\beta/N = c_g\big(1 - \lambda_\beta/N\big)$.

Term	Explanation
N	the number of user passwords in a dataset
$\mathcal{P}(N)$	the set of all integer partitions of N
$\mathcal{P}$	the set of all integer partitions
pwd_u	the password of user u
pwd_i	the i^{th} most likely password in D
f_i	$\#$ users u that selected $pwd_u = pwd_i$
$f = f_1 \geq f_2 \geq f_2 \ldots$	Original password frequency list. Also an integer partition $f \in \mathcal{P}(N)$
$\tilde{f}$	a perturbed password frequency list

TABLE I: Notation

password frequency statistics. Notably this also includes any attack that an adversary with significant background knowledge might mount. Let $S \subseteq Range(g_u)$ denote the set of outcomes that the user u would consider harmful and let f^{-u} denote the password frequency list after deleting u's data. Because $g_u \circ \mathcal{A}$ preserves (ϵ, δ)-differential privacy we have

$$\Pr\left[g_u\left(\mathcal{A}(f)\right) \in S\right] \leq e^{\epsilon} \Pr\left[g_u\left(\mathcal{A}\left(f^{-u}\right)\right) \in S\right] + \delta \ .$$

Intuitively, this implies adversary's attack g_u must have a small probability of harming our user u given the perturbed password frequency list $\mathcal{A}(f)$ unless the attack g_u already had a reasonable chance of harming u when we omit his data entirely! Thus, a differentially private mechanism ensures that the password frequency list we release will not be useful to an adversary who is trying to guess a particular user u's password. We stress that g_u could describe attacks from any current or future adversary. Even if the adversary learns the passwords for the first $n-1$ users the security guarantees still apply for the last user. We remark that differential privacy does not ensure that the adversary won't be able guess the user's password — if the user selected a weak password then he was at risk before we publish the perturbed frequency lists. However, we can guarantee that the perturbed password frequency list will never help the adversary guess the user's password.

Group Privacy: A mechanism that preserves (ϵ, δ)-differential privacy for individual users will also preserve $(2\epsilon, \delta \cdot (1 + e^{\epsilon}))$-differential privacy for groups of size two. Group differential privacy would imply that the adversary cannot distinguish between the original dataset and the dataset in which any two users deleted their records. Thus, we can still provide strong security guarantees for a user creates two duplicate accounts with related passwords — albeit with slightly weaker security parameters. If a user creates more an more duplicate accounts then the security guarantees would continue to degrade in this manner.

D. Goal

Our goal is to develop a differentially private mechanism that is both efficient and accurate.

Efficiency. In our case the range of outcomes $\mathcal{P}$ consists of infinitely many partitions. Furthermore, the input size N can be quite large.[2] We will not be content to simply find a mechanism that runs in polynomial time in $N, 1/\epsilon$ and $\log(1/\delta)$. Our goal is to develop a mechanism that we can actually implement and run. Because N is typically large in our applications this implies our algorithm must run in sub-quadratic time in N.

Accuracy. Given an input partition $x \in \mathcal{P}(N)$ we want to ensure that our output partition $y \in \mathcal{P}$ minimizes cumulative error (e.g. $\mathbf{dist}(x, y)$). In private data analysis it is desirable to achieve normalized error $O\left(N^{-1/2}\right)$ so that the error we introduce to preserve privacy does not exceed our statistical sampling error which typically scales as $O\left(N^{-1/2}\right)$. Thus, our goal will be to ensure that with high probability we have normalized error $\frac{\mathbf{dist}(x,y)}{N} \leq \frac{c}{\sqrt{N}}$ for some small constant c.

Standard Techniques. Our goal of minimizing cumulative error is much stronger than weaker goals like ensuring that the worst case error on a single entry (e.g., $\max_i |x_i - x'_i|$) is small or ensuring that the L2 norm $\|x - x'\|_2$ is small. These weaker goals could be achieved with standard techniques for preserving differential privacy like the Laplacian mechanism or or the Gaussian mechanism [13]. However, we stress that standard techniques like these do not guarantee small cumulative error rates. These mechanisms add noise e_i to each component (e.g., $\tilde{x}_i = x_i + e_i$) where the noise e_i is drawn from a Laplacian distribution of from a Gaussian distribution with the appropriate parameters (afterwards we can sort the $\tilde{x}_i$ values to ensure that $\tilde{x}_i \geq \tilde{x}_{i+1}$). Theoretically, the cumulative error can be as large as $\mathbf{dist}(x, y) = \Omega\left(N \times e^{-\epsilon}\right)$. For example, if we set $x'_i = x_i \pm 1$ for $i \leq N$ then the worst case error would be 1 and the L2 norm would be $\sqrt{N}$, but we would have $\mathbf{dist}(x, x') = N/2$. We tested these standard mechanisms on the RockYou password dataset and found that the empirical error rate was just as high in practice.

III. Analysis of Exponential Mechanism

Our mechanism for releasing password frequency lists is based on the exponential mechanism of McSherry and Talwar [27]. In our context the definition of the exponential mechanism is relatively simple. Given an input dataset $f \in \mathcal{P}$ the exponential mechanism $\mathcal{E}^{\epsilon}(f)$ simply outputs each possible outcome $\tilde{f} \in \mathcal{P}$ with probability proportional to $\exp\left(-\epsilon \cdot \mathbf{dist}\left(f, \tilde{f}\right)\right)$. Intuitively, a partition $\tilde{f} \in \mathcal{P}$ is less likely to be output whenever $\mathbf{dist}\left(f, \tilde{f}\right)$ is large. Theorem 1, due to McSherry and Talwar [27], states that the exponential mechanism preserves differential privacy.

Formally, when we sample from $\mathcal{E}^{\epsilon}(f)$ the probability of each outcome $\tilde{f}$ is $w_f \cdot \exp\left(-\epsilon \cdot \mathbf{dist}\left(f, \tilde{f}\right)\right)$, where $w_f \in \mathbb{R}$ is the unique constant such that

$$w_f \doteq 1 \Bigg/ \left(\sum_{\tilde{f} \in \mathcal{P}} \exp\left(-\epsilon \cdot \mathbf{dist}\left(f, \tilde{f}\right)\right)\right) \ .$$

a) Exponential Mechanism is Well Defined: Because our outcome space $\mathcal{P}$ is infinite we need to argue that this distribution is well defined (i.e. that the constant exists and $w_f > 0$). The proof of Fact 2 is in the appendix. We note that Fact 2 is a much weaker claim than Theorem 2 below.

Fact 2. *The sum* $\sum_{\tilde{f} \in \mathcal{P}} \exp\left(-\epsilon \cdot \mathbf{dist}\left(f, \tilde{f}\right)\right)$ *converges.*

[2] Indeed, Bonneau demonstrated that most interesting strength metrics for a password frequency list require N to be in the millions to be accurately approximated [5].

b) The Exponential Mechanism preserves Differential Privacy: The next result states that the exponential mechanism does preserve differential privacy. The proof relies on the fact that $\mathbf{dist}\left(f, f'\right) = \frac{1}{2}$ for neighboring partitions $f \sim f'$.

Theorem 1. *[27] The exponential mechanism $\mathcal{E}^{\epsilon}$ preserves ϵ-differential privacy.*

Theorem 1 is a restatement of the Theorem of McSherry and Talwar [27] once we observe that the global sensitivity of the function $\mathbf{dist}$ is $\Delta \doteq \max_{f \sim f'} \mathbf{dist}\left(f, f'\right) = \frac{1}{2}$. The proof is quite simple. We include it in the appendix for completeness.

c) The Exponential Mechanism Introduces Minimal Cummulative Distortion: Theorem 2, our main technical result in this section, shows that the exponential mechanism would introduce minimal cumulative distortion $\mathbf{dist}\left(f, \tilde{f}\right) = O\left(\sqrt{N}/\epsilon\right)$ if we had an efficient algorithm to sample from this distribution.

Theorem 2. *Let $f \in \mathcal{P}\left(N\right)$, $\epsilon > \frac{48\pi^2}{\sqrt{N}}$ and $\delta \geq e^{1-\sqrt{N}/2}$ be given and let $\tilde{f} \leftarrow \mathcal{E}^{\epsilon}\left(f\right)$ denote a random sample from the exponential mechanism. Then except with probability δ we will have*

$$\mathbf{dist}\left(f, \tilde{f}\right) \leq \frac{c_1\sqrt{N} + c_2 \ln\left(\frac{1}{\delta}\right)}{\epsilon} ,$$

where $c_1 = 2\pi\sqrt{\frac{2}{3}} + o(1)$ and $c_2 = 2 + o(1)$ are constants.

We note that in all of our experiments we will have $\epsilon > \frac{48\pi^2}{\sqrt{N}}$ and $\delta \geq e^{1-\sqrt{N}/2}$ as $N \geq 10^7$. It is possible obtain similar results for other ranges of ϵ and δ, though the constants c_1 and c_2 in Theorem 2 would change.

The proof of Theorem 2 is in the appendix. Given $f \in \mathcal{P}\left(N\right)$ we prove that $2\left|\mathcal{P}\left(N+d\right)\right|$ is an upper bound on the number of partitions $f' \in \mathcal{P}$ with $\mathbf{dist}\left(f, f'\right) = d/2$ — see Lemma 4 in the appendix. Applying union bounds the cumulative probability weight of all such partitions f' is at most $2\left|\mathcal{P}\left(N+d\right)\right| w_f \exp\left(-\epsilon \cdot d/2\right)$. Pribitkin [10] proved that $\left|\mathcal{P}\left(N\right)\right| \leq N^{-3/4}\exp\left(\pi\sqrt{\frac{2N}{3}}\right)$ so the cumulative probability weight of all partitions in f' with $\mathbf{dist}(f, f') = d/2$ will drop exponentially as d increases. We remark that Theorem 2 is simply an upper bound on the cumulative error. It is reasonable to expect that the empirical error will be even lower in practice. Indeed, this is what we find in Section V.

IV. Differentially Private Mechanism

In the last section we saw that the exponential mechanism preserves differential privacy and introduces minimal cumulative distortion. However, we still need an efficient algorithm to sample from the exponential mechanism when N is large (e.g., $N \approx 7 \times 10^7$). A naive sampling algorithm would take exponential time, and there is strong evidence that there is no polynomial time algorithm to sample from the exponential mechanism in general— see discussion in the Related Work section. However, these results do not imply that it is hard to sample from the exponential mechanism over integer partitions. In this section we introduce a novel relaxation of the exponential mechanism that preserves (ϵ, δ)-differential privacy for a negligible value of δ, and we demonstrate that there is an efficient $O\left(N^{1.5}/\epsilon\right)$-time algorithm to sample from this relaxed exponential mechanism.

A. A Relaxation of the Exponential Mechanism

Our first step in developing an efficient sampling algorithm is to propose a novel relaxation of the exponential mechanism. The standard exponential mechanism places a probability weight on each partition $\tilde{f} \in \mathcal{P}$, but there are infinitely many partitions. In our relaxation we allow the mechanism to completely ignore partitions $\tilde{f} \in \mathcal{P}$ that are 'very far' from the true distribution (e.g., $\mathbf{dist}\left(f, \tilde{f}\right) \geq \Omega\left(\frac{\sqrt{N}+\ln(1/\delta)}{\epsilon}\right)$). Our relaxed mechanism will preserve (ϵ, δ)-differential privacy for a negligibly small value of δ. Like the standard exponential mechanism the relaxed exponential mechanism will introduce minimal cumulative distortion.

Let $\mathcal{S} = \left\{S_f \subseteq \mathcal{P}\right\}$ denote a family of subsets of integer partitions — one for each partition $f \in \mathcal{P}$. Given a set family we define the $\mathcal{S}$-restricted exponential mechanism. The probability of any outcome $\tilde{f} \notin S_f$ under our relaxed mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}}\left(f\right)$ is the probability of the same outcome under the original exponential mechanism $\mathcal{E}^{\epsilon}\left(f\right)$ conditioned on the event that the outcome is not in S_f. Intuitively, we could think of the restricted exponential mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}}\left(f\right)$ as repeatedly running the basic exponential mechanism until we get an output value that is not in our restricted set S_f. Formally, we have

$$\Pr_{y \leftarrow \mathcal{E}^{\epsilon}_{\mathcal{S}}(f)}\left[y \in S_f\right] = 0 ,$$

and for every $x \notin S_f$

$$\Pr_{y \leftarrow \mathcal{E}^{\epsilon}_{\mathcal{S}}(f)}\left[y = x\right] = \Pr_{y \leftarrow \mathcal{E}^{\epsilon}(f)}\left[y = x \middle| y \notin S_f\right] .$$

We can show that the restricted exponential mechanism preserves (ϵ, δ)-differential privacy as long as each of the restricted sets S_f in our set family has negligibly small probability mass — see Definition 3.

Definition 3. *We say that a set family $\mathcal{S} = \left\{S_f \subseteq \mathcal{P}\right\}$ is δ-negligible if for every partition $f \in \mathcal{P}$ we have*

$$\Pr_{y \leftarrow \mathcal{E}^{\epsilon}(f)}\left[y \in S_f\right] \leq \delta .$$

Theorem 3 states that our modified exponential mechanism preserves (ϵ, δ')-differential privacy for a negligibly small value δ' as long as the underlying set family $\mathcal{S}$ is δ-negligible. The intuition behind the proof of Theorem 3 is that the statistical distance between the distribution induced by the standard exponential mechanism $\mathcal{E}^{\epsilon}\left(f\right)$ and the distribution induced by our modified version $\mathcal{E}^{\epsilon}_{\mathcal{S}}\left(f\right)$ is negligibly small. The standard exponential mechanism preserves pure ϵ-differential privacy (Theorem 1) and the δ' term accounts for the negligibly small statistical distance between the two distributions.

Theorem 3. *For any δ-negligible set family $\mathcal{S}$ the mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}}$ preserves (ϵ, δ')-differential privacy for $\delta' = \delta + e^{\epsilon}\delta$.*

Corollary 1 extends Theorem 1 to bound the error of our modified exponential mechanism. Corollary 1 follows easily from Theorem 1 because the statistical distance between the modified exponential mechanism and the regular exponential mechanism is negligibly close.

Corollary 1. *Let $f \in \mathcal{P}(N)$, $\epsilon > \frac{48\pi^2}{\sqrt{N}}$ and $\delta \geq e^{1-\sqrt{N}/2}$ be given and let $\tilde{f} \leftarrow \mathcal{E}^\epsilon(f)$ denote a random sample from the modified exponential mechanism with a δ_1-negligible set family $\mathcal{S} = \{S_f \subseteq \mathcal{P}\}$. Then except with probability $\delta_1 + \delta_2$ we will have*

$$\mathbf{dist}\left(f, \tilde{f}\right) \leq \frac{c_1\sqrt{N} + c_2 \ln\left(\frac{1}{\delta_2}\right)}{\epsilon} ,$$

where $c_1 = 2\pi\sqrt{\frac{2}{3}} + o(1)$ and $c_2 = 2 + o(1)$ are constants.

In fact for our particular restricted set family the above claim will hold with probability δ_2 because the set S_f of restricted partitions will only contain partitions that are far away from f.

B. Restricted Set Families

In this section we introduce the restricted set family used by our relaxed exponential mechanism. Intuitively, we define a set S_f^d which contains only partitions that are distance $\geq d$ from f. Formally, given a fixed integer $d \in \mathbb{N}$ and a fixed partition $f \in \mathcal{P}$ we define

$$U_j \doteq \max_{\substack{x \in \mathcal{P} \\ \mathbf{dist}(f,x) \leq d}} x_j \quad \text{and} \quad L_j \doteq \min_{\substack{x \in \mathcal{P} \\ \mathbf{dist}(f,x) \leq d}} x_j \ . \qquad (1)$$

Intuitively, U_j (resp. L_j) is an upper (resp. lower) bound on the value $\tilde{f}_j$ for *any* partition $\tilde{f}$ with distance $\mathbf{dist}\left(f, \tilde{f}\right) \leq d$. A partition $\tilde{f}$ is restricted if and only if we can find an index i such that $\tilde{f}_i$ is outside of the range $[L_i, U_i]$. Formally,

$$S_f^d \doteq \{x \in \mathcal{P} \mid \exists i \geq 1 . x_i \notin [L_i, U_i]\} \ .$$

Thus, for any partition $\tilde{f} \in \mathcal{P}$ with $\mathbf{dist}\left(f, \tilde{f}\right) \leq d$ we have $\tilde{f} \notin S_f^d$. This follows immediately from the definition of U_i and L_i. However, the converse is not necessarily true (i.e. there may be partitions $\tilde{f} \notin S_f^d$ with $\mathbf{dist}\left(f, \tilde{f}\right) \geq d$).

We first observe, Lemma 1, that we can ensure that our set family $\mathcal{S}$ is δ-negligible by choosing $d = \Omega\left(\epsilon^{-1}\sqrt{N} + \epsilon^{-1}\ln\delta^{-1}\right)$. Thus, by Theorem 3 it suffices to show that we can run the modified exponential mechanism with the set family $\mathcal{S} = \left\{S_f^d \,\middle|\, f \in \mathcal{P}\right\}$.

Lemma 1. *Given $f \in \mathcal{P}(N)$ let $S_f^d = \{x \in \mathcal{P} \mid \exists i . f_i \notin [L_i, U_i]\}$ with the U_i's and L_i's defined in equation 1. Let $\epsilon > \frac{48\pi^2}{\sqrt{N}}$ and $\delta \geq e^{1-\sqrt{N}/2}$ and let $d_f = \frac{c_1 \cdot \sqrt{N} - c_2 \cdot \ln\delta}{\epsilon}$, where the constants $c_1 = 2\pi\sqrt{2/3} + o(1)$ and $c_2 = 2 + o(1)$ are the same as in Theorem 2. Then the set family $\mathcal{S} = \left\{S_f^{d_f} \,\middle|\, f \in \mathcal{P}\right\}$ is δ-negligible.*

Proof: We already observed that $S_f^d \subseteq \{x \in \mathcal{P} \mid \mathbf{dist}(x, f) \geq d\}$. Now the proof of Lemma 1 follows immediately from Theorem 2, where we show that

$$\Pr_{y \leftarrow \mathcal{E}^\epsilon(f)}\left[\mathbf{dist}(x, f) \geq d_f\right] \leq \delta \ .$$

$\blacksquare$

C. Dynamic Programming

Our implementation of the modified exponential mechanism $\mathcal{E}_{\mathcal{S}}^\epsilon(f)$ uses dynamic programming. To gain some intuition for why dynamic programming is an appropriate technique we first observe that for any partition $\tilde{f} \in \mathcal{P}$ the weight of $\tilde{f}$ decomposes nicely. In particular, for any index t we have

$$e^{-\sum_i\left|\tilde{f}_i - f_i\right| \cdot \epsilon/2} = \left(e^{-\sum_{i<t}\left|\tilde{f}_i - f_i\right| \cdot \epsilon/2}\right)\left(e^{-\sum_{i \geq t}\left|\tilde{f}_i - f_i\right| \cdot \epsilon/2}\right) .$$

This structure suggests that we might be able develop a recursive formula to express the cumulative weight of all partitions that are consistent with the first $t-1$ values $\tilde{f}_1, \ldots, \tilde{f}_{t-1}$.

Given a partition $\tilde{f} = \tilde{f}_1, \tilde{f}_2, \ldots$ we use $\tilde{f}^t = \tilde{f}_t, \tilde{f}_{t+1}, \ldots$ to denote the partition formed by dropping the first $t-1$ values of $\tilde{f}$. Suppose that at some point during the sampling procedure we have selected values for $\tilde{f}_1, \ldots, \tilde{f}_{t-1}$ we can view $\tilde{f}^t$ as a potential completion of our sampling procedure. In general, a completion $x \in \mathcal{P}$ of $\tilde{f}_1, \ldots, \tilde{f}_{t-1}$ is valid if and only if $x_1 \leq \tilde{f}_{t-1}$ and $\tilde{x}_j \in [L_{j+t}, U_{j+t}]$ for all $j \geq 1$ (so that $\tilde{f} \notin S_f^d$). We use

$$S_{f^t}^d \doteq \{x \in \mathcal{P} \mid \ \exists j \geq 1 . x_j \notin [L_{t+j}, U_{t+j}]\}$$

to denote the set of completions that are excluded by the set S_f^d.

We are now ready to introduce a formula for computing weights in our sampling procedure. We set

$$w_{i,U,\epsilon} \doteq \sum_{n \geq 0} \sum_{\substack{x \in \mathcal{P}(n) \setminus S_{f^i}^d \\ x_1 = U}} \exp\left(-\epsilon \cdot \mathbf{dist}\left(f^i, x\right)\right) \ .$$

Intuitively, given *any* $\tilde{f}_1 \geq \ldots \geq \tilde{f}_{i-1} \geq U$, $w_{i,U,\epsilon}$ represents the cumulative weight of all valid candidate completions $\tilde{f}^i$ with first component $x_1 = U$. Thus, the weights $w_{i,U,\epsilon}$ can be used to sample the next value $\tilde{f}_i$. In particular, for any $U \leq T$ we have

$$\Pr_{\tilde{f} \leftarrow \mathcal{E}_{\mathcal{S}}^\epsilon(f)}\left[\tilde{f}_i = U \,\middle|\, \tilde{f}_{i-1} = T\right] = \frac{w_{i,U,\epsilon}}{W_{i,T,\epsilon}} \ , \qquad (2)$$

where

$$W_{i,T,\epsilon} = \sum_{U' = L_i}^{\min\{U_i, T\}} w_{i,U',\epsilon} \ .$$

We first note that if we had the values $w_{i,U,\epsilon}$ for each pair (i, U) such that (1) $i \leq N + d$ and (2) $L_i \leq U \leq U_i$ then we could easily sample from the modified exponential mechanism by repeatedly invoking equation 2. Once we have sampled $\tilde{f}_1, \ldots, \tilde{f}_{i-1}$ then we can simply apply equation 2 to obtain the next value $\tilde{f}_i$. [3]

It remains to show that we can compute the values $w_{i,U,\epsilon}$ efficiently. We first note that there are at most $2dN + 2d^2 =$

[3] We will pretend that there is a dummy value $\tilde{f}_0 \doteq f_1 + 2d$ to handle the edge case when we sample the first value $\tilde{f}_1$.

$O\left(\frac{N^{1.5}-\ln\delta}{\epsilon}\right)$ pairs (i,U) that satisfy both of our constraints. Lemmas 2 and 3 demonstrate that there is a simple recursive formula for expressing $w_{i,U,\epsilon}$ using either the values $w_{i+1,U',\epsilon}$ or the values $w_{i,U,\epsilon}$.

Lemma 2. *For all (i,U) such that $i \le N+d$ and $L_i \le U \le U_i$ we have*

$$w_{i,U,\epsilon} = \exp\left(\frac{-\epsilon\left|U-f_i\right|}{2}\right)\cdot\sum_{U'=L_{i+1}}^{\min\{U,U_{i+1}\}} w_{i+1,U',\epsilon} \quad .$$

The proof of Lemma 2 is in the appendix. We could fill in our dynamic programming table using only the recursive formula from Lemma 2, but this would require $O(d)$ work per entry. Our next recursive formula, Lemma 3, allows us to fill in each cell in our dynamic programming table in (amortized) constant time per cell. Lemma 3 only allows us to compute $w_{i,U,\epsilon}$ if $U > L_i$; Otherwise, if $U = L_i$ we need to use Lemma 2. Fortunately, we only need to do this $O(N)$ times so our total work is still $O(Nd)$. Lemma 3 follows easily from Lemma 2.

Lemma 3. *For all (i,U) such that $i \le N+d$ and $L_i < U \le U_i$ we have $w_{i,U,\epsilon} =$*

$$e^{-\epsilon\cdot\Delta(U,i)/2} w_{i,U-1,\epsilon} + \left(e^{-\epsilon\cdot|U-f_i|/2}\right)\sum_{U'=U}^{\min\{U,U_{i+1}\}} w_{i+1,U',\epsilon} \ ,$$

where

$$\Delta(U,i) = \begin{cases} 1, & \textit{if } U \ge f_i + 1 \\ -1, & \textit{otherwise.} \end{cases}$$

Proof: We rely on Lemma 2. We have

$$w_{i,U,\epsilon} = e^{-\epsilon\cdot|U-f_i|/2} \sum_{U'=L_{i+1}}^{\min\{U_{i+1},U\}} w_{i+1,U',\epsilon} \ ,$$

and

$$w_{i,U-1,\epsilon} = e^{-\epsilon\cdot|U-1-f_i|/2} \sum_{U'=L_{i+1}}^{\min\{U_{i+1},U-1\}} w_{i+1,U',\epsilon} \ .$$

Thus,

$$\begin{aligned} w_{i,U,\epsilon} &= \left(\frac{e^{-\epsilon\cdot|U-f_i|/2}}{e^{-\epsilon\cdot|U-1-f_i|/2}}\right) w_{i,U-1,\epsilon} \\ &+ \ e^{-\epsilon\times|U-f_i|/2} \sum_{U'=U}^{\min\{U,U_{i+1}\}} w_{i+1,U',\epsilon} \ . \end{aligned}$$

where

$$\frac{e^{-\epsilon\cdot|U-f_i|/2}}{e^{-\epsilon\cdot|U-1-f_i|/2}} = e^{-\epsilon\cdot\Delta(U,i)/2} \ .$$

■

Algorithm 1 Exponential Mechanism: Publish Frequency Vector

```
Sample
Input: $f \in \mathcal{P}(N)$, $\epsilon > 0$, $\delta > 0$.
$d \leftarrow \frac{c_1\cdot\sqrt{N}-c_2\cdot\ln\delta}{\epsilon}$        ▷ $c_1 = 2\pi\sqrt{2/3}+o(1)$ and
                                          ▷ $c_2 = 2+o(1)$ from Thm 2
for $i = 1,\ldots,N+d$ do
    $U_i \leftarrow$ GetUpperBound$(i,d,f)$
    $L_i \leftarrow$ GetLowerBound$(i,d,f)$
$\tilde{f}_0 \leftarrow f_1 + d$                   ▷ Dummy Value for Edge Case

for $i = 1,\ldots,N+d$ do
    $U_{max} \leftarrow \min\{\tilde{f}_{i-1}, U_i\}$
                              ▷ $\tilde{f}_i$ cannot be bigger than $\tilde{f}_{i-1}$ or $U_i$
    $W \leftarrow 0$
    for $U' = L_i,\ldots,U_{max}$ do
        $W_{U'} \leftarrow$ CompWeights$(U', i+1, d, f, \epsilon)$
        $W \leftarrow W + W_{U'}$
    $r \leftarrow$ Unif $(0,1)$
    for $U' = L_i,\ldots,U_{max}$ do
        $r \leftarrow r - \frac{W_{U'}}{W}$

        if $r \le 0$ then
            $\tilde{f}_i \leftarrow U'$
            break
$\tilde{f} \leftarrow \left(\tilde{f}_1,\ldots,\tilde{f}_{N+d}\right)$ return $\tilde{f}$
```

a) Dynamic Programming Algorithm: Our efficient sampling algorithm consists of two phases: preprocessing and sampling. During the preprocessing phase, Algorithm 2, we compute all of the weights $w_{i,U,\epsilon}$. Algorithm 2 relies on the helper functions **GetUpperBound** and **GetLowerBound**, which return the values U_i and L_i respectively. These helper functions are described in Algorithm 3. During the second phase, Algorithm 1, we repeatedly apply equation 2 to sample from the restricted exponential mechanism. A simple amortized analysis of algorithm 3 shows that we can compute each upper bound U_i and each lower bound L_i in time $O(U_i)$ and $O(L_i)$ respectively. Thus, the total time spent computing upper/lower bounds is at most $\sum_{i=1}^{N+d} U_i + L_i = O(Nd)$.

We now show that our sampling algorithm is efficient and correct.

Theorem 4. *Algorithm 1 runs in time and space $O\left(\frac{N\sqrt{N}}{\epsilon} + N\ln\delta^{-1}\right)$ and correctly samples from the relaxed exponential mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}}$.*

Proof: (sketch) Assume inductively that we have correctly computed the weights $w_{i',U,\epsilon}$ for $i' > i$. Whenever $U \ge L_i + 1$ then Lemma 3 implies that we obtain the correct weight $w_{i,U,\epsilon}$ in Algorithm 2. Similarly, whenever $U = L_i$ Lemma 2 implies that we will compute the correct value of $w_{i,U,\epsilon}$. For the base cases we note that whenever $U \notin [L_i, U_i]$ we must have $w_{i,U,\epsilon} = 0$ and whenever $U = 0$ and $f_i = 0$ we have

$$w_{i,U,\epsilon} = \sum_{n\ge 0} \sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^i} \\ x_1 = U}} \exp\left(-\epsilon\cdot\mathbf{dist}\left(f^i, x\right)\right) = e^{-\epsilon\cdot 0} = 1 \ .$$

Algorithm 2 Exponential Mechanism: Preprocessing Stage

```
function CompWeights($U, i, d, f, \epsilon$)
    $key \leftarrow (U, i)$
    if $DP.Contains(key)$ then
        return $DP.GetValue(key)$
    $U_i \leftarrow$ GetUpperBound($i, d, f$), and
    $L_i \leftarrow$ GetLowerBound($i, d, f$).
    if $U \notin [L_i, U_i]$ then return $0$
    if $f_i = 0$ & $U = 0$ then return $1$
    $U_{i+1} \leftarrow$ GetUpperBound($i+1, d, f$), and
    $L_{i+1} \leftarrow$ GetLowerBound($i+1, d, f$).
    $val \leftarrow 0$
    if $U \geq L_i + 1$ then
        $\Delta(U, i) \leftarrow \begin{cases} 1, & \text{if } U \geq f_i + 1 \\ -1, & \text{otherwise.} \end{cases}$
        $w_{i,U-1,\epsilon} \leftarrow$ CompWeights($U-1, i, d, f, \epsilon$)
        $val \leftarrow val + e^{-\epsilon \times \Delta(U,i)/2} \times w_{i,U-1,\epsilon}$
        if $U \leq U_{i+1}$ then
            $w_{i+1,U,\epsilon} \leftarrow$ CompWeights($U, i+1, d, f, \epsilon$)
            $val \leftarrow val + e^{-\epsilon|U-f_i|/2} \times w_{i+1,U,\epsilon}$
    else
        for $U' = L_i + 1, \ldots, \min\{U, U_{i+1}\}$ do
            $W_{U'} \leftarrow$ CompWeights($U', i+1, d, f, \epsilon$)
            $W \leftarrow W + e^{-\epsilon|U-f_i|/2} \cdot W_{U'}$
    $DP.Add(key, val)$
    return $val$
```

Correctness of Algorithm 1 now follows from equation 2.

The most expensive step is filling in the dynamic programming table in Algorithm 2. This requires time and space $O(Nd)$ where $d = O\left(\frac{\sqrt{N} - \ln \delta}{\epsilon}\right)$. ■

V. Experimental Results

In this section we demonstrate that our dynamic programming based implementation of the exponential mechanism is efficient enough to release password frequency statistics for real world datasets.

A. Experimental Setup

1) Implementation: We implemented Algorithm 1 in C#. Our experiments were run on a computer with a 3.00 GHz Intel Core i7-4610M CPU and 16 GB of RAM.

2) Password Dataset: We used data from the RockYou password breach [20] to analyze Algorithm 1 empirically. The RockYou dataset contains passwords from $N \approx 32.6$ million RockYou users, but the dataset only contains ≈ 14.3 million different passwords. Approximately, 11.9 million RockYou users selected a unique password. The other ≈ 2.5 million passwords were shared by multiple users. The most popular password (pwd_1 = '123456') was shared by ≈ 0.3 million RockYou users ($p_1 \approx 0.01$). These passwords were leaked when hackers broke into the RockYou authentication server where the passwords were unfortunately stored in the clear. RockYou users were not required to select passwords containing a mix of upper and lower case letters or a mix of numbers and letters.

Algorithm 3 Exponential Mechanism: Helper Functions

```
▷ Returns $U_i \doteq \max\{x_i \mid x \in \mathcal{P} \wedge \mathbf{dist}(x, f) \leq d\}$.
  Runs in time $O(d)$.
function GetUpperBound($i, d, f$)
    $U \leftarrow f_i$                                  ▷ Current height
    $A \leftarrow 2d$                    ▷ Remaining addition budget

    ▷ Find how much we could increase $f_i$ by adding $A$
      items. To ensure that $f_j \geq f_{j+1}$ for all $j$ we may need
      to use some of our addition budget to increase $f_j$ for
      some $j < i$ along the way.
    while $A \geq 0$ do
        $j \leftarrow \min\{j \leq i \mid f_j \leq U\}$
        ▷ We can find $j$ in time $O(\log(i - j + 1))$ using
          binary search, and we reduce potential (e.g., $A$)
          by $i - j + 1$ in each iteration.
        $\Delta A \leftarrow i - j + 1$
        $A \leftarrow A - \Delta A$
        if $A \geq 0$ then $U \leftarrow U + 1$
        else return $U$
    return $U$

▷ Returns $L_i \doteq \min\{x_i \mid x \in \mathcal{P} \wedge \mathbf{dist}(x, f) \leq d\}$.
  Runs in time $O(d)$.
function GetLowerBound($i, d, f$)
    $L \leftarrow f_i$                                  ▷ Current height
    $R \leftarrow 2d$                     ▷ Remaining removal budget

    ▷ Find how much we could increase $f_i$ by removing $R$
      items. To ensure that $f_j \geq f_{j+1}$ for all $j$ we may need
      to use some of our removal budget to decrease $f_j$ for
      some $j < i$ along the way.
    while $R \geq 0 \wedge L > 0$ do
        $j \leftarrow \max\{j \geq i \mid f_j \geq L\}$
        ▷ We can find $j$ in time $O(\log(i - j + 1))$ using
          binary search, and we reduce potential (e.g., $R$)
          by $i - j + 1$ in each iteration.
        $\Delta R \leftarrow j - i + 1$
        $R \leftarrow R - \Delta R$
        if $R \geq 0 \wedge L > 0$ then $L \leftarrow L - 1$
    return $L$
```

3) Experiment: In each of our experiments we ran Algorithm 1 to sample from the restricted exponential mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}^d}$ using the restricted set family $\mathcal{S}^d$ from Section IV. In each of our experiments we varied the parameter $\epsilon \in \{8, 4, 2, 1, 0.5, 0.25, 0.2, 0.1, 0.05, 0.02, 0.002\}$ and we fixed $\delta = 2^{-100}$. We recall that Lemma 1 implies that $\mathcal{S}^d$ is at least δ-negligible for $d \geq O\left(\frac{\sqrt{N} - \ln \delta}{\epsilon}\right)$. Theoretically, the number of entries in the dynamic programming table is bounded by $2d^2 + 2dN$, but this number can be even smaller in practice. Figure 1 shows how the number of entries in our dynamic programming table increases with $1/\epsilon$ — the $1/\epsilon$ axis is shown in log scale so that each point is visible. For the smallest values of ϵ the dynamic programming table was a little bit too big to fit in memory. We overcame this problem by only storing

ϵ	$\overline{\mathbf{dist}}$	σ	max	min
8	28.8	4.52	42	18
4	228.8	11.79	250	192
2	663.5	20.30	717	623
1	1,330.5	37.17	1,416	1,241
0.5	2,328.2	56.3	2,479	2,199
0.25	3,768.1	89.9	3,944	3,427
0.2	4,355.7	116.7	4,638	4,070
0.1	6,752.6	243.7	7,450	6,237
0.05	10,204.2	356.7	11,143	9,305
0.02	17,542.9	699.1	19,661	16,101
0.002	61,937.1	4,004.9	71,387	53,658

TABLE II: Summary of Results. # samples = 100 in each trial, and $\delta \leq 2^{-100}$

pieces of the dynamic programming table in memory and recomputing the rest on the fly. Algorithm 1 consists of two phases: a pre-processing stage and the actual sampling phase. While the pre-processing stage, **CompWeights**, tended to be a bit more expensive we only need to complete this stage once for each value of ϵ. Once the pre-processing stage was complete we can run the sampling algorithm **Sample** to draw many independent samples from $\mathcal{E}^{\epsilon}_{\mathcal{S}^d}$. For each value of ϵ we drew a set $\tilde{F}_\epsilon \subset \mathcal{P}$ of $\left|\tilde{F}_\epsilon\right| = 100$ independent samples $\mathcal{E}^{\epsilon}_{\mathcal{S}^d}$ and measured the average distance

$$\overline{\mathbf{dist}} = \sum_{\tilde{f} \in \tilde{F}_\epsilon} \mathbf{dist}\left(f, \tilde{f}\right) \ .$$

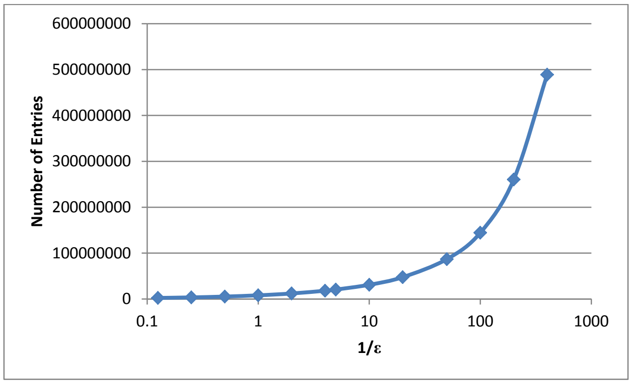

Fig. 1: $1/\epsilon$ vs. Dynamic Programming Table Size

B. Results

Our results are summarized in Figure 2 and in Table II. Figure 2 plots the average normalized cumulative distance, $\overline{\mathbf{dist}}/N$, versus $1/\epsilon$. The $1/\epsilon$ axis is shown in log-scale so that each of the points is visible. Privacy increases as we move to the right. The red bars indicate the maximum and minimum values of $\mathbf{dist}\left(f, \tilde{f}\right)/N$ over all samples $\tilde{f} \in \tilde{F}_\epsilon$. Table II shows the precise numeric values of $\overline{\mathbf{dist}}$, σ, max and min.

1) Discussion: In each of our experiments the cumulative error was quite small. Even for very high privacy parameters (e.g., $\epsilon = 0.002$) the normalized cumulative error (i.e. $\mathbf{dist}\left(f, \tilde{f}\right)/N$) was less than 2.2×10^{-3} in all hundred trials. Thus, the sanitized frequency list $\tilde{f}$ will still be useful for password researchers (e.g., $\tilde{\lambda}_\beta$ will be accurate estimate of λ_β the number of passwords that could be compromised with β guesses per user).

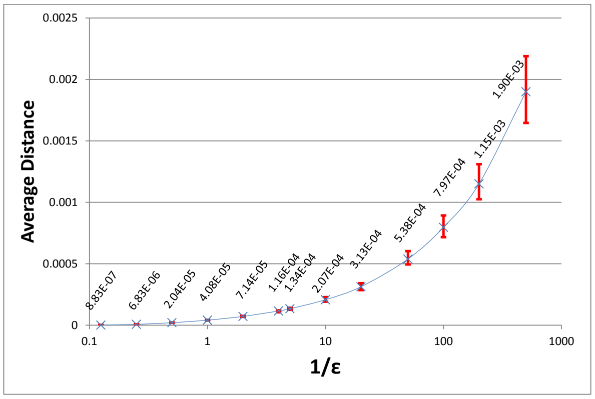

Fig. 2: $1/\epsilon$ vs. $\overline{\mathbf{dist}}/N$. Privacy increases as we move to the right. Red bars indicate maximum and minimum values of **dist** over 100 trials.

a) Efficiency Considerations: As we saw in Figure 1 it becomes increasingly challenging to fit the dynamic programming table into memory as $1/\epsilon$ increases. We believe that the memory requirements could be significantly reduced in practice. Currently, we need to set $d = O\left(\frac{\sqrt{N}+\ln(1/\delta)}{\epsilon}\right)$ so that Lemma 1 guarantees that the set family $\mathcal{S}_d$ is δ-negligible. However, we found strong empirical evidence that the sets S^d_f are δ-negligible for much smaller values of d. For example, we found that outputs from the mechanisms $\mathcal{E}^{\epsilon}_{\mathcal{S}_{d/100}}$ and $\mathcal{E}^{\epsilon}_{\mathcal{S}_d}$ appear to be indistinguishable. Future work might explore the possibility of computing instance specific upper bounds on the value of d. Such bounds might drastically reduce the space requirements of our sampling algorithm.

VI. Yahoo! Dataset

With Yahoo!'s permission we used our differentially private algorithm to release password frequency statistics for approximately 70 million Yahoo! passwords originally collected by Bonneau in 2012 [5]. The data is available for download at https://goo.gl/lG6MHd, with the main data file having a SHA-256 digest of `061137ea3cc129c7d9f501295cb194e0c6fa158ac-ac702f893cba3cfd5f44efe`. This dataset includes frequency lists for many subsets of the data for different groups of users. Comparing statistics between different groups enabled many novel observations, such as a stronger distribution of passwords chosen by older users or those with a payment card stored on file.

a) Original privacy considerations: To mitigate security risks inherent to plaintext password data, these frequency lists were collected by deploying a proxy server in front of live login servers to record a pseudorandom function[4] $f\left(pwd_u, s\right)$ of each password used in a successful authentication (only recording an observation once per individual user). Each observation was added to multiple frequency lists based on

[4] HMAC-SHA256 was used as the pseudorandom function f.

demographic facts of the user logging in. The secret key s used in computing the pseudorandom function was discarded after collecting the frequency lists and before they were made available to the researcher, destroying information about the individual passwords and leaving only a set of frequency lists.

While Yahoo! allowed Bonneau [5] to publish a summary analysis of these passwords, they declined to allow publication of any of the password frequency lists themselves.

b) Selecting ϵ: Because we are releasing frequency lists for multiple demographic groups we need to take into consideration the fact that a single user's data might occur in multiple groups. While there are 52 different groups of users including the "All Users" group an individual user could occur in at most 23 different groups. For example, there are five different age groups ($13-24$, $25-34$, $35-44$, $45-54$ and ≥ 55), but an individual user can be a member of at most one of these groups. Differential privacy offers a nice composition guarantee in this instance: if we run an $(\epsilon/23, \delta/23)$-differentially private algorithm to release the password frequency lists for each group then the composite mechanism preserves (ϵ, δ)-differential privacy [13]. We set $\epsilon_{all} = 0.25$ for the "All Passwords" group and we set $\epsilon' = 0.25/22$ for a composite privacy level of $\epsilon = \epsilon_{all} + 22 \cdot \epsilon' = 0.5$. In each case we set $\delta' < 2^{-100}$ so that the composite value of $\delta = 23 \cdot \delta'$ was negligibly small.

c) Results: For each frequency list to be published, we have re-computed the same statistics computed in Bonneau's original paper [5] using our sanitized data. The results are shown in Table III. The metrics are defined in [5], but briefly:

- N represents the total of passwords observed (M in the notation of [5])
- $\tilde{\lambda}_\beta$ represents[5] the size of a uniform distribution with equivalent resistance to guessing by an attacker with a fixed budget of β guesses, expressed in bits. $\tilde{\lambda}_1$ is equivalent to the min-entropy H_∞.
- $\tilde{G}_\alpha$ represents the size of a uniform distribution with equivalent resistance to guessing by an attacker who makes enough guesses to have a minimum probability α of success, expressed in bits.

Of the 52 distributions published, all but 3 produced virtually indistinguishable results from those in the original analysis, differing from the published statistics (except for the apparent number of users) by no more than 0.1 bits. Part of this is the fact that these metrics were originally reported on a logarithmic scale (e.g. "in bits"). Fine-grained data was not published and it was argued that this was not necessary or useful for practical security analysis. The exceptions (Korean language users, users with more than 5 password resets, and users with more than 5 password changes) were all relatively small groups featuring fewer than 1 million users.

d) Discussion: Once again the empirical accuracy of our algorithm greatly exceeded our theoretical error bounds. Our theoretical results already imply reasonably strong bounds on the error of password statistics like $\tilde{\lambda}_\beta$ or $\tilde{G}_\alpha$. In particular we know that with high probability we will have $\mathbf{dist}\left(f, \tilde{f}\right) \leq d$ with $d = O(\sqrt{N}/\epsilon)$. For the purposes of estimating $\tilde{\lambda}_\beta$ or $\tilde{G}_\alpha$ the worst case is when $\tilde{f}_1 = f_1 + 2 \cdot d$. In this case our estimate of the min-entropy $\tilde{\lambda}_1$ might be off by a factor of at most

$$\log_2\left(\frac{N}{f_1}\right) - \log_2\left(\frac{N}{f_1 + 2 \cdot d}\right) = \log_2\left(\frac{f_1 + 2 \cdot d}{f_1}\right) ,$$

bits. Even this worst case bound is relatively small for password datasets with a larger f_1. If we assume that $f_1/N \approx 1\%$, as was observed for most of the frequency lists in the Yahoo! corpus, then the theoretical error would scale as $\log_2\left(\frac{0.01 \cdot N}{0.01 \cdot N + 2 + \sqrt{N}/\epsilon}\right)$. Thus the error approaches 0 as $N \to \infty$, but it might still noticeable for smaller sample sizes (e.g., Korean language users, users with more than 5 password resets, and users with more than 5 password changes).

VII. Related Work

a) Privacy Models: De-anonymization attacks [28], [35] highlight the problems with heuristic approaches to data sanitization (e.g., omitting "personally identifiable information" from the dataset). In response to de-anonymization attacks researchers began to propose formal data privacy models like k-anonymity [35], [32] and ℓ-diversity [25]. We follow the framework of differential privacy [12], because it provides the strongest and most rigorous privacy guarantees. Unlike k-anonymity and ℓ-diversity, differential privacy is composable and the privacy guarantees hold against an adversary with arbitrary amount of background knowledge. Kifer and Machanavajjhala [22] argued that even differential privacy might not provide sufficient privacy guarantees whenever we want to hide the *influence* of an individual. Indeed, this is an important consideration in settings where an individual can influence the records of *many* other individuals (e.g., genetic datasets). However, in our passwords setting it is reasonable to assume the user's secret decision to select one password instead of another will not influence the passwords that other users select[6].

b) Differential Privacy: There is a vast body of literature around differential privacy. This work has explored tighter composition bounds [14], impossibility results[9], differential privacy for social networks etc.. We do not have enough space to discuss the vast literature on differential privacy. Instead we refer an interested reader to the textbook of Dwork and Roth [13] which contains an excellent summary of much of this research. While differentially private algorithms typically perturb the answers to queries to preserve privacy [12], small perturbations do not necessarily destroy the utility of these answers for a data analyst. In fact, Dwork et al. [11] recently showed that differential privacy can have the positive side effect of preventing over-fitting!

Hay et al. [18] considered the problem of releasing the degree distribution of a graph in a differentially private manner. At a fundamental level a degree distribution is an integer partition. We note our algorithm provides stronger utility

[5] Note that earlier we used $\tilde{\lambda}$ to represent λ computed on a perturbed dataset. In this section we omit this as it is clear from context if this value is computed on the original or perturbed data.

[6] We are assuming of course that the same human doesn't create hundreds of duplicate accounts.

	Original Data						**Sanitized Data**					
	N	$\hat{\hat{\lambda}}_1$	$\hat{\hat{\lambda}}_{10}$	$\hat{\hat{\lambda}}_{100}$	$\hat{\hat{G}}_{0.25}$	$\hat{\hat{G}}_{0.5}$	$\tilde{N}$	$\hat{\hat{\lambda}}_1$	$\hat{\hat{\lambda}}_{10}$	$\hat{\hat{\lambda}}_{100}$	$\hat{\hat{G}}_{0.25}$	$\hat{\hat{G}}_{0.5}$
all passwords	69301337	6.5	9.1	11.4	17.6	21.6	*69299074*	6.5	9.1	11.4	17.6	21.6
gender (self-reported)												
female	30545765	6.9	9.3	11.5	17.2	21.1	*30533392*	6.9	9.3	11.5	17.2	21.1
male	38624554	6.3	8.8	11.3	17.7	21.8	*38617094*	6.3	8.8	11.3	17.7	21.8
age (self-reported)												
13–24	18199547	6.3	8.7	11.1	16.7	20.9	*18198383*	6.3	8.7	11.1	16.7	20.9
25–34	22380694	6.2	8.8	11.2	17.1	21.2	*22382478*	6.2	8.8	11.2	17.1	21.2
35–44	12983954	6.8	9.4	11.7	17.4	21.3	*12985267*	6.8	9.4	11.7	17.4	21.3
45–54	8075887	7.3	9.8	11.8	17.3	21.3	*8075158*	7.3	9.8	11.8	*17.4*	21.3
$\geq$ 55	7110689	7.5	9.8	11.8	17.3	21.4	*7110081*	7.5	9.8	11.8	17.3	21.4
language preference												
Chinese	1564364	6.5	8.6	11.1	17.3	22.0	*1571348*	6.5	*8.7*	11.1	*17.2*	*21.8*
German	1127474	7.4	9.7	11.3	15.8	19.7	*1132306*	7.4	*9.6*	*11.2*	15.8	*19.5*
English	55805764	6.5	9.0	11.3	17.4	21.5	*55798806*	6.5	9.0	11.3	17.4	21.5
French	2084219	6.9	9.0	10.9	14.8	18.6	*2086600*	6.9	9.0	10.9	14.8	18.6
Indonesian	1061540	5.5	7.9	10.2	14.3	17.0	*1063979*	5.5	7.9	10.2	*14.2*	17.0
Italian	811133	6.8	9.0	10.7	14.5	18.0	*812764*	*6.9*	9.0	10.7	14.5	18.0
Korean	530759	7.5	9.5	11.7	18.1	22.7	*560670*	*7.6*	*9.1*	*10.7*	*16.8*	*21.6*
Portuguese	2060256	6.5	9.0	11.0	15.6	18.8	*2064086*	6.5	9.0	11.0	15.6	18.8
Spanish	3065901	6.6	9.1	11.0	15.6	19.7	*3068101*	*6.7*	*9.2*	11.0	15.6	19.7
tenure of account												
$\leq$ 1 y	5182527	6.9	9.1	11.7	18.0	22.5	*5183617*	6.9	*9.2*	11.7	18.0	22.5
1–2 years	5182527	6.9	9.1	11.7	18.0	22.5	*5182550*	6.9	*9.2*	11.7	18.0	22.5
2–3 years	12261556	6.2	8.6	11.2	17.7	21.8	*12260532*	6.2	8.6	11.2	17.7	21.8
3–4 years	10332348	6.2	8.8	11.3	17.5	21.6	*10334603*	6.2	8.8	11.3	17.5	21.6
4–5 years	9290840	6.1	8.8	11.2	17.2	21.2	*9287123*	6.1	8.8	11.2	17.2	21.2
$\geq$ 5 years	29104856	6.8	9.3	11.5	17.2	21.2	*29093199*	6.8	9.3	11.5	17.2	21.2
password requirements at registration												
none	20434875	6.6	9.2	11.4	16.8	20.7	*20426607*	6.6	9.2	11.4	16.8	20.7
6 char. minimum	13332334	6.5	9.0	11.4	17.6	21.6	*13329231*	6.5	9.0	11.4	17.6	21.6
last recorded login												
$<$ 30 days	32627777	6.5	9.0	11.4	17.5	21.5	*32617294*	6.5	9.0	11.4	17.5	21.5
$<$ 90 days	55777259	6.5	9.0	11.4	17.5	21.5	55777259	6.5	9.0	11.4	17.5	21.5
$>$ 90 days	8212643	7.0	9.5	11.7	17.7	21.9	8212643	7.0	9.5	11.7	17.7	21.9
number of login locations												
1	16447906	6.0	8.6	11.2	17.1	21.1	*16444143*	6.0	8.6	11.2	17.1	21.1
$\geq$2	52853431	6.7	9.2	11.5	17.7	21.7	*52838680*	6.7	9.2	11.5	17.7	21.7
$\geq$ 10	17146723	7.3	9.7	11.8	18.3	22.6	*17142893*	7.3	9.7	11.8	18.3	22.6
number of password changes												
none	52117133	6.2	8.8	11.2	17.1	20.9	*52104736*	6.2	8.8	11.2	17.1	20.9
1	9608164	8.3	10.4	12.3	18.8	23.2	*9605232*	8.3	10.4	12.3	18.8	23.2
$>$1	7576040	8.6	10.7	12.5	19.5	24.2	*7579817*	8.6	10.7	12.5	19.5	24.2
$\geq$5	930035	9.1	10.9	12.7	19.7	25.9	*953730*	9.1	*10.3*	*11.6*	*19.0*	*25.4*
number of password resets (forgotten password)												
none	61805038	6.4	8.9	11.3	17.3	21.3	*61788266*	6.4	8.9	11.3	17.3	21.3
1	4378667	8.2	10.5	12.5	19.2	23.8	*4382442*	8.2	10.5	12.5	19.2	*23.7*
$>$1	3117632	8.7	10.8	12.8	19.7	24.6	*3123591*	8.7	10.8	*12.7*	*19.6*	*24.5*
$\geq$5	387469	8.7	10.6	12.8	19.9	26.6	*424250*	8.7	*9.3*	*10.5*	*17.0*	*23.9*
amount of data stored with Yahoo!												
1st quartile	9830792	5.6	8.2	10.8	17.3	21.5	*9825880*	5.6	8.2	10.8	17.3	21.5
2nd quartile	20702119	6.3	8.8	11.3	17.5	21.5	*20700076*	6.3	8.8	11.3	17.5	21.5
3rd quartile	21307618	6.8	9.3	11.5	17.5	21.4	*21305961*	6.8	9.3	11.5	17.5	21.4
4th quartile	17447029	7.6	10.0	11.9	17.8	22.0	*17443212*	7.6	10.0	11.9	17.8	22.0
usage of different Yahoo!features												
media sharing	5976663	7.7	10.1	12.0	18.0	22.3	*5978011*	7.7	*10.0*	12.0	18.0	22.3
retail	2139160	8.8	10.5	11.9	16.8	21.4	*2141012*	8.8	10.5	11.9	16.8	*21.3*
webmail	15965774	6.3	8.8	11.3	17.4	21.2	*15961586*	6.3	8.8	11.3	17.4	21.2
chat	37337890	6.2	8.7	11.2	17.1	21.2	*37328344*	6.2	8.7	11.2	17.1	21.2
social networking	14204900	7.1	9.6	11.7	17.7	21.8	*14206515*	7.1	9.6	11.7	17.7	21.8
mobile access	20676566	6.7	9.3	11.4	17.1	21.1	*20673120*	6.7	9.3	11.4	17.1	21.1
Android client	1359713	8.3	10.3	12.0	17.3	21.5	*1365725*	8.3	10.3	*11.8*	*17.2*	*21.4*
iPhone client	6222547	8.1	10.1	11.9	17.6	21.6	*6225078*	8.1	10.1	11.9	17.6	21.6
RIM client	3843404	7.6	10.0	11.8	17.2	21.1	*3846342*	7.6	10.0	11.8	17.2	*21.0*

TABLE III: Statistics for the original Yahoo! password distributions collected and published by Bonneau in 2012 [5] (left columns) and recomputation of those statistics using our public release of the same data sanitized with our differential privacy approach (right columns). Statistics for the sanitized distribution are italicized when they are different from those originally published (to the precision originally published).

guarantees than [18]. Their algorithm has the theoretical guarantee that $\sum_i \left|f_i - \tilde{f}_i\right|^2 = O\left(\sqrt{N}\log^3 N/\epsilon^2\right)$, while our algorithm provides the much stronger guarantee for L1 error $\sum_i \left|f_i - \tilde{f}_i\right| = O\left(\sqrt{N}/\epsilon\right)$. The former requirement would be satisfied by setting $\tilde{f}_i = 1 + f_i$ for each $i < \sqrt{N}\log^3 N/\epsilon^2$, but then we would have cumulative error $\mathbf{dist}\left(f, \tilde{f}\right) = O\left(\sqrt{N}\log^3 N/\epsilon^2\right)$. Thus, our algorithms could also prove useful in differentially private analysis of social networks.

c) Exponential Mechanism: The exponential mechanism was proposed by McSherry and Talwar [27]. Blum et al. [4] showed that the exponential mechanism could be used to answer large classes of queries accurately provided that the dimension of the dataset was reasonably small and that the number of records in the dataset was large[7].

We note that the mechanism from [4] was computationally inefficient and it only works for datasets with reasonably low dimension. Hardt et al. [16] found an alternative differentially private algorithm that can provide accurate answers to many queries given a very low dimensional dataset (e.g., there algorithm runs in time proportional to $|\mathcal{X}|$, where $\mathcal{X}$ denotes the set of all possible data records). However, Ullman [36] showed that, in general, it is computationally hard to answer $O\left(n^{2+\epsilon}\right)$ adaptively chosen queries unless cryptographic traitor-tracing schemes can be broken. Efficient implementations of the exponential mechanism remain rare. We used dynamic programming algorithm to provide an efficient implementation of the exponential mechanism for integer partitions. Could similar techniques be used to implement the exponential mechanism in other settings?

d) Password Composition Policies: Many organizations require their users to follow strict rules when selecting passwords (e.g., use a mix of upper/lower case letters, include numbers and change the password frequently) [33]. Several empirical studies have shown that these password composition policies frustrate users and result in more forgotten passwords [24], [15] and may sometimes even cause users to select weaker passwords [23], [3]. It is our hope that our differentially private algorithm for releasing password frequency statistics might lead to the collection and release of empirical data that would improve our understanding of the security impact of these policies.

e) Measuring Password Strength: Many metrics have been proposed for measuring the resistance of a password distribution to guessing. The most general are *non-parametric*, that is, they don't assume anything about the structure of the password distribution and require knowledge of the probability of each password independently.[8] This allows measuring an ideal attacker who is able to always guess the most likely password which hasn't already been tried. Importantly, non-parametric models can always be computed from a password frequency list alone and do not require analyzing the semantic meaning or representation of passwords themselves.

In this model, guessing-entropy [34], [26]

$$\sum_{i=1}^{n} i \cdot p_i$$

measures the average number of guesses needed to crack a single password. While intuitive, Bonneau demonstrated that this metric produces absurd results on real password datasets due to the presence of extremely difficult-to-guess outliers [6]. Boztaş [7] proposed a metric called β-guesswork, which measured the success rate for an adversary with β guesses per account $\sum_{i=1}^{\beta} p_i$. Pliam proposed a similar metric called α-guesswork [30]. This metric measure how many guesses the adversary would need to achieve success rate α. Min-entropy $H_\infty = -\log p_1$ measures the likelihood of the most common password and represents a tool for analyzing worst-case security [3].

Bonneau argued that no single metric is adequate as each is inherently bound to a specific attacker model and that fair comparison requires considering the entire *guessing curve* (i.e. a plot of $\tilde{\lambda}_\beta$ for all values of β). The original published analysis of the Yahoo! data included several guessing metrics for different attacker models. In Section VI we re-computed these metrics on the Yahoo! data using the perturbed frequency lists produced by our algorithm, finding that our results were very close. This may be construed as evidence that the original statistics published by Bonneau [5] were granular enough to already provide reasonable privacy guarantees to individuals. For example, it is possible that many of these statistics already satisfied the notion of of noiseless privacy which was introduced by Bhaskar et al. [1].

VIII. Conclusions and Future Work

We presented an efficient mechanism for releasing perturbed password frequency lists. Our algorithm guarantees minimal distortion while satisfying a strong security notion (approximate differential privacy). Our empirical evaluation over the RockYou dataset demonstrated that our algorithm is scalable and that it introduces low distortion in practice. This appealing combination of security and utility convinced Yahoo! to release the password frequency lists from a dataset represnting 70 million users. Our evaluation on this data further demonstrates the practicality of our approach. At a high level, analysis of the perturbed data would have produced the same scientific conclusions (with nearly identical computed statistics) and as analysis on the raw data itself.

Thus, using our methods the data could have been publicly released in perturbed form rather than shared on a limited basis with a trusted researcher. As it were, this original collaboration

[7] More formally, given a class $\mathcal{Q}$ of counting queries, the exponential mechanism could be used to release a small sanitized dataset, which would, with high probability, provide accurate answers for *every* query $q \in \mathcal{Q}$ provided that their are enough records in the dataset. To provide α-accurate answers they needed a dataset of size $n = \tilde{O}\left(\frac{\log(|\mathcal{Q}||\mathcal{X}|)+\log\delta^{-1}}{\alpha^3\epsilon}\right)$, where $VCDim\left(\mathcal{Q}\right) \leq \log|\mathcal{Q}|$ is the Vapnik-Chervonekis dimension of the query class $\mathcal{Q}$, $\mathcal{X}$ is the set of all possible data records and ϵ is the privacy parameter.

[8] By contrast, other metrics might assume the password distribution takes a specific form, such as a Zipf distribution, and try to estimate parameters from observed data. Other metrics might assume that pieces of passwords representing numbers and letters are chosen independently and can therefore be estimated independently. We ignore such parametric models in our discussion.

only occurred because a student researcher with an interest in password statistics happened to be taken on as an intern with the Yahoo! authentication team. This experiment has never been repeated at Yahoo! or elsewhere despite the possibility of learning more about password selection by collecting such data. Other password researchers are usually left to work with whatever data is leaked publicly from hapless organizations who suffer data breaches.

This situation is bad for scientific progress. We hope that, given our differential privacy approach, other organizations will follow Yahoo!'s lead and make new password datasets publicly available to the research community.

Acknowledgments

This work was completed in part while the first author was visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant #CNS-1523467. The research was also supported by an AFOSR MURI on Science of Cybersecurity as well as grants from the NSF TRUST Center. Joseph Bonneau is supported by a Secure Usability Fellowship from the Open Technology Fund and Simply Secure. The views expressed in this paper are those of the authors and do not necessarily reflect the views of the Simons Institute or the National Science Foundation.

References

[1] R. Bhaskar, A. Bhowmick, V. Goyal, S. Laxman, and A. Thakurta, "Noiseless database privacy," in *Advances in Cryptology–ASIACRYPT 2011*. Springer, 2011, pp. 215–232.

[2] R. Biddle, S. Chiasson, and P. Van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Surveys (CSUR)*, vol. 44, no. 4, p. 19, 2012.

[3] J. Blocki, S. Komanduri, A. Procaccia, and O. Sheffet, "Optimizing password composition policies," in *Proceedings of the fourteenth ACM conference on Electronic commerce*. ACM, 2013, pp. 105–122.

[4] A. Blum, K. Ligett, and A. Roth, "A learning theory approach to noninteractive database privacy," *Journal of the ACM (JACM)*, vol. 60, no. 2, p. 12, 2013.

[5] J. Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords," in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 538–552.

[6] ——, "Guessing human-chosen secrets," Ph.D. dissertation, University of Cambridge, May 2012. [Online]. Available: http://www.jbonneau.com/doc/2012-jbonneau-phd_thesis.pdf

[7] S. Boztas, "Entropies, guessing, and cryptography," *Department of Mathematics, Royal Melbourne Institute of Technology, Tech. Rep*, vol. 6, 1999.

[8] S. Brostoff and M. Sasse, "Are Passfaces more usable than passwords: A field trial investigation," in *People and Computers XIV-Usability or Else: Proceedings of HCI*, 2000, pp. 405–424.

[9] M. Bun, J. Ullman, and S. Vadhan, "Fingerprinting codes and the price of approximate differential privacy," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*. ACM, 2014, pp. 1–10.

[10] W. de Azevedo Pribitkin, "Simple upper bounds for partition functions," *The Ramanujan Journal*, vol. 18, no. 1, pp. 113–119, 2009. [Online]. Available: http://dx.doi.org/10.1007/s11139-007-9022-z

[11] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth, "Generalization in adaptive data analysis and holdout reuse," *arXiv preprint arXiv:1506.02629*, 2015.

[12] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography*. Springer, 2006, pp. 265–284.

[13] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2013.

[14] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*. IEEE, 2010, pp. 51–60.

[15] D. Florencio and C. Herley, "A large-scale study of web password habits," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 657–666.

[16] M. Hardt, K. Ligett, and F. McSherry, "A simple and practical algorithm for differentially private data release," in *Advances in Neural Information Processing Systems*, 2012, pp. 2339–2347.

[17] G. H. Hardy and S. Ramanujan, "Asymptotic formulæ in combinatory analysis," *Proceedings of the London Mathematical Society*, vol. 2, no. 1, pp. 75–115, 1918.

[18] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE, 2009, pp. 169–178.

[19] C. Herley and P. van Oorschot, "A research agenda acknowledging the persistence of passwords," *IEEE Symposium Security and Privacy*, vol. 10, no. 1, pp. 28–36, 2012.

[20] Imperva, "Consumer password worst practices," 2010, retrived 1/22/2013. [Online]. Available: http://www.imperva.com/docs/WP_Consumer_Password_Worst_Practices.pdf

[21] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, "The design and analysis of graphical passwords," in *Proceedings of the 8th conference on USENIX Security Symposium - Volume 8*. Berkeley, CA, USA: USENIX Association, 1999, pp. 1–1. [Online]. Available: http://portal.acm.org/citation.cfm?id=1251421.1251422

[22] D. Kifer and A. Machanavajjhala, "No free lunch in data privacy," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 2011, pp. 193–204.

[23] S. Komanduri, R. Shay, P. Kelley, M. Mazurek, L. Bauer, N. Christin, L. Cranor, and S. Egelman, "Of passwords and people: measuring the effect of password-composition policies," in *Proceedings of the 2011 annual conference on Human factors in computing systems*. ACM, 2011, pp. 2595–2604.

[24] H. Kruger, T. Steyn, B. Medlin, and L. Drevin, "An empirical assessment of factors impeding effective password management," *Journal of Information Privacy and Security*, vol. 4, no. 4, pp. 45–59, 2008.

[25] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 3, 2007.

[26] J. Massey, "Guessing and entropy," in *Information Theory, 1994. Proceedings., 1994 IEEE International Symposium on*. IEEE, 1994, p. 204.

[27] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*. IEEE, 2007, pp. 94–103.

[28] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.

[29] C. Percival and S. Josefsson, "The scrypt password-based key derivation function," 2012.

[30] J. Pliam, "On the incomparability of entropy and marginal guesswork in brute-force attacks," *Progress in CryptologyINDOCRYPT 2000*, pp. 113–123, 2000.

[31] N. Provos and D. Mazieres, "Bcrypt algorithm."

[32] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," Technical report, SRI International, Tech. Rep., 1998.

[33] K. Scarfone and M. Souppaya, "Nist special publication 800-118: Guide to enterprise password management (draft)," April 2009.

[34] C. Shannon and W. Weaver, *The mathematical theory of communication*. Citeseer, 1959.

[35] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[36] J. Ullman, “Answering n {2+ o (1)} counting queries with differential privacy is hard,” in *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 2013, pp. 361–370.

Appendix

Reminder of Theorem 1. *[27] The exponential mechanism $\mathcal{E}^\epsilon$ preserves ϵ-differential privacy.*

Proof of Theorem 1. Suppose that $f \in \mathcal{P}$ and $f' \in \mathcal{P}$ are neighboring integer partitions. For any $x \in \mathcal{P}$ we have

$$\mathbf{dist}\left(f',x\right) - \frac{1}{2} \leq \mathbf{dist}\left(f,x\right) \leq \mathbf{dist}\left(f',x\right) + \frac{1}{2} \ .$$

Therefore,

$$\begin{aligned}\frac{\Pr_{y\leftarrow\mathcal{E}^\epsilon(f)}\left[x=y\right]}{\Pr_{y\leftarrow\mathcal{E}^\epsilon(f')}\left[x=y\right]} &= \frac{w_f \exp\left(-\epsilon\cdot\mathbf{dist}\left(f,x\right)\right)}{w_{f'}\exp\left(-\epsilon\cdot\mathbf{dist}\left(f',x\right)\right)}\\ &\leq \frac{w_f}{w_{f'}} e^{\epsilon\left|\mathbf{dist}(f,x)-\mathbf{dist}\left(f',x\right)\right|}\\ &\leq \left(\max_{\tilde{f}\in\mathcal{P}} \frac{e^{-\epsilon\cdot\mathbf{dist}\left(f,\tilde{f}\right)}}{e^{\epsilon\cdot\mathbf{dist}\left(f',\tilde{f}\right)}}\right) e^{\epsilon/2}\\ &\leq e^\epsilon \ .\end{aligned}$$

□

Reminder of Theorem 2. *Let $f \in \mathcal{P}\left(N\right)$, $\epsilon > \frac{48\pi^2}{\sqrt{N}}$ and $\delta \geq e^{1-\sqrt{N}/2}$ be given and let $\tilde{f} \leftarrow \mathcal{E}^\epsilon\left(f\right)$ denote a random sample from the exponential mechanism. Then except with probability δ we will have*

$$\mathbf{dist}\left(f,\tilde{f}\right) \leq \frac{c_1\sqrt{N} + c_2\ln\left(\frac{1}{\delta}\right)}{\epsilon} \ ,$$

where $c_1 = 2\pi\sqrt{\frac{2}{3}} + o(1)$ and $c_2 = 2 + o(1)$ are constants.

We will use the following lemma to prove Theorem 2.

Lemma 4. $\left|\left\{f' \in \mathcal{P} \mid \mathbf{dist}\left(f,f'\right) = d/2\right\}\right| \leq 2\left|\mathcal{P}\left(N+d\right)\right|$ *for any $f \in \mathcal{P}\left(N\right)$*

Proof: Let $D_d = \left\{f' \in \mathcal{P} \mid \mathbf{dist}\left(f,f'\right) = d/2\right\}$ be the subset of all partitions f' at distance $d/2$ from f. We partition D_d into two sets D_d^+ and D_d^- such that $f' \in D_d^+$ if and only if $f_1' \geq f$. It suffices to show that there is a map $\mu : D_d \to \mathcal{P}\left(N+d\right)$ such that μ is injective over both of the domains D_d^+ and D_d^-. Given $f' \in D_d$ let N' denote the integer such that $f' \in \mathcal{P}\left(N'\right)$. We set $\mu(f') = z$ where $z_i = f_i'$ for $i > 1$ and $z_1 = f_1' + N + d - N'$. If $x, y \in D_d^+$ and $x \neq y$ then there must be some index $i > 1$ for which $x_i \neq y_i$. Suppose for contradiction that $x_i = y_i$ for all $i > 1$ then we must have $x_1 - f_1 = |x_1 - f_1| = |y_1 - f_1| = y_1 - f_1$ because $\mathbf{dist}\left(f,x\right) = d/2 = \mathbf{dist}\left(f,y\right)$ and $x_1, y_1 \geq f_1$. Thus, $x_i = y_i$ for all $i \geq 1$. Contradiction! Thus, $\mu(x) \neq \mu(y)$ because there exists $i > 1$ s.t $\mu(x)_i = x_i \neq y_i = \mu(y)_i$. A similar argument shows that $\mu(x) \neq \mu(y)$ whenever $x \neq y$ and $x, y \in D_d^-$. ■

The following fact will also be useful in our proof.

Fact 3. *For all $0 \leq x \leq 0.7968$ we have $x \geq 1 - e^{-2x}$.*

Proof: Consider the continuous function $f(x) = 1 - e^{-2x} - x$. It suffices to show that $f(x) \geq 0$ over the range $[0, 0.7968]$ since $1 - e^{-2x} = x + f(x) \geq x$. The derivative $f'(x) = 2e^{-2x} - 1$ has one positive zero at $x = \frac{\ln 2}{2} \approx 0.34657$. We have $f'(x) \geq 0$ for $x \in \left[0, \frac{\ln 2}{2}\right]$ so f is increasing over this range. Thus $f(x) \geq 0$ over this range since $f(0) = 0$. Similarly, $f'(x) \leq 0$ over the range $x \in \left[\frac{\ln 2}{2}, \infty\right]$ and thus $f(x) \geq 0$ over the range $\left[\frac{\ln 2}{2}, 0.7968\right]$ because $f(0.7968) \geq 0$. ■

Proof of Theorem 2. We rely on the following upper bound on $|\mathcal{P}(N)|$ due to Pribitkin [10]

$$\left|\mathcal{P}\left(N\right)\right| \leq \frac{1}{N^{3/4}}\exp\left(\pi\sqrt{\frac{2N}{3}}\right) \ .$$

Given an input partition $f \in \mathcal{P}\left(N\right)$ we have

$$\Pr_{y\leftarrow\mathcal{E}^\epsilon(f)}\left[y = \tilde{f}\right] = w_f \exp\left(-\epsilon\cdot\mathbf{dist}\left(f,\tilde{f}\right)\right) \ ,$$

for some constant w_f which denotes the probability that the exponential mechanism outputs f itself — because $\mathbf{dist}\left(f,f\right) = 0$. Combining Pribitkin’s [10] bound with Lemma 4 we can can upper bound the number of partitions $f' \in \mathcal{P}$ with $\mathbf{dist}\big(f,f'\big) = d/2$. Thus, by the union bound the probability that we output *any* f' with distance $\mathbf{dist}\big(f,f'\big) = d/2$ is at most $\Pr_{y\leftarrow\mathcal{E}^\epsilon(f)}\big[\mathbf{dist}\big(f,y\big) = d/2\big]$

$$\begin{aligned}&\leq 2w_f\left|\mathcal{P}\left(N+d\right)\right|\cdot\exp\left(-\epsilon\cdot d/2\right)\\ &\leq \left(\frac{2w_f}{\left(N+d\right)^{3/4}}\right)\exp\left(\frac{-\epsilon d}{2} + \pi\sqrt{\frac{2N+2d}{3}}\right)\\ &\doteq \gamma_d \ .\end{aligned}$$

Thus, it suffices to show that

$$\sum_{d=d_0}^{\infty}\frac{\gamma_d}{w_f} \leq \delta \ , \text{ where } d_0 = \frac{2\pi\sqrt{\left(\frac{2N}{3} + \frac{4\sqrt{N}}{\epsilon}\right)} + 2\ln\left(t\right)}{\epsilon}$$

and t will be selected later. Because $d + N \geq \frac{8\pi^2}{3\epsilon^2}$ we have

$$\begin{aligned}\frac{\gamma_{d+1}}{\gamma_d} &= \left(\frac{N+d}{N+d+1}\right)^{3/4}\frac{e^{-\epsilon(d+1)/2+\pi\sqrt{\frac{2N+2d+2}{3}}}}{e^{-\epsilon d/2+\pi\sqrt{\frac{2N+2d}{3}}}}\\ &= e^{-\epsilon/2}\left(\frac{N+d}{N+d+1}\right)^{3/4}\left(e^{\pi\sqrt{\frac{2}{3}}\left(\sqrt{N+d+1}-\sqrt{N+d}\right)}\right)\\ &\leq e^{-\epsilon/2+\frac{\pi\sqrt{2/3}}{\sqrt{N+d}+\sqrt{N+d+1}}} \leq e^{-\epsilon/2+\frac{\pi\sqrt{2/3}}{2\sqrt{N+d}}}\\ &\leq e^{-\epsilon/2+\frac{\epsilon}{4}} = e^{-\epsilon/4} \ .\end{aligned}$$

Therefore, we have

$$\begin{aligned}\sum_{d=d_0}^{\infty}\gamma_d &\leq \gamma_{d_0} + \sum_{t=1}^{\infty}\gamma_{d_0}\prod_{i=1}^{t}\left(\frac{\gamma_{d_0+i+1}}{\gamma_{d_0+i}}\right)\\ &\leq \sum_{t=0}^{\infty}\gamma_{d_0}\left(\max_{i\geq 1}\left\{\frac{\gamma_{d_0+i+1}}{\gamma_{d_0+i}}\right\}\right)^t\\ &\leq \sum_{t=0}^{\infty}\gamma_{d_0}e^{-t\epsilon/4} \leq \frac{\gamma_{d_0}}{1-e^{\epsilon/4}} \ .\end{aligned}$$

Assuming for now that $d_0 \leq \frac{\sqrt{6N}}{\epsilon}$ we get

$$\begin{aligned}
\frac{\gamma_{d_0}}{w_f} &= \frac{2\exp\left(\frac{-\epsilon d_0}{2} + \pi\sqrt{\frac{2N+2d_0}{3}}\right)}{\left(N+d_0\right)^{3/4}} \\
&= \frac{2\exp\left(\pi\sqrt{\frac{2}{3}}\left(\sqrt{N+d_0} - \sqrt{N+6\sqrt{N}/\epsilon}\right) - \ln t\right)}{\left(N+d_0\right)^{3/4}} \\
&\leq \frac{2}{t\left(N+d_0\right)^{3/4}} \ .
\end{aligned}$$

Thus, if we set $t = \frac{2}{(N+d_0)^{3/4}\left(1-e^{-\epsilon/4}\right)\delta}$ we have

$$\sum_{d=d_0}^{\infty} \frac{\gamma_d}{w_f} \leq \frac{2}{t\left(1-e^{\epsilon/4}\right)\left(N+d_0\right)^{3/4}} \leq \delta \ .$$

It remains to check that $d_0 \leq \frac{\sqrt{6N}}{\epsilon}$. Note that if $t \leq e^{\sqrt{N}/2}$ then we would be done since

$$\begin{aligned}
d_0 &= \frac{2\pi\sqrt{\left(\frac{2N}{3} + \frac{4\sqrt{N}}{\epsilon}\right)} + 2\ln\left(t\right)}{\epsilon} \\
&\leq \frac{\sqrt{\left(\frac{8\pi^2 N}{3} + \frac{16\pi^2 N}{48\pi^2}\right)} + 2\ln\left(t\right)}{\epsilon} \\
&\leq \frac{5.5\sqrt{N} + 2\ln\left(t\right)}{\epsilon} \leq \frac{6N}{\epsilon} \ .
\end{aligned}$$

Because $\epsilon \geq 48\pi^2/\sqrt{N}$ we have $\left(1-e^{-\epsilon/4}\right) \geq \left(1-e^{-12\pi^2/\sqrt{N}}\right)$. By setting $x = 6\pi^2/\sqrt{N}$ in Fact 3 we have $\left(1-e^{-12\pi^2/\sqrt{N}}\right) \geq 6\pi^2/\sqrt{N} \geq 2\left(N+d_0\right)^{-3/4}$ whenever $x < 0.7968$. Thus, if $x < 0.7968$ we have $t \leq \frac{\left(N+d_0\right)^{-3/4}}{6\pi^2/\sqrt{N}} \cdot \left(\frac{2}{\delta}\right) \leq \frac{2}{\delta} \leq e^{\sqrt{N}/2}$. Similarly, when $x > 0.7968$ we have $\left(1-e^{-\epsilon/4}\right) \geq \left(1-e^{-12\pi^2/\sqrt{N}}\right) = 1-e^{-2x} \geq 1-e^{-2\cdot 0.7968} \geq 0.7968$ so that $t \leq \frac{2\left(N+d_0\right)^{-3/4}}{0.7968\cdot\delta} \leq \frac{2.52}{\delta} \leq e^{\sqrt{N}/2}$.

□

Reminder of Theorem 3. *For any δ-negligible set family $\mathcal{S}$ the mechanism $\mathcal{E}^{\epsilon}_{\mathcal{S}}$ preserves (ϵ, δ')-differential privacy for $\delta' = \delta + e^{\epsilon}\delta$.*

Proof of Theorem 3. Let $f \sim f'$ be neighboring partitions. The statistical distance between the distributions $\mathcal{E}^{\epsilon}\left(f\right)$ and $\mathcal{E}^{\epsilon}_{\mathcal{S}}\left(f\right)$ is at most

$$\sum_{\tilde{f}\in\mathcal{P}} \left| \Pr_{y\leftarrow\mathcal{E}^{\epsilon}(f)}\left[y=\tilde{f}\right] - \Pr_{\mathcal{E}^{\epsilon}_{\mathcal{S}}(f)}\left[y=\tilde{f}\right] \right| \leq \delta \ .$$

Similarly, the statistical distance between the distributions $\mathcal{E}^{\epsilon}\left(f'\right)$ and $\mathcal{E}^{\epsilon}_{\mathcal{S}}\left(f'\right)$ is at most δ. Thus, for any set $S \subseteq \mathcal{P}$ we have

$$\begin{aligned}
\Pr_{y\leftarrow\mathcal{E}^{\epsilon}_{\mathcal{S}}(f)}\left[y\in S\right] &\leq \Pr_{y\leftarrow\mathcal{E}^{\epsilon}(f)}\left[y\in S\right] + \delta \\
&\leq e^{\epsilon}\cdot \Pr_{y\leftarrow\mathcal{E}^{\epsilon}(f')}\left[y\in S\right] + \delta \\
&\leq e^{\epsilon}\cdot \Pr_{y\leftarrow\mathcal{E}^{\epsilon}_{\mathcal{S}}(f')}\left[y\in S\right] + e^{\epsilon}\delta + \delta \ .
\end{aligned}$$

□

Reminder of Fact 2. *The sum $\sum_{\tilde{f}\in\mathcal{P}} \exp\left(-\epsilon\cdot\mathbf{dist}\left(f,\tilde{f}\right)\right)$ converges.*

Proof of Fact 2. Observe that for *any* partition $\tilde{f} \in \mathcal{P}\left(N+2d\right)$ we must have $\mathbf{dist}\left(f,\tilde{f}\right) \geq d$. Applying Pribitkin's [10] bound on $\left|\mathcal{P}\left(N\right)\right|$ we get

$$\begin{aligned}
&\textstyle\sum_{\tilde{f}\in\mathcal{P}} \exp\left(-\epsilon\cdot\mathbf{dist}\left(f,\tilde{f}\right)\right) \\
&\leq \sum_{d\in\mathbb{Z}}\sum_{\tilde{f}\in\mathcal{P}(N+d)} \exp\left(-\epsilon\cdot\mathbf{dist}\left(f,\tilde{f}\right)\right) \\
&\leq \sum_{d\in\mathbb{Z}}\sum_{\tilde{f}\in\mathcal{P}(N+d)} \exp\left(-\epsilon\cdot\frac{d}{2}\right) \\
&\leq \sum_{d\in\mathbb{Z}} \exp\left(\pi\frac{\sqrt{2N+2d}}{\sqrt{3}} - \frac{\epsilon\cdot d}{2}\right) \ ,
\end{aligned}$$

where it is clear that the last sum converges. □

Reminder of Lemma 2. *For all (i,U) such that $i \leq N+d$ and $L_i \leq U \leq U_i$ we have*

$$w_{i,U,\epsilon} = \exp\left(\frac{-\epsilon\left|U-f_i\right|}{2}\right)\cdot \sum_{U'=L_{i+1}}^{\min\{U,U_{i+1}\}} w_{i+1,U',\epsilon} \ .$$

Proof of Lemma 2.

$$\begin{aligned}
w_{i,U,\epsilon} &= \sum_{n\geq 0}\sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^i}\\ x_1=U}} \exp\left(-\epsilon\times\mathbf{dist}\left(f^i,x\right)\right) \\
&= \sum_{n\geq 0}\sum_{U'=0}^{U}\sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^i}\\ x_1=U,x_2=U'}} \exp\left(-\epsilon\times\mathbf{dist}\left(f^i,x\right)\right) \\
&= \sum_{n\geq 0}\sum_{U'=0}^{U}\sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^i}\\ x_1=U,x_2=U'}} e^{-\epsilon\times\mathbf{dist}\left(f^{i+1},x^2\right)-\epsilon\times|f_i-U|/2} \\
&= e^{-\epsilon|f_i-U|/2}\sum_{U'=0}^{U}\sum_{n\geq 0}\sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^i}\\ x_1=U,x_2=U'}} e^{-\epsilon\times\mathbf{dist}\left(f^{i+1},x^2\right)} \\
&= e^{-\epsilon|f_i-U|/2}\sum_{U'=0}^{U}\sum_{n\geq 0}\sum_{\substack{x\in\mathcal{P}(n)\setminus S^d_{f^{i+1}}\\ x_1=U'}} e^{-\epsilon\times\mathbf{dist}\left(f^{i+1},x\right)} \\
&= e^{-\epsilon|f_i-U|/2}\sum_{U'=L_{i+1}}^{\min\{U,U_{i+1}\}} w_{i+1,U,\epsilon} \ .
\end{aligned}$$

□