

# Towards a Rigorous Statistical Analysis of Empirical Password Datasets

Jeremiah Blocki\*  
jblocki@purdue.edu  
Purdue University

Peiyuan Liu\*  
liu2039@purdue.edu  
Purdue University

## ABSTRACT

In this paper we consider the following problem: given  $N$  independent samples  $S = (s_1, \dots, s_N)$  from an unknown distribution  $\mathcal{P}$  over passwords  $pwd_1, pwd_2, \dots$  can we generate high confidence upper/lower bounds on the guessing curve  $\lambda_G \triangleq \sum_{i=1}^G p_i$  where  $p_i = \Pr[pwd_i]$  and the passwords are ordered such that  $p_i \geq p_{i+1}$ . Intuitively,  $\lambda_G$  represents the probability that an attacker who knows the distribution  $\mathcal{P}$  can guess a random password  $pwd \leftarrow \mathcal{P}$  within  $G$  guesses. Understanding how the guessing curve  $\lambda_G$  increases with the number of guesses  $G$  can help quantify the damage of a password cracking attack and inform password policies. Despite an abundance of large (breached) password datasets upper/lower bounding  $\lambda_G$  remains a challenging problem. We introduce several statistical techniques to derive tighter upper/lower bounds on the guessing curve  $\lambda_G$  which hold with high confidence. We apply our techniques to analyze 9 large password datasets finding that our new lower bounds dramatically improve upon prior work. Our empirical analysis shows that even state-of-the-art password cracking models are significantly less guess efficient than an attacker who knows the distribution. When  $G$  is not too large we find that our upper/lower bounds on  $\lambda_G$  are both very close to the empirical distribution  $\hat{\lambda}_G$  which justifies the use of the empirical distribution in settings where the guessing number  $G$  is not too large i.e.,  $G \ll N$  closely approximates  $\lambda_G$ . The analysis also highlights regions of the curve where we can, with high confidence, conclude that the empirical distribution significantly overestimates the real guessing curve  $\lambda_G$ . Our new statistical techniques yield substantially tighter upper/lower bounds on  $\lambda_G$  though there are still regions of the curve where the best upper/lower bounds diverge significantly.

## KEYWORDS

Password Distributions, Password Cracking, Upper Bounds, Lower Bounds, Statistical Analysis

## 1 INTRODUCTION

Understanding and characterizing the distribution over user chosen passwords is a key challenge in the field of cybersecurity with important implications towards the development of robust security policies. How many passwords could an offline attacker crack with  $G$  guesses per account? How expensive does a password hashing algorithm need to be to deter an offline brute-force attacker? Can we strengthen the password distribution by imposing restrictions, e.g., requiring passwords to include numbers and/or capital letters, on the passwords that users can pick? If so are the security gains significant enough to justify the usability costs?

Supposing that  $p_i$  denotes the probability of the  $i$ th most likely password in our (unknown) password distribution  $\mathcal{P}$  we would like to estimate  $\lambda_G \triangleq \sum_{i=1}^G p_i$  the probability that an attacker can crack a random user's password within  $G$  guesses. Unfortunately, there is no known succinct description of the user password distribution  $\mathcal{P}$  which can be used to compute  $\lambda_G$  directly. However, in the past decade breaches at organizations such as RockYou, LinkedIn and 000webhost have resulted in the release of many large password datasets<sup>1</sup>. Viewing each of these datasets  $S$  as  $N = |S|$  independent samples from an unknown password distribution  $\mathcal{P}$  we would like to use the sampled datasets  $S$  to characterize the guessing curve  $\lambda_G$ .

Given a large password dataset one common method to estimate  $\lambda_G$  is to use the empirical distribution e.g., as in [3, 7–9, 13, 19, 20]. In particular, given a dataset  $S$  of  $N$  user passwords we can let  $\hat{f}_i$  denote the frequency of the  $i$ th most common password  $pwd_i$  in the sample  $S$  and define a distribution in which  $\Pr[pwd_i] = \hat{p}_i \triangleq \hat{f}_i/N$ . We can then estimate that an offline attacker making  $G$  guesses per account will crack each password with probability  $\hat{\lambda}_G = \sum_{i=1}^G \hat{p}_i$ . This approach often requires very large samples  $N \gg G$  before the estimate  $\hat{\lambda}_G$  stabilizes and (most likely) yields highly inaccurate predictions when i.e.,  $G > N$  if  $\hat{\lambda}_G = 1$  regardless of the sample  $S$  of the underlying distribution  $\mathcal{P}$ .

Another approach to estimate  $\lambda_G$  is to extract guessing curves from a state-of-the-art password cracking model  $M$ . The password cracking model have been developed using Neural Networks [45], Probabilistic Context Free Grammars [55, 59], Markov Models [18, 25, 41, 47] and sophisticated rule lists [40] used in password cracking software such as John the Ripper (JtR) [33] and Hashcat [32]. Given a sample  $S = \{s_1, \dots, s_n\}$  and a password cracking model  $M$  one can assign guessing numbers  $g_{i,M}$  to each of the passwords in the sample  $S$  indicating that an attacker using model  $M$  will require  $g_{i,M}$  guesses to crack the password  $s_i$ . While the guessing numbers  $g_{i,M}$  can be computed exactly this can be a time consuming process so efficient Monte Carlo algorithms [24] are often used to approximate  $g_{i,M}$ . Once we have the guessing numbers we can approximate  $\lambda_G$  with the heuristic estimate  $\tilde{\lambda}_G = \frac{\text{Count}(S, G)}{N}$  where  $\text{Count}(S, G) = |\{i : g_{i,M} \leq G\}|$  counts the number of passwords in  $S$  with at most  $G$  guesses. One benefit of this approach is that we typically do not require large samples sizes  $N = |S|$  to obtain a stable characterization of the guessing curve  $\tilde{\lambda}_G$ . Unfortunately, neither estimate  $\tilde{\lambda}_G$  or  $\hat{\lambda}_G$  is guaranteed to approximate the real guessing curve  $\lambda_G$  and the two estimates  $\hat{\lambda}_G$  and  $\tilde{\lambda}_G$  are often wildly divergent e.g., see [40, Figure 3]. For example, with the 000webhost dataset for  $G = 10592935$  we have  $\tilde{\lambda}_G = 0.15504$  even

<sup>1</sup>Passwords in the some breaches (e.g., RockYou) were stored in plaintext. In other breaches like LinkedIn the password hashes were not salted making it much easier for an offline attacker to crack most of the passwords.

\*Both authors contributed equally to this research and are ordered alphabetically.

for the most sophisticated password models while for our empirical distribution we have  $\hat{\lambda}_G = 1$ . This could be because  $\hat{\lambda}_G$  overestimates  $\lambda_G$  and/or because  $\hat{\lambda}_G$  underestimates  $\lambda_G$ . As we observed previously we always have  $\hat{\lambda}_N = 1$  even if we draw our sample  $S$  from the uniform distribution over 100-bit passwords. In this case given a large sample  $N = 2^{33}$  (larger than the global population) we would still have  $\lambda_N = N2^{-56} = 2^{-67} \ll \hat{\lambda}_N = 1$ . On the other hand it is also possible that  $\hat{\lambda}_G$  greatly underestimate the probability that an attacker can crack each password.

This divergence poses a significant challenge for a defender who ideally would like to set security policies based on  $\lambda_G$ . If the defender bases his policies on the guessing curve  $\hat{\lambda}_G$  derived from current state of the art cracking models it is possible that the guessing curve (and the policy recommendations) will change if a more sophisticated model is developed tomorrow. On the other hand  $\hat{\lambda}_G$  may be an overly pessimistic estimate especially if  $G$  is large and our password dataset is small.

Despite their shortcomings and many attempts to replace them passwords remain entrenched as the dominant form of authentication on the internet and are likely to play a critical role in the foreseeable future because they are easy to use, easy to deploy and users are already familiar with them [14]. Thus, characterizing the distribution  $\lambda_G$  of user chosen passwords will continue to be an important challenge in the field of cybersecurity.

In this paper we address the following questions: Can we confidently derive accurate upper and lower bounds on  $\lambda_G$ ? When (if ever) can we use the empirical distribution  $\hat{\lambda}_G$  to accurately model the real distribution? How guess efficient are state-of-the-art password cracking models?

## 1.1 Our Contributions

We present two methods to upper bound  $\lambda_G$  and four methods to lower bound  $\lambda_G$ . All of our bounds can be shown to hold with high probability over the randomly selected password samples.

**Upper Bounds.** We first show that  $\lambda_G$  is (with high probability) upper bounded by the empirical estimate  $\hat{\lambda}_G$  i.e., with high probability we have  $\lambda_G \leq \hat{\lambda}_G + \epsilon$  for any constant  $\epsilon > 0$ . Empirical analysis shows that this upper bound is often tight when  $G$  is moderately large i.e., the upper/lower bounds are very close. However, the upper bound becomes less and less tight as  $G$  increases and plateaus at  $\hat{\lambda}_G = 1$  once  $G \geq \text{Distinct}(S)$  exceeds the number of distinct passwords in our sample  $S$  we will have — even if the real value  $\lambda_G$  is small. Thus, we develop a second approach to upper bound  $\lambda_G$  when  $G$  is large using linear programming (LP). Our LP approach adapts techniques of Valiant and Valiant [54] for estimating properties of a distribution with a sub-linear number of samples. Intuitively, our LP searches for a distribution which maximizes  $\lambda_G$  subject to the constraint that the distribution is “sufficiently consistent” with our sample  $S$ . We show that all of the consistency constraints that we add to our linear programming must hold with high probability. Thus, the upper bound we derive will hold with high probability. Empirical analysis shows that our LP upper bounds allows us to push past the  $G = \text{Distinct}(S)$  barrier and generate tighter upper bounds.

**Lower Bounds.** We first give a simple algorithm to derive high confidence lower bounds on  $\lambda_G$  inspired by Good-Turing frequency estimation. The algorithm randomly partitions our sample  $S$  into two components  $D_1$  and  $D_2$  of size  $N - d$  and  $d$  respectively, identifies the set  $T(D_1, G)$  containing the  $G$  most common passwords in  $D_1$ , and counts the number of passwords in  $D_2$  which appear in the set  $T(D_1, G)$ . Empirical analysis shows that for smaller guessing numbers (e.g.,  $G \leq 10^6$ ) our upper/lower bounds are very close indicating that the empirical guessing curve  $\hat{\lambda}_G$  closely approximates the real (unknown) curve  $\lambda_G$ . The above approach is limited in the sense that the lower bound will plateau at  $\approx 1 - \frac{\text{Unique}(S)}{N}$  once  $G > \text{Distinct}(S)$ . Here,  $\text{Unique}(S)$  counts the number of passwords which appear exactly once in our sample  $S$ . We provide two techniques to push past this barrier and derive stronger lower bounds when  $G$  is large. We can adapt the LP described earlier to search for a distribution that *minimizes*  $\lambda_G$  subject to the constraint that the distribution is “sufficiently consistent” with our sample  $S$ . Empirical analysis confirms that this approach generates tighter lower bounds when  $G > \text{Distinct}(S)$  allowing us to push past the  $1 - \frac{\text{Unique}(S)}{N}$  barrier. Both of these lower bounds significantly outperform a prior lower bound of Blocki et al. [9] e.g., for RockYou dataset when  $G = 1.3 \times 10^8$  two of our lower bounds reach 62.64% and 72.70% respectively while the prior lower bound is only 53.95%. Finally, we show how a password cracking model  $M$  can be used to extend the first lower bound. As before we partition  $S$  into two components  $D_1$  and  $D_2$  and if  $G > \text{Distinct}(D_1)$  then we count the number of passwords in  $D_2$  which *either* appear in  $D_1$  or that appear in the top  $G - \text{Distinct}(D_1)$  guesses generated by our model  $M$ . Empirical analysis shows that this combined bound can improve on our prior lower bounds when  $G$  is very large.

**Empirical Analysis.** We apply our theoretical upper and lower bounds to analyze nine password datasets (i.e. Yahoo!, RockYou, LinkedIn, 000webhost, Neopets, Battlefield Heroes, Brazzers, Clixsense, CSDN), and we compare our bounds with state-of-the-art password cracking techniques such as Markov models [18, 25, 41, 47], Probabilistic Context-free Grammars (PCFG) [55, 59], neural networks [45], John the Ripper (JtR) and Hashcat. Our empirical analysis shows that our upper and lower bounds are very close for smaller values of  $G$ . This provides an answer to our second question. If the upper bound  $\hat{\lambda}_G + \epsilon$  is close to our lower bound on  $\lambda_G$  then  $\hat{\lambda}_G$  is a good approximation of the real guessing curve  $\lambda_G$ . Thus, we can use the empirical distribution  $\hat{\lambda}_G$  to estimate  $\lambda_G$  whenever  $G$  is not too large. By contrast, for some larger values of  $G$  we can demonstrate with high confidence that the empirical guessing curve  $\hat{\lambda}(G)$  significantly overestimates  $\lambda_G$  indicating that the empirical distribution such not be used to approximate the real guessing curve in these settings. We also compare our upper/lower bounds to CDF-Zipf curves fit to the empirical dataset [57]. While CDF-Zipf curve are consistent with our upper/lower bounds for several datasets, we also identify cases where the CDF-Zipf curve overestimates  $\lambda_G$  by *at least* 12%.

We find that our lower bounds on  $\lambda_G$  are often significantly higher than the guessing curves obtained using state-of-the-art password cracking models [24, 40, 45, 59]. For example, an attacker making  $G \approx 8.4$  million guesses per account would crack *at most*

14% of passwords in the 000webhost using any of the state-of-the-art password cracking models we analyzed. By contrast, our high confidence<sup>2</sup> lower bounds show that an attacker who knows the password distribution would crack *at least* 39.16% of 000webhost passwords. Similar observations held for other datasets. This provides compelling statistical evidence even the most sophisticated password cracking models still have a large room for improvement in guess efficiency. There has been a push towards designing moderately expensive password hashing algorithms to discourage offline attacks e.g., see [2, 5, 9, 26, 48]. If password guessing becomes more expensive then attacker will have additional incentive to develop guess efficient models.

## 2 ATTACK MODEL AND NOTATION

**Notation:** We let  $\mathcal{P}$  denote an arbitrary password distribution over passwords  $pwd_1, pwd_2, \dots$  and we let  $p_i$  denote the probability of sampling  $pwd_i$ . We use  $s \leftarrow \mathcal{P}$  to denote a random sample from the distribution and  $S \leftarrow \mathcal{P}^N$  to denote a multiset of  $N$  independent and identically distributed samples from  $\mathcal{P}$ . We will occasionally abuse notation and also use  $\mathcal{P}$  to denote the set of passwords  $\{pwd_1, pwd_2, \dots\}$ . We assume that the passwords are ordered in descending order of probability such that  $p_1 \geq p_2 \geq p_3 \dots$  and in general  $p_i \geq p_{i+1}$ . We use  $S = \{s_1, \dots, s_N\}$  to denote a multiset of  $N$  independent and identically distributed samples from  $\mathcal{P}$ . Note that passwords are sampled with replacement so we could have  $s_i = s_j$  for  $i < j$ . It will be convenient to define  $f_i^S \doteq |\{j : s_j = pwd_i\}|$  as the number of times  $pwd_i$  appears in the sample  $S$  – the superscript  $S$  may be omitted when the sample set  $S$  is clear from context (Note that if  $i < j$  we could still have  $f_i^S < f_j^S$  even though  $p_i > p_j$  since the passwords  $pwd_i$  are ordered by probability *not* by their frequency in the sample  $S$ .) We will also define  $F_i^S \doteq |\{pwd_j : f_j^S = i\}|$  as the number of distinct passwords that appear exactly  $i$  times in  $S$  and denote  $F^S = (F_1^S, F_2^S, \dots, F_N^S)$  as the frequency encoding of our sample  $S$ . Under this notation we have  $\text{Unique}(S) = F_1^S$ ,  $\text{Distinct}(S) = \sum_{i \leq N} F_i^S$  and  $N = \sum_{i \leq N} i \times F_i^S$ .

It will be convenient to let  $L_S$  denote a list of all distinct passwords in  $S$  ordered by frequency  $f_i^S$  – ties can be broken in arbitrary order e.g., lexicographic. We also define the set  $T(S, G)$  which contains the first  $G$  passwords in  $L_S$ . If  $G \geq \text{Distinct}(S)$  then  $T(S, G)$  is simply the set of all distinct passwords in the sample  $S$ .

**Attacker Model:** We consider an attacker who has *perfect knowledge* of the distribution  $\mathcal{P}$  i.e., for each  $i$  the attacker knows  $pwd_i$  and  $p_i$ . However, the attacker is given no additional information about the sample set  $S$  of user passwords beyond that the passwords were sampled iid from  $\mathcal{P}$ . For each sampled user password  $s_i \in S$  the attacker is given  $G$  guesses to crack the password  $s_i$ . For an online attacker  $G$  will typically be small as an authentication server can lock the account after several consecutive incorrect login attempts. By contrast,  $G$  will be much larger for an offline attacker who has stolen the (salted) cryptographic hash of the user’s password can check as many passwords as s/he wants by comparing the (salted) cryptographic hash  $h^i = H(u_i, s_i)$  with the hash

$h_j^i = H(u_i, pwd_j)$  for each  $j \leq G$ . An offline attacker is limited only by the resources s/he is willing to invest cracking and by the cost of repeatedly evaluating the password hash function.

Whether  $G$  is large or small the optimal strategy for a perfect knowledge attacker is always to check the password guesses  $pwd_1, pwd_2, \dots, pwd_G$  in order. We use  $\lambda(S, G) \doteq \sum_{i \leq G} f_i^S / N$  to denote the percentage of passwords in  $S$  cracked by a perfect knowledge attacker making  $G$  guesses per user. Observe that the expectation of  $\lambda(S, G)$  is  $\lambda_G = \mathbb{E}(\lambda(S, G)) = \sum_{i \leq G} p_i$  as the attacker cracks each individual password  $s_i$  if and only if  $s_i = pwd_j$  for some  $j \leq G$ . In the next section we will show that the random variable  $\lambda(S, G)$  is tightly concentrated around its mean  $\lambda_G$  i.e., except with negligible probability we will have  $|\lambda_G - \lambda(S, G)| \leq \epsilon$ . In this sense upper/lower bounding  $\lambda_G$  and  $\lambda(G, S)$  can be seen as (nearly) equivalent problems.

One objection to our attacker model is that we would not expect a real world password cracker to have *perfect* knowledge of the password distribution  $\mathcal{P}$ . While this may be true, we argue that there are many compelling reasons to study the cracking curve  $\lambda_G$  for a perfect knowledge attacker. First, if security policies were derived based on current state-of-the-art cracking models then these policies might need to be updated every time the attacker develops a better model. Arguably, a more conservative approach would be to base security policies and recommendations directly on  $\lambda_G$  when feasible since  $\lambda_G$  can be viewed as an upper bound on the success probability of any password cracking model the attacker might develop. If we have tight upper/lower bounds on  $\lambda_G$  then we can derive stable recommendations for security policies. Second, analyzing  $\lambda_G$  can help us to understand the extent to which current state-of-the-art cracking models could be improved with a better model or with additional training data from new password breaches. It is quite possible that password cracking models continue to improve in the future to the extent that the model generates (essentially) the same guesses as perfect knowledge attacker in (essentially) the same order. Finally, if our attacker is attempting to crack all (most) of the passwords in a sample  $S = \{s_1, \dots, s_n\} \leftarrow \mathcal{P}^N$  then the attacker can keep track of all of the passwords  $L_{prior}$  that have been cracked so far and, even if the model  $M$  is not guess efficient, the performance of the attacker will start to approach  $\lambda_G$  over time<sup>3</sup>.

## 3 THEORETICAL UPPER/LOWER BOUNDS

In this section we introduce several algorithms to generate high confidence upper bounds and lower bounds on  $\lambda_G$  given  $N$  independent and identically distributed (iid) samples  $S = \{s_1, \dots, s_n\}$  from our password distribution  $\mathcal{P}$ . An upper bound  $\text{UB}(G, S)$  (resp.  $\text{LB}(G, S)$ ) derived from the sample  $S$  holds with confidence  $1 - \delta$  if  $\Pr[\lambda_G \geq \text{UB}(G, S)] \leq \delta$  (resp.  $\Pr[\lambda_G \leq \text{LB}(G, S)] \leq \delta$ ) where the randomness is taken over the selection of  $S \leftarrow \mathcal{P}^N$ . Before presenting our results, we first introduce the well-known bounded differences inequality [44] (also called McDiarmid’s inequality), which will be useful in our proofs.

<sup>2</sup>The probability of an errant lower (or upper) bound can be upper bounded by a small constant  $\delta$ . In our experiments we tuned our parameters such that  $\delta \leq 0.01$ .

<sup>3</sup>For example, suppose that password samples  $s_i$  and  $s_j$  are identical. Even if it takes the attacker  $g_{i,M} \gg N$  guesses to crack  $s_i$  the first time, the attacker can guarantee that it takes *at most*  $N$  guess to crack  $s_j$  by first checking if  $s_j \in L_{prior}$ . The attacker can further optimize his attack by sorting the list  $L_{prior}$  based on frequency.

**THEOREM 3.1.** (*Bounded Differences Inequality [44]*) Suppose that  $(X_1, \dots, X_n) \in \Omega$  are independent random variables. Let  $f : \Omega \rightarrow \mathbb{R}$  satisfy the bounded differences property with constants  $c_1, \dots, c_n$ , i.e., for all  $i \in \{1, \dots, n\}$  and all  $x, x' \in \Omega$  that differ only at the  $i$ -th coordinate, the output of the function  $|f(x) - f(x')| \leq c_i$ . Then,

$$\Pr[f(X_1, \dots, X_n) - \mathbb{E}(f(X_1, \dots, X_n)) \geq t] \leq \exp\left(\frac{-2t^2}{\sum_{i=1}^n c_i^2}\right);$$

$$\Pr[f(X_1, \dots, X_n) - \mathbb{E}(f(X_1, \dots, X_n)) \leq -t] \leq \exp\left(\frac{-2t^2}{\sum_{i=1}^n c_i^2}\right).$$

As an immediate application of McDiarmid's inequality we can prove that, except with negligible probability, we have  $|\lambda(S, G) - \lambda_G| \leq \epsilon$  i.e.,  $\lambda(S, G)$  is tightly concentrated on its expectation  $\lambda_G$  when the sample size  $N$  is large enough — see Theorem 3.2. Thus, one strategy to derive high confidence upper/lower bound  $\lambda_G$  is to derive a high confidence upper/lower bound for  $\lambda(S, G)$  and we will immediately obtain a high confidence upper/lower bound for  $\lambda_G$  as a corollary.

**THEOREM 3.2.** For any guessing number  $G \geq 0$  and any  $0 \leq \epsilon \leq 1$  we have:

$$\Pr[\lambda(S, G) \leq \lambda_G + \epsilon] \geq 1 - \exp(-2N\epsilon^2)$$

$$\Pr[\lambda(S, G) \geq \lambda_G - \epsilon] \geq 1 - \exp(-2N\epsilon^2)$$

where the randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

**PROOF.** Recall that the samples  $s_1, s_2, \dots, s_N$  in  $S$  are  $N$  independent random variables sampled from the real password distribution  $\mathcal{P}$ . For any two sample set  $S = \{s_1, \dots, s_i, \dots, s_N\}$  and  $S' = \{s_1, \dots, s_{i'}, \dots, s_N\}$  that only differs on one sample  $s_i$  and  $s_{i'}$ , the output of the function  $|\lambda(S, G) - \lambda(S', G)| \leq 1$ . Therefore, using Theorem 3.1, for any parameter  $0 \leq \epsilon \leq 1$  we have:

$$\begin{aligned} \Pr[\lambda(S, G) \geq \lambda_G + \epsilon_1] &\leq \exp(-2N\epsilon_1^2) \\ \Rightarrow \Pr[\lambda(S, G) \leq \lambda_G + \epsilon_1] &\geq 1 - \exp(-2N\epsilon_1^2) \\ \Pr[\lambda(S, G) \leq \lambda_G - \epsilon_1] &\leq \exp(-2N\epsilon_1^2) \\ \Rightarrow \Pr[\lambda(S, G) \geq \lambda_G - \epsilon_1] &\geq 1 - \exp(-2N\epsilon_1^2) \end{aligned}$$

□

### 3.1 A Lower Bound for $G \geq N$ Derived from Existing Work

Fixing arbitrary parameters  $L \geq 1$  and  $j \geq 2$  Blocki et al. [9] proposed the formula  $f(S, L) = \frac{1}{N} \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}}$  as a lower bound for the expected number of passwords cracked by a rational attacker<sup>4</sup> when the value of a cracked password is at least  $v > NL$ . In Theorem 3.3 we show that the same formula  $f(S, L)$  can be used to derive high confidence lower bounds for

<sup>4</sup>Intuitively, a rational password cracker will continue checking passwords as long as the marginal reward  $p_i v$  of checking the next password  $pwd_i$  exceeds the marginal guessing cost. Assuming the cost to check each password guess is at most 1 then as  $p_i v \geq 1$  the attacker will continue guessing and check the next password  $pwd_i$ .

$\lambda_{NL}$  i.e., fixing  $G = NL$  and a slack parameter  $t$  we show that  $\Pr[\lambda(S, G) \geq f(S, L) - t/N] \geq 1 - \exp\left(\frac{-2t^2}{N^2}\right)$ .

**THEOREM 3.3.** For any  $L \geq 1$ ,  $t_1 \geq 0$  and any integers  $N \geq 1$  and  $j \geq 2$  setting  $G = NL$  we have:

$$\Pr\left[\lambda(S, G) \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right)\right] \geq 1 - \exp\left(\frac{-2t^2}{Nj^2}\right)$$

where the randomness is taken over the sample set  $S \leftarrow \mathcal{P}^N$ .

We leave the proofs of Theorem 3.3 to Appendix A.1 as the analysis closely follows Blocki et al. [9]. We then apply McDiarmid's inequality to argue that the lower bound holds with high probability. As an immediate corollary of Theorem 3.3 we can show that, except with negligible probability, we have  $\lambda_{NL} \geq f(S, L) - t/N - \epsilon$  for any constant  $\epsilon > 0$ .

**COROLLARY 3.4.** Given a sample set  $S \leftarrow \mathcal{P}^N$  with size  $N$ , for any  $L \geq 1$ ,  $t \geq 0$ ,  $0 \leq \epsilon \leq 1$ , any integer  $j \geq 2$ , the expected percentage of passwords cracked by a perfect knowledge attacker making  $G = N \times L$  guesses is bounded as below:

$$\Pr[\lambda_G \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right) - \epsilon] \geq 1 - \delta$$

where  $\delta = \exp\left(\frac{-2t^2}{Nj^2}\right) + \exp(-2N\epsilon^2)$ , and the randomness is taken over the sample set  $S \leftarrow \mathcal{P}^N$ .

While the lower bound is useful in some cases we note that Theorem 3.3 and Corollary 3.4 will apply when the guessing number  $G = NL$  is larger than the sample size  $N$  since  $L \geq 1$ . Furthermore, empirical analysis demonstrates this lower bound is much worse than the other lower bounds we propose.

### 3.2 Empirical Distribution as an Upper Bound

In this section we show that we can upper bound  $\lambda_G$  using the empirical distribution  $\hat{\lambda}_G$ . In particular, we argue that for any sample  $S$  we have  $\hat{\lambda}_G \geq \lambda(S, G)$ . As an immediate corollary we get that  $\Pr[\lambda_G > \hat{\lambda}_G + \epsilon] \leq \delta$  where  $\delta = \exp(-2N\epsilon^2)$ .

**THEOREM 3.5.** For any sample set  $S \leftarrow \mathcal{P}^N$  with size  $N$  and any  $G > 0$  we have  $\lambda(S, G) \leq \frac{1}{N} \sum_{i: pwd_i \in T(S, G)} f_i^S$ .

**PROOF.** Observe that

$$\begin{aligned} \sum_{i: pwd_i \in T(S, G)} f_i^S &= \max_{1 \leq j_1 < j_2 < \dots < j_G \leq |P|} \sum_{i \in \{j_1, j_2, \dots, j_G\}} f_i^S \\ &\geq \sum_{i \leq G} f_i^S = N \times \lambda(S, G) \end{aligned}$$

It follows that  $\lambda(S, G) \leq \frac{1}{N} \sum_{i \in T(S, G)} f_i^S$  as claimed. □

Recall that  $T(S, G)$  is the set of  $G$  most frequent passwords in the sample  $S$ . Applying Theorem 3.2 to Theorem 3.5, we can immediately obtain the upper bound of  $\lambda_G$  as below:

COROLLARY 3.6. For any guessing number  $G \geq 0$  we have

$$\Pr \left[ \lambda_G \leq \frac{1}{N} \sum_{i \in T(S, G)} f_i + \epsilon \right] \geq 1 - \exp(-2N\epsilon^2)$$

where the randomness is taken over the selection of  $S \leftarrow \mathcal{P}^N$ .

### 3.3 An Improved Lower Bound for $G < N$

The lower bound in Section 3.1 only applies  $G \geq N$ . In this section, we introduce a new idea that can generate tighter lower bound for small guessing number  $G < N$ .

Given a parameter  $d$ , we partition  $S$  into two sets  $D_1 = \{s_1, \dots, s_{N-d}\}$  and  $D_2 = \{s_{N-d+1}, \dots, s_N\}$  with size  $N - d$  and  $d$ . Since samples in  $S$  are independently randomly sampled from the password distribution  $\mathcal{P}$ ,  $D_1$  and  $D_2$  can be considered as two independent sample sets from  $\mathcal{P}^{N-d}$  and  $\mathcal{P}^d$  respectively. Let  $h(D_1, D_2, G) = |\{N - d + 1 \leq i \leq N : s_i \in T(D_1, G)\}|$  be the number of samples in  $D_2$  that are also in  $T(D_1, G)$ . Recall that  $T(D_1, G)$  is the set of  $G$  most frequent passwords in  $D_1$ . Adding a slack term  $t/d$  we argue that  $\Pr[\lambda_G \leq \frac{1}{d}(h(D_1, D_2, G) - t)] \leq \delta$  where  $\delta = \exp\left(\frac{-2t^2}{d}\right)$ .

THEOREM 3.7. For any guessing number  $G \geq 1$  and any parameters  $0 < d < N$  and  $t \geq 0$ , we have

$$\Pr[\lambda_G \geq \frac{1}{d}(h(D_1, D_2, G) - t)] \geq 1 - \exp\left(\frac{-2t^2}{d}\right)$$

where the randomness is taken over the samples  $D_1 \leftarrow \mathcal{P}^{N-d}$  and  $D_2 \leftarrow \mathcal{P}^d$ .

PROOF. Fixing any  $D_1$  we have  $\mathbb{E}_{D_2}[h(D_1, D_2, G)] = d \times \sum_{i \in T(D_1, G)} p_i \leq d \times \sum_{i \leq G} p_i = d\lambda_G$  where the expectation is taken over the selection of  $D_2$ . Given an arbitrary dataset  $D = \{s_1, \dots, s_d\} \in \mathcal{P}^d$ , an item  $s \in \mathcal{P}$  and an index  $j \leq d$ , we define  $D^{j,s} = \{s_1, \dots, s_{j-1}, s, s_{j+1}, \dots, s_d\}$  where the randomness is taken over the sample  $S$  of size  $N$ . to be the new dataset obtained by swapping item  $s_j$  for  $s$ . Observe that for all  $D_1 \in \mathcal{P}^{N-d}$  we have

$$\sup_{D_2 \in \mathcal{P}^d, s \in \mathcal{P}, j \leq d} |h(D_1, D_2, G) - h(D_1, D_2^{j,s}, G)| \leq 1.$$

Thus, we can apply the Bounded Differences Inequality (Theorem 3.1) to get:

$$\begin{aligned} \Pr[h(D_1, D_2, G) \leq d \times \sum_{i \leq G} p_i + t] \\ \geq \Pr[h(D_1, D_2, G) \leq d \times \sum_{i \in T(D_1, G)} p_i + t] &\geq 1 - \exp\left(\frac{-2t^2}{d}\right) \\ \Rightarrow \Pr[\lambda_G \geq \frac{1}{d}(h(D_1, D_2, G) - t)] \\ = \Pr[\sum_{i \leq G} p_i \geq \frac{1}{d}(h(D_1, D_2, G) - t)] &\geq 1 - \exp\left(\frac{-2t^2}{d}\right) \end{aligned}$$

The last line follows from the observation that that  $\lambda_G = \sum_{i \leq G} p_i \geq \sum_{i \in T(D_1, G)} p_i$ .  $\square$

When applying Theorem 3.2 we can select  $d \ll N$  and set  $t = \sqrt{(d/2) \ln(1/\delta)}$  to get ensure that  $t/d = o(1)$  is small and our probability of error is at most  $\delta$ . We also that if the samples in the dataset  $S$  have been sorted (e.g., in lexicographic order or in

order of frequency) we can still apply Theorem 3.2 to derive our lower bound after randomly shuffling  $S$ . After shuffling  $S$  we can still view the first  $N - d$  (resp. last  $d$ ) entries as a random sample from  $\mathcal{P}^{N-d}$  (resp.  $\mathcal{P}^d$ ). As a corollary of Theorem 3.2 we can also derive a high confidence lower bound for  $\lambda(S, G)$  — see Corollary D.1 in Appendix D.

The lower bound in Theorem 3.2 will plateau at  $G = \text{Distinct}(S)$  since the set  $T(D_1, G)$  already contains all of the passwords in  $D_1$ . When  $d \ll N$  and  $G = \text{Distinct}(S)$  the lower bound will closely match the Good-Turing estimate  $1 - \frac{\text{Distinct}(S)}{N}$  for the total probability mass of all passwords in the sample  $S$ .

### 3.4 An Extended Lower Bound Using Empirical Attacks

The lower bound from Section 3.3 plateaus when  $G \geq \text{Distinct}(S)$ . Is it possible to derive tighter lower bounds for larger values of  $G$ ? In this section we show that any password cracking model can be used to derive high confidence lower bounds on  $\lambda_G$  and then show how our prior lower bound can be combined with a password cracking model  $M$  to derive tighter bounds.

Let  $M(D_1, G)$  be the set of top  $G$  password guesses output by an attack model  $M$  trained on  $D_1$ . We now follow the same approach as before and partition  $S$  into two sets  $D_1 = \{s_1, \dots, s_{N-d}\}$ ,  $D_2 = \{s_{N-d+1}, \dots, s_N\}$ . Let  $h'_M(D_1, D_2, G)$  be the number of passwords cracked in  $D_2$  by making guesses in  $M(D_1, G)$ . We can prove a generalized lower bound of  $\lambda_G$  — see Theorem 3.8 below.

THEOREM 3.8. For any guessing number  $G > 0$  and any parameters  $0 < d < N$ ,  $t \geq 0$  we have:

$$\Pr[\lambda_G \geq \frac{1}{d}(h'_M(D_1, D_2, G) - t)] \geq 1 - \exp\left(\frac{-2t^2}{d}\right)$$

where the randomness is taken over the sample  $S$  of size  $N$ .

PROOF. Since  $p_1, p_2, \dots, p_i, \dots$  are sorted in decreasing order, we have  $\mathbb{E}(h'_M(D_1, D_2, G)) = d \times \sum_{i: \text{pwd}_i \in M(D_1, G)} p_i \leq d \times \sum_{i \leq G} p_i = d \times \lambda_G$ . Using Theorem 3.1, we have:

$$\begin{aligned} \Pr[h'_M(D_2, G) \leq d \times \lambda_G + t] \\ \geq \Pr[h'_M(D_2, G) \leq d \times \sum_{i: \text{pwd}_i \in M(D_1, G)} p_i + t] &\geq 1 - \exp\left(\frac{-2t^2}{d}\right) \\ \Rightarrow \Pr[\lambda_G \geq \frac{1}{d}(h'_M(D_2, G) - t)] &\geq 1 - \exp\left(\frac{-2t^2}{d}\right) \end{aligned}$$

$\square$

As an immediate corollary of Theorem 3.8 we can also lower bound  $\lambda(S, G)$  — see Corollary D.2 in Appendix D. As a more useful corollary given any model  $M$  we can define a model  $M^*$  such that  $M^*(D_1, G)$  first outputs the top  $G$  passwords in  $D_1$  and then, if  $G' = G - \text{Distinct}(D_1) > 0$ , appends  $M(D_1, G')$  the top  $G'$  passwords from the model  $M$  trained on  $D_1$  excluding passwords already in  $D_1$ . Note that for  $G \leq \text{Distinct}(D_1)$  we have  $h'_{M^*}(D_1, D_2, G) = h(D_1, D_2, G)$  where  $h(D_1, D_2, G)$  counts the number of samples in  $D_2$  that appear in the top  $G$  samples from  $D_1$ . For  $G \geq \text{Distinct}(D_1)$  the function  $h'_{M^*}(D_1, D_2, G)$  counts the number of samples in  $D_2$  that either (1) appear in  $D_1$  or (2) appear in  $M(D_1, G')$ . Thus, at minimum we always have  $h'_{M^*}(D_1, D_2, G) \geq$

$\max\{h(D_1, D_2, G), h'_{M^*}(D_1, D_2, G')\}$  where  $G' = G - \text{Distinct}(D_1)$ . Intuitively, the lower bound will be at least as good as our prior approach from Theorem 3.7 and at least as good as the model  $M$ .

**COROLLARY 3.9.** *Let  $M$  be a password cracking model and let parameters  $G, d > 0, t > 0$  and  $\epsilon > 0$  be given then*

$$\begin{aligned} \Pr[\lambda_G \geq \frac{1}{d}(h'_{M^*}(D_1, D_2, G) - t)] &\geq 1 - \exp\left(\frac{-2t^2}{d}\right) \\ \Pr[\lambda(S, G) \geq \frac{1}{d}(h'_{M^*}(D_1, D_2, G) - t) - \epsilon] \\ &\geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right) \end{aligned}$$

where the randomness is taken over the sample set  $S$  of size  $N$ .

### 3.5 Upper And Lower bounds Using Linear Programming

In the previous section we showed how to use a password cracking model  $M$  to extend our prior sampling based lower bound on  $\lambda_G$  when  $G > \text{Distinct}(S)$ . However, if the password cracking model  $M$  is not guess efficient it is still possible that the lower bound will (temporarily) plateau at  $G = \text{Distinct}(S)$ . In this section we propose a different approach to generate upper and lower bounds using linear programming which is inspired by work of Valiant and Valiant [54]. As we will see in our empirical analysis these LP bounds tend to be tighter when the guessing number is large though the bounds are slightly worse when  $G \ll \text{Distinct}(S)$ .

Intuitively, we derive our upper (resp. lower) bounds by designing a linear program to find a distribution  $\mathcal{P}$  that maximizes (resp. minimizes)  $\lambda_G$  subject to various consistency constraints that uses our sample  $S$  to restrict the set of feasible distributions  $\mathcal{P}$ . One of the first tasks will be to identify useful consistency constraints that should hold with high probability over the selection of  $S$ . This ensures that with high probability (whp) the actual password distribution is in our feasible set.

**3.5.1 An Ideal Linear Programming Task.** We begin by making an idealized assumption about the (unknown) password distribution  $\mathcal{P}$ . We will show later on how this simplifying assumption can be eliminated. For now we will fix a finite probability mesh  $X_l = \{x_1, \dots, x_l\}$  and assume that for all  $i$  we have  $p_i \in X$  i.e., the mesh contains every probability value in our distribution. In this case we can encode the distribution  $\mathcal{P}$  as a histogram  $H = h_1, \dots, h_\ell$  where  $h_i$  denotes the number of items in the support of  $\mathcal{P}$  which occur with probability  $x_i$ . For convenience we will assume that  $x_1 > x_2 > \dots > x_l$ . We view  $h_1, \dots, h_\ell \geq 0$  as variables in our linear program (relaxing the natural constraint that  $h_i$  is an integer). Since this is a distribution we can add the linear constraint  $\sum_i h_i x_i = 1$ .

Given our sample  $S$  of size  $N$  recall that  $F_i^S$  denotes the number of distinct passwords that appear exactly  $i$  times in our sample  $S$ . Observe that if a  $\text{pwd}$  occurs with probability  $p$  in our distribution  $\mathcal{P}$  then  $\text{bpdf}(i, N, p) \doteq \binom{N}{i} p^i (1-p)^{N-i}$  (binomial probability density function) is the probability that the item is sampled exactly  $i$  times. Thus, the expected value of  $F_i^S$  is  $\sum_{j=1}^l h_j \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  is the expected probability mass of all items that were sampled exactly  $i$  times. Adapting ideas from Good-Turing Frequency estimation we can argue that (whp)  $\sum_{j=1}^l h_j \times$

$x_j \times \text{bpdf}(i, N, x_j)$  will be close to  $\frac{(i+1)F_{i+1}^S}{N-i}$ . In particular, Lemma 3.10 shows that, except with probability  $2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ , we will have  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$  where  $\epsilon_{2,i} > 0$  is a small constant. Thus, if we ensure that  $\sum_{i=1}^{i'} 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right) \leq \epsilon$  we can add the constraints from Lemma 3.10 for each  $i \leq i'$  and ensure that, except with probability  $\epsilon$ , that all of these constraints will hold. In this case the original distribution  $\mathcal{P}$  defines a feasible solution to the linear program.

Intuitively, if we search for a distribution that maximizes (resp. minimizes)  $\lambda_G$  we obtain upper bound (resp. lower bound) since the real distribution  $\mathcal{P}$  will be one of the feasible solutions (whp). Given a histogram  $H$  and a guessing number  $G$  we can define  $i(G, H) = \max\{j : \sum_{i=1}^{j-1} h_i \leq G\}$  and  $c(G, H) = G - \sum_{i=1}^{i(G, H)-1} h_i$ . Observe that if the attacker attempts  $G$  password guesses they will succeed with probability  $c(G, H)x_{i(G, H)} + \sum_{i=1}^{i(G, H)-1} x_i h_i$  i.e., for  $i < i(G, H)$  they will check all  $h_i$  passwords with probability  $x_i$  and for  $i = i(G, H)$  they can only check  $c(G, H) < h_{i(G, H)}$  of the passwords which occur with probability  $x_{i(G, H)}$ . We would like to find  $h_1, \dots, h_l$  to maximize (or minimize)  $c(G, H)x_{i(G, H)} + \sum_{i=1}^{i(G, H)-1} x_i h_i$ . However, the optimization goal is not quite linear due to the dependence on  $i(G, H)$ .

To solve this problem we introduce another parameter  $\text{idx}$  which represents a guess for the value of  $i(G, H)$  and we define a separate linear program for each possible guess  $1 \leq \text{idx} \leq l$ . Fixing  $\text{idx}$  we introduce a new variable  $c$  which intuitively corresponds to  $c(G, H)$  and we add the constraints that  $0 \leq c \leq h_{\text{idx}}$  and that  $c = G - \sum_{i=1}^{\text{idx}-1} h_i$ . Now for each fixed  $\text{idx}$  the objective  $c x_{\text{idx}} + \sum_{i=1}^{\text{idx}-1} x_i h_i$  is linear and can be maximized/minimized. Letting  $y_{\text{idx}}^*$  denote the value of optimal solution to our LP with the parameter  $\text{idx}$  we can combine these solutions to get our final upper/lower bound i.e.,  $y^* = \max\{y_{\text{idx}}^* : \text{idx} \leq l\}$  (resp.  $y^* = \min\{y_{\text{idx}}^* : \text{idx} \leq l\}$ ) when computing our upper bound (resp. lower bound) where each  $y_{\text{idx}}^*$  is the value we obtain when maximizing (resp. minimizing) the corresponding LP.

Our linear program  $\text{LP1}(G, b, X_l, F^S, \text{idx}, i', \epsilon_2)$  is shown below (Linear Programming Task 1). The inputs include the guessing budget  $G$ , the probability mesh  $X_l = \{x_1, \dots, x_l\}$ , the set  $F^S = \{F_1^S, \dots, F_N^S\}$ , a bit  $b \in \{-1, 1\}$  which indicates whether we are looking for an upper or lower bound, a guess  $\text{idx}$  for  $i(G, H)$  and parameters  $i'$  and  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\}$  related to the consistency constraints.

The following lemma indicates that constraint (2) hold with high probability over the selection of  $S \leftarrow \mathcal{P}^N$  when we select the parameters  $i'$  and  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\}$  properly.

**LEMMA 3.10.** *For any  $i \geq 0$  and  $0 \leq \epsilon_{2,i} \leq 1$ , we have  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \leq \sum_j h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$  with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$  where the probability is taken over the selection of our sample  $S \leftarrow \mathcal{P}^N$ .*

PROOF. We first prove that  $\sum_j h_j \times x_j \times \text{bpdf}(i, N, x_j)$  can be bounded using  $\frac{i+1}{N-i} \mathbb{E}(F_{i+1}^S)$  as below:

$$\begin{aligned}
& \sum_j h_j \times x_j \times \text{bpdf}(i, N, x_j) \\
&= \sum_j h_j \times x_j \times \frac{N!}{(N-i)!i!} x_j^i (1-x_j)^{N-i} \\
&= \sum_j h_j \times \frac{i+1}{N-i} \times (1-x_j) \times \frac{N!}{(N-i-1)!(i+1)!} x_j^{i+1} (1-x_j)^{N-i-1} \\
&= \frac{i+1}{N-i} \sum_j h_j \times (1-x_j) \times \text{bpdf}(i+1, N, x_j) \\
&= \frac{i+1}{N-i} \mathbb{E}(F_{i+1}^S) - \frac{i+1}{N-i} \sum_j h_j \times x_j \times \text{bpdf}(i+1, N, x_j)
\end{aligned}$$

Since  $0 \leq \frac{i+1}{N-i} \sum_j h_j \times x_j \times \text{bpdf}(i+1, N, x_j) \leq \frac{i+1}{N-i}$ , we have  $\frac{i+1}{N-i} \mathbb{E}(F_{i+1}^S) - \frac{i+1}{N-i} \leq \sum_j h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{i+1}{N-i} \mathbb{E}(F_{i+1}^S)$ .

**Linear Programming Task 1:**  $\text{LP1}(G, b, X_l, F^S, idx, i', \epsilon_2)$   
**Input Parameters:**  $G, b, X_l = \{x_1, \dots, x_l\}, F^S = \{F_1^S, \dots, F_N^S\}, idx, i', \epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\}$   
**Variables:**  $h_1, \dots, h_l, c$   
**Objective:**  $\min \left( b \times (\sum_{j < idx} h_j \times x_j + c \times x_{idx}) \right)$   
**Constraints:**  
(1)  $\sum_{j < idx} h_j + c = G$   
(2)  $\forall 0 \leq i \leq i', \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$   
(3)  $\sum_{j=1}^l h_j \times x_j = 1$   
(4)  $0 \leq c \leq h_{idx}$

Next we will prove that when  $i$  is small,  $F_{i+1}^S$  is highly concentrated on  $\mathbb{E}(F_{i+1}^S)$ . We define  $Y_1, \dots, Y_N$  to be  $N$  independent password sample random variables, and consider  $F_{i+1}^S = h(Y_1, \dots, Y_N)$  to be a function that outputs the number of distinct passwords that appear exact  $i+1$  times among all  $N$  samples in the sample set  $S$ . Then we have  $\sup_{y_1, \dots, y_i, \dots, y_N, y'_i} |h(y_1, \dots, y_i, \dots, y_N) - h(y_1, \dots, y'_i, \dots, y_N)| \leq 1$ .

Using Theorem 3.1, for any real value of  $t_1$  we have:

$$\begin{aligned}
\Pr[h(Y_1, \dots, Y_N) \geq \mathbb{E}(F_{i+1}^S) - t_1] &\geq 1 - \exp\left(-\frac{2t_1^2}{N}\right) \\
\Pr[h(Y_1, \dots, Y_N) \leq \mathbb{E}(F_{i+1}^S) + t_1] &\geq 1 - \exp\left(-\frac{2t_1^2}{N}\right)
\end{aligned}$$

Denote  $t_1 = \frac{N-i}{i+1} \epsilon_{2,i}$ . Since  $h(Y_1, \dots, Y_N) = F_{i+1}^S$  we have  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \leq \sum_j h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$  with probability at least  $1 - 2 \times \exp\left(-\frac{2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ .  $\square$

When the sample size  $N$  is large enough and  $i$  is small, the bounds we prove in the lemma above hold with high probability by selecting proper value of  $\epsilon_{2,i}$ .

Using Lemma 3.10 and conclusions we present above, we can derive the high-confidence upper and lower bounds of  $\lambda_G$  as below:

**THEOREM 3.11.** *Given a probability distribution  $\mathcal{P}$  which is consistent with a finite mesh  $X_l = \{x_1, \dots, x_l\}$  for any  $G \geq 0$ , any  $i' \geq 0$ , any  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}$ , we have:*

$$\begin{aligned}
\Pr\left[\lambda_G \geq \min_{1 \leq idx \leq l} \text{LP1}(X_l, F^S, idx, G, 1, i', \epsilon_2)\right] &\geq 1 - \delta \\
\Pr\left[\lambda_G \leq \max_{1 \leq idx \leq l} |\text{LP1}(X_l, F^S, idx, G, -1, i', \epsilon_2)|\right] &\geq 1 - \delta
\end{aligned}$$

where  $\delta = 2 \times \sum_{0 \leq i \leq i'} \exp\left(-\frac{2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$  and the randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

As a corollary of Theorem 3.11 we can also bound  $\lambda(S, G)$  — see Corollary D.3 in Appendix D.

**3.5.2 Intermediate Step: Linear Programming with Countably Infinite Probability Mesh.** Our prior linear programming based lower bound relied on an idealized assumption that the real distribution  $\mathcal{P}$  is consistent with a finite probability mesh  $X_l = \{x_1, \dots, x_l\}$  i.e., for all  $i$  the probability  $p_i \in X_l$  of password  $\text{pwd}_i$  lies in the mesh. In this section we show how to extend the prior approach when  $\mathcal{P}$  is consistent with an infinite mesh  $X = \{x_1, x_2, \dots, x_l, x_{l+1}, \dots\}$  with  $x_1 > x_2 > \dots$  i.e., for all  $i$  the probability  $p_i \in X$  of password  $\text{pwd}_i$  lies in the mesh  $X$ . In the next assumption we will show how to remove the assumption that  $\mathcal{P}$  is perfectly consistent with  $X$  as long as the mesh  $X$  is sufficiently fine-grained. For every integer  $l$  we can define the mesh  $X_l = \{x_1, \dots, x_l\}$  which contains the top  $l$  values in  $X$ . Suppose that  $H = (h_1, h_2, \dots)$  is a histogram encoding of  $\mathcal{P}$  and let  $p = \sum_{i \geq l+1} x_i h_i$  be the total probability mass of all passwords in  $\mathcal{P}$  with probability smaller than  $x_l$ . Observe that we have  $p + \sum_{i=1}^l x_i h_i = 1$ . Our key idea to ensure that our linear programs remain finite is to introduce a new variable  $p$  and eliminate the variable  $h_j$  for each  $j > l$ .

As before the expected probability mass of all items that don't appear in our sample  $S$  (i.e., items that appear with frequency  $i = 0$ ) is  $\sum_{j=1}^\infty x_j h_j \text{bpdf}(0, N, x_j)$  where  $p \geq \sum_{j=l+1}^\infty x_j h_j \text{bpdf}(0, N, x_j) \geq \sum_{j=l+1}^\infty x_j h_j \text{bpdf}(0, N, x_l) = p \times \text{bpdf}(0, N, x_l)$ . Similarly, when  $i > 0$  the expected probability mass of all items that appear in  $S$  with frequency  $i = 1$  is  $\sum_{j=1}^\infty x_j h_j \text{bpdf}(i, N, x_j)$ . Assuming that  $x_l < 1/N$  we have  $0 \leq \sum_{j=l+1}^\infty x_j h_j \text{bpdf}(i, N, x_j) \leq \sum_{j=l+1}^\infty x_j h_j \text{bpdf}(i, N, x_l) = p \times \text{bpdf}(i, N, x_l)$ . We can apply the above observations to modify the second constraint(s) from LP1. Our updated linear program (LP1a) in shown in Linear Programming Task 1a.

Lemma 3.12 shows that with high probability over the selection of  $S$  each of the constraints in LP1a will be consistent with the distribution  $\mathcal{P}$  provided that  $x_l < 1/N$  and  $\mathcal{P}$  is  $l$ -partially consistent with the mesh  $X$ . We say that  $\mathcal{P}$  is  $l$ -partially consistent with the mesh  $X$  if for all passwords  $\text{pwd}_i$  in the support of  $\mathcal{P}$  with corresponding probability  $p_i$  we either have (1)  $p_i \in X_l$  or (2)  $p_i < x_l$ . Intuitively, Lemma 3.12 follows by upper/lower bounding  $\sum_{j=l+1}^\infty x_j h_j \text{bpdf}(i, N, x_j)$  and then applying our prior bounds from Lemma 3.10. The full proof can be found in Appendix A.2.

**LEMMA 3.12.** *Fix any  $i \geq 0$  and  $0 \leq \epsilon_{2,i} \leq 1$  and assume that  $x_l < \frac{1}{N}$  and that  $\mathcal{P}$  is  $l$ -partially consistent with our mesh  $X$ . If  $i = 0$  let  $W_0$  be an indicator random variable which is 1 if and only if  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^l h_j x_j \cdot \text{bpdf}(i, N, x_j)$  and*

$\sum_{j=1}^l h_j x_j \cdot \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l)$ . Similarly, if  $i > 0$  let  $W_i$  be an indicator random variable which is 1 if and only if  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \epsilon_{2,i}$ . Then the constraints hold with probability

$$\Pr[W_i = 1] \geq 1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$$

where the randomness is taken over the selection of  $S \leftarrow \mathcal{P}^N$ .

**Linear Programming Task 1a:** LP1a( $G, b, X_l, F^S, idx, i', \epsilon_2$ )  
**Input Parameters:**  $G, b, X_l = \{x_1, \dots, x_l\}, F^S = \{F_1^S, \dots, F_N^S\}, idx, i', \epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,l'}\}$   
**Variables:**  $h_1, \dots, h_l, c, p$   
**Objective:**  $\min \left( b \times (\sum_{j < idx} h_j \times x_j + c \times x_{idx}) \right)$   
**Constraints:**  
 (1)  $\sum_{j < idx} h_j + c = G$   
 (2)  $\forall 0 \leq i \leq i'$ :  
     (a) for  $i = 0, \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l)$   
     (b) for  $1 \leq i \leq i', \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$   
 (3)  $\sum_{j=1}^l h_j \times x_j = 1 - p$   
 (4)  $0 \leq c \leq h_{idx}$   
**(Note:** we consider  $idx = 1, 2, \dots, l+1$ . When  $idx = l+1$ , we define  $h_{l+1} = G; x_{l+1} = 0$  for  $b = 1; x_{l+1} = x_l$  for  $b = -1$ .)

Similar to Theorem 3.11, using Lemma 3.12, we can derive the high-confidence upper and lower bounds of  $\lambda_G$  as below:

**THEOREM 3.13.** Suppose that the probability distribution  $\mathcal{P}$  is  $l$ -partially consistent with a mesh  $X = \{x_1, x_2, \dots\}$  then for any  $G \geq 0$ , any  $i' \geq 0$ , any  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}$ , we have

$$\Pr \left[ \lambda_G \geq \min_{1 \leq idx \leq l} \text{LP1a}(X_l, F^S, idx, G, 1, i', \epsilon_2) \right] \geq 1 - \delta$$

$$\Pr \left[ \lambda_G \leq \max_{1 \leq idx \leq l} |\text{LP1a}(X_l, F^S, idx, G, -1, i', \epsilon_2)| \right] \geq 1 - \delta$$

where each inequality holds with probability at least  $\delta = 1 - 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . The randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

As an immediate corollary of Theorem 3.13 and Theorem 3.2 we can also bound  $\lambda(S, G)$  — see Corollary D.4 in Appendix D.

**3.5.3 Final LP: Eliminating Ideal Mesh Assumptions.** In this section, we present our final linear program to bound  $\lambda_G$  without making any idealized assumptions about the distribution  $\mathcal{P}$  fitting a mesh  $X$ . Following [54], we fix a particular mesh  $X_{l,q} = \{x_1, \dots, x_l\}$  where where  $x_l = \frac{1}{10^4 N}$ ,  $l = \lfloor \frac{\ln(\frac{1}{x_l})}{\ln q} \rfloor + 1$  and for each  $1 \leq i < l$  we have  $x_i = q \cdot x_{i+1} = q^{l-i} x_l$ . Here,  $q > 1$  is a parameter that determines how fine-grained our mesh values are.

We also consider an ideal mesh  $X^r = \{x_1^r, x_2^r, \dots\} = \{\Pr[pwd] : pwd \in \mathcal{P}\}$  for the real distribution  $\mathcal{P}$  along with a corresponding histogram  $H^r = \{h_1^r, h_2^r, \dots\}$  where  $h_i^r \triangleq |\{pwd \in \mathcal{P} : \Pr[pwd] = x_i^r\}|$ . Given a guessing number  $G > 0$  we have  $\lambda_G = X_{i(G,H)}^r c(G, H) + \sum_{j=1}^{i(G,H)-1} x_i^r h_i^r$  where we define  $i(G, H) = \max\{j : \sum_{i=1}^{j-1} h_i \leq G\}$  and  $c(G, H) = G - \sum_{i=1}^{i(G,H)-1} h_i$  as before. Of course the ideal mesh  $X^r$  and the associated histogram  $H^r$  are unknown so we cannot simply evaluate the formula above. However, we can still use the ideal mesh  $X^r$  in our analysis.

Intuitively, we define a procedure  $\text{Round}(X^r, H^r, X)$  which transforms  $\mathcal{P}$  into a similar distribution  $\mathcal{P}'$  such that  $\mathcal{P}'$  is  $l$ -consistent with the mesh  $X_{l,q}$ . We then add slack terms to our linear constraints to ensure that  $\mathcal{P}'$  is consistent with all of the constraints in our linear program. For upper bounds our rounding procedure will ensure that  $\lambda'_G \geq \lambda_G$  so that we can directly use the linear program to upper bound  $\lambda_G$ . Similarly, for lower bounds our rounding procedure will ensure that  $\lambda'_G \leq \lambda_G$  so we can directly use the linear program to lower bound  $\lambda_G$ . The rounding procedures are formally described in Appendix B. Intuitively, to lower bound (resp. upper bound  $\lambda_G$ ) we will first round down (resp. round up) each real probability value  $x_k^r$  from our ideal mesh to the closest mesh value in  $X_{l,q}$  that is smaller (resp. greater) than  $x_k^r$ . We relax the linear constraint that  $p + \sum_{j \leq l} h_j x_j = 1$  to require that  $p + \sum_{j \leq l} h_j x_j \approx 1$ .

The linear program  $\text{LP}_{\text{lower}}$  to lower bound  $\lambda_G$  is shown below. The linear program  $\text{LP}_{\text{upper}}$  to upper bound  $\lambda_G$  is similar to  $\text{LP}_{\text{lower}}$ . We show it in Appendix B.3.

**Linear Programming Task 2:**  
 $\text{LP}_{\text{lower}}(G, X_l, F^S, idx, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})$   
**Input Parameters:**  $G, X_l = \{x_1, \dots, x_l\}, F^S = \{F_1^S, \dots, F_N^S\}, idx, i', \epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\}, \hat{x}_{\epsilon_3} = \{\hat{x}_{\epsilon_{3,0}}, \dots, \hat{x}_{\epsilon_{3,i'}}\}, \epsilon_3 = \{\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1, 0 \leq i \leq i'\}$   
**Variables:**  $h_1, \dots, h_l, c, p$   
**Objective:**  $\min \left( \sum_{j < idx} h_j \times x_j + c \times x_{idx} \right)$   
**Constraints:**  
 (1)  $\sum_{j < idx} h_j + c = G$   
 (2)  $\forall 0 \leq i \leq i'$ :  
     (a) for  $i = 0, \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq (1 + \epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, q x_l) \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}})$   
     (b) for  $1 \leq i \leq i', \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, q x_l) \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq (1 + \epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}})$   
 (3)  $\frac{1-p}{q} \leq \sum_{j=1}^l h_j \times x_j \leq 1 - p$   
 (4)  $0 \leq c \leq h_{idx}$   
**(Note:** we consider  $idx = 1, 2, \dots, l+1$ . When  $idx = l+1$ , we define  $h_{l+1} = G$  and  $x_{l+1} = 0$ .)

Theorem 3.14 shows that the upper/lower bounds we obtain hold with high confidence — see Appendix B for the complete proof.

**THEOREM 3.14.** Given an unknown password distribution  $\mathcal{P}$  for any  $G \geq 0$ , integer  $i' \geq 0$ ,  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}, \hat{x}_{\epsilon_3} =$



$\{\hat{x}_{\epsilon_{3,0}}, \dots, \hat{x}_{\epsilon_{3,i'}}\}$ ,  $\epsilon_3 = \{\epsilon_{3,i} = \frac{1}{q^{i+1}} (\frac{1-\hat{x}_{\epsilon_{3,i}}}{1-q\hat{x}_{\epsilon_{3,i}}})^{N-i} - 1, 0 \leq i \leq i'\}$ , we have:

$$\Pr \left[ \lambda_G \geq \min_{1 \leq idx \leq l+1} \text{LP}(\text{lower}(X_{l,q}, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})) \right] \geq 1 - \delta$$

$$\Pr \left[ \lambda_G \leq \max_{1 \leq idx \leq l+1} \text{LP}(\text{upper}(X_{l,q}, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})) \right] \geq 1 - \delta$$

where  $\delta = 2 \times \sum_{0 \leq i \leq i'} \exp \left( \frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2} \right)$  and the randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

As a corollary of Theorem 3.14 we can lower bound  $\lambda(S, G)$  — see Corollary D.5 in Appendix D.

## 4 EMPIRICAL ANALYSIS

In this section we apply our techniques to upper/lower bound  $\lambda_G$  to analyze several empirical password datasets. We compare these bounds to guessing curves generated by state of the art password cracking models.

### 4.1 Datasets

We use nine empirical password datasets in our analysis: Yahoo!, RockYou, LinkedIn, 000webhost, Neopets, Battlefield Heroes, Brazzers, Cliksense and CSDN — for the last six datasets we use the sanitized versions prepared by Liu et al [40]. Table 1 provides basic information about each dataset.  $N$  represents the total size of the dataset and **# Distinct** represents the number of distinct passwords after eliminating duplicates. Similarly, **# Unique** represents the number of passwords that appear exactly once in  $S$ . In our analysis we view each dataset  $S$  as representing  $N$  i.i.d. samples from an unknown password distribution. Good-Turing frequency estimation tells us that the total probability mass of unseen passwords is approximately  $1 - \sum_{pwd \in S} \Pr[pwd] \approx \frac{\# \text{ unique}}{N}$  which means for  $G = \# \text{ Distinct}$  we have  $\lambda_G \geq \sum_{pwd \in S} \Pr[pwd] \approx \frac{N - \# \text{ unique}}{N}$ . Thus, for Yahoo! (resp. CSDN) we should have  $\lambda_G \geq 0.575$  (resp.  $\lambda_G \geq 0.443$ ).

Two of the password datasets (Yahoo! [8, 13] and LinkedIn [31]) are actually differentially private frequency lists and do not include plaintext passwords. For these datasets we can still apply our techniques to upper/lower bound  $\lambda_G$ , but we cannot compare our bounds with the empirical password cracking models. While the datasets were slightly perturbed to satisfy differential privacy, Blocki et al. [8] showed that the L1 distortion is minimal i.e., the additional error term is  $O(1/\sqrt{N})$ .

**Ethical Considerations.** Many of the password datasets we analyze contain stolen passwords that were subsequently leaked on the internet and the usage of this data raises important ethical considerations. We did not crack any new passwords as part of our analysis and the breached datasets are already publicly available so our usage of the data does not pose any additional risk to users.

### 4.2 Password Cracking Models

We use the empirical attack results in Liu et al [40] to compare with our bounds for 000webhost, Neopets, Battlefield Heroes, Brazzers,

**Table 1: Basic Information of Evaluation Datasets**

Dataset ( $S$ )	# Passwords ( $N$ )	# Distinct	# Unique
Yahoo! [8]	69301337	33895873	29452171
RockYou [23]	32603388	14344391	11884632
LinkedIn [31]	174292189	57431283	21424510
000webhost [28]	15268903	10592935	9006529
Neopets [21]	68345757	27987227	21509860
Battlefield Heroes [56]	541016	416130	373549
Brazzers [22]	925614	587934	491136
Cliksense [29]	2222529	1628577	1455585
CSDN [62]	6428449	4037749	3581824

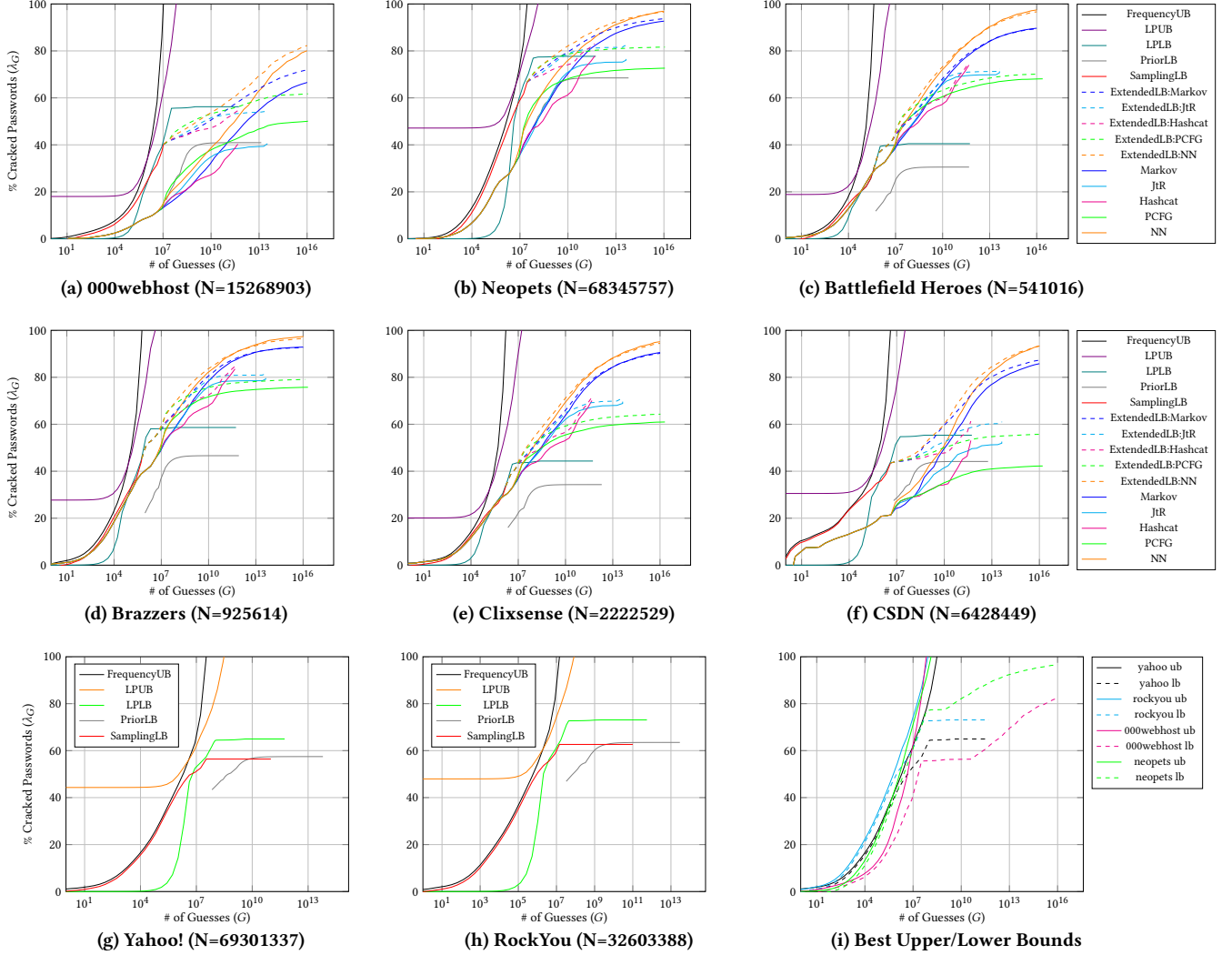
Cliksense and CSDN datasets and to generate an extended lower bound using Corollary 3.9. Liu et al [40] evaluate 10 password cracking models on the six datasets based on Markov model, PCFG, neural networks, Hashcat, and JtR techniques. In our analysis, we focus on the best performing ones of each password cracking technique: the neural network model (denote as **NN**) in Melicher et al [45], the Markov 4-gram model (denote as **Markov**) in Dell’Amico and Filippone [24], the original PCFG model [59] (denote as **PCFG**), the extended JtR [40] (denote as **JtR**), and Hashcat (denote as **Hashcat**) implemented in Liu et al [40]. Some other models (e.g. Markov backoff model [24] and the PCFG model in Komanduri [36]) in Liu et al [40] may outperform **Markov** or **PCFG** in some ranges of  $G$ , but for every value of  $G$  their performance is worse than at least one of the models we selected.

### 4.3 Evaluation

For each password dataset  $S$  we generate upper/lower bounds on  $\lambda_G$  using our results from Section 3 and compare the upper/lower bounds with the guessing curves derived from password cracking models. Our upper/lower bounds and empirical attack results for 000webhost, Neopets, Battlefield Heros, Brazzers, Cliksense, and CSDN, Yahoo!, and RockYou are shown in Figures 1(a)-(h).

In these figures we use FrequencyUB( $S, G$ ) (resp. LPUB( $S, G$ )) to denote the upper bound obtained from Corollary 3.6 (resp. Theorem 3.14) with password dataset  $S$ . Similarly, we use SamplingLB( $S, G$ ) (resp. LPLB) to denote the lower bound obtained by applying Theorem 3.7 (resp. Theorem 3.14). For comparison we also plot PriorLB( $S, G, j$ ) which denotes the lower bound from Corollary 3.4 based on results of Blocki et al. [9] — specifically we set  $\text{PriorLB}(S, G) = \max_{2 \leq j \leq 1000} \text{PriorLB}(S, G, j)$  where  $\text{PriorLB}(S, G, j)$  is the lower bound we obtain after fixing the parameter  $j$  in Corollary 3.4. Two of the lower bounds LBUP and LPLB require us to solve linear programs as a subroutine. We used Gurobi 9.0.1 [30] as our linear programming solver.

For each of the upper/lower bounds there is an error term  $\delta$  which upper bounds the probability that our bound is wrong. The error term  $\delta$  will depend on our choice of parameters. For example, in Corollary 3.4 (resp. Theorem 3.7) the parameters  $t, \epsilon$  (resp.  $t, d$ ) determines  $\delta = \exp(-2t^2/(Nj^2)) + \exp(2N\epsilon^2)$  (resp.  $\delta = \exp(-2t^2/d)$ ). Briefly, we always select parameters such that  $\delta \leq 0.01$ . For example, to generate SamplingLB( $S, G$ ) we set  $d = 2.5 \times 10^4$  and  $t \geq \sqrt{d \ln(1/\delta)/2}$  in Theorem 3.7 i.e., we randomly partition  $S$  into  $D_1 \in \mathcal{P}^{N-d}$  and  $D_2 \in \mathcal{P}^d$  and return our lower



**Figure 1: 000webhost, Neopets, Battlefield Heroes, Brazzers, Clixsense, CSDN, Yahoo! and RockYou Guessing Curves, and Best Bounds**

bound  $(h(D_1, D_2, G) - t)/d$  where  $h(D_1, D_2, G)$  counts the number of passwords in  $D_2$  that are top  $G$  passwords in  $D_1$ . See Appendix E for concrete details on how we specify all of these relevant parameters for other upper/lower bounds.

We compare our upper/lower bounds with empirical password guessing curves derived from state of the art password cracking models. Specifically, we compare with the guessing curves generated by Liu et al [40]. In particular, for each password dataset  $S$  they first subsample 25,000 passwords to obtain a smaller dataset  $D_2$ . Then for each password  $pwd \in D_2$  and password cracking model  $M$  they compute a guessing number  $g_{M,pwd}$  for that password (often using Monte Carlo strength estimation [24]). Finally, for a guessing bound  $G$  we can estimate that the model will crack  $\tilde{\lambda}_{G,M} = |\{pwd \in D_2 : g_{M,pwd} \leq G\}|/|D_2|$  passwords. We can also apply Corollary 3.9 to derive a new lower bound on  $\lambda_G$  by combining the results from our model  $M$  with our sampling based lower bound  $\text{SamplingLB}(S, G)$  — we use  $\text{ExtendedLB}(S, G, M)$  to denote

the extended lower bound and highlight these lower bounds using dotted lines in Figure 1.

#### 4.4 Discussion

Figure 1 shows that when  $G$  is small our upper bound  $\text{FrequencyUB}(G, S)$  and lower bound  $\text{SamplingLB}(G, S)$  are very close to each other. For example, when  $G \leq 262144$  (resp.  $G \leq 1.048576 \times 10^6$ ) the difference between the upper bound  $\text{FrequencyUB}$  and the lower bound  $\text{SamplingLB}$  for Yahoo! dataset is smaller than 1.39% (resp. 2.46%). As long as  $\text{FrequencyUB}(G, S) - \text{SamplingLB}(G, S)$  the empirical distribution  $\hat{\lambda}_G$  will give us an accurate approximation of the guessing curve  $\lambda_G$  from the real (unknown) password distribution. However, as  $G$  becomes large the gap  $\text{FrequencyUB}(G, S) - \text{SamplingLB}(G, S)$  begins to widen e.g., for the Yahoo! dataset when  $G = 1.6777216 \times 10^7$  we have  $\text{FrequencyUB}(G, S) - \text{SamplingLB}(G, S) = 22.74\%$ . While  $\text{FrequencyUB}(G, S)$  and lower bound  $\text{SamplingLB}(G, S)$  give the best upper/lower bounds for smaller  $G$  we can see that the bounds reach plateau once  $G \geq \# \text{Distinct}$  e.g., for the Yahoo!

dataset  $\text{SamplingLB}(G, S)$  and  $\text{FrequencyUB}(G, S)$  remain constant for all  $G \geq 3.3885218 \times 10^7$ . Once  $G \geq \# \text{Distinct}$  we need new ideas upon the lower bound  $\lambda_G \geq \frac{N - \# \text{unique}}{N}$  or the upper bound  $\lambda_G \leq 1$ .

Our linear programming bounds  $\text{LPUB}$  and  $\text{LPLB}$  push past the  $G \leq \# \text{Distinct}$  barrier and allow us to obtain tighter upper and lower bounds even when  $G \geq \# \text{Distinct}$ . The linear programming bounds ( $\text{LPUB}$  and  $\text{LPLB}$ ) are worse when  $G$  is small e.g., for Yahoo! dataset when  $G = 262144$   $\text{FrequencyLB}$  and  $\text{SamplingLB}$  tightly bound  $\lambda_G$  as  $34.14\% \leq \lambda_G \leq 35.53\%$  while the LP bounds are  $3.58\% \leq \lambda_G \leq 45.68\%$ . However, as  $G$  increases we find that the linear programming approach yields significantly tighter bounds e.g., for the Yahoo! dataset when  $G = 6.7108864 \times 10^7$  our LP bounds show that  $61.75\% \leq \lambda_G \leq 77.72\%$  while our frequency and sampling based bounds  $56.41\% \leq \lambda_G \leq 100\%$  are much less tight. In such cases when  $\hat{\lambda}_G > \text{LPUP}(G, S)$  the empirical distribution *should not* be used to estimate the real guessing curve.

Similar to  $\text{SamplingLB}(G, S)$  our linear programming lower bound  $\text{LPLB}(G, S)$  also plateaus when  $G$  is sufficiently large, but it plateaus at a higher value e.g., 64.959% for Yahoo! dataset. We also note that both of our lower bounds  $\text{SamplingLB}$  and  $\text{LPLB}$  dramatically outperform the lower bound  $\text{PriorLB}$  based on prior work of Blocki et al [9].

**Are Password Cracking Models Guess Efficient?** Figure 1 demonstrates empirical cracking models are often much less guess efficient than an perfect knowledge attacker. In particular, if  $\hat{\lambda}_{G,M}$  denotes the percentage of passwords cracked by model  $M$  withing  $G$  guesses and  $\hat{\lambda}_{G,M} < \max\{\text{SamplingLB}(G, S), \text{LPLB}(G, S)\}$  then we can be confident that a perfect knowledge attacker would crack more passwords. For example, Figure 1a (resp. Figure 1d) shows that an attacker making  $G = 8390551$  (resp.  $G = 2101738$ ) guesses would crack *at most* 14% (resp. 42.14%) of 000webhost (resp. Brazzers) passwords using any password cracking model. By contrast, our lower bounds indicate that an attacker who knows the password distribution will crack *at least* 39.16% (resp. 58.05%) of 000webhost (resp. Brazzers) passwords. These results indicate that even state of the art password cracking models can be improved substantially. NIST guidelines for password hashing requires the use of moderately expensive password hashing functions such as PBKDF2 or even memory hard functions such as scrypt [48], Argon2 [5] or DRSample [2]. As guessing costs increase attackers will have additional incentives to develop password cracking models that are guess efficient.

**Reducing the Uncertain Region:** The lower bound  $\text{ExtendedLB}$  extends our sampling based approach with empirical password cracking models. The lower bound eventually improves on  $\text{SamplingLB}$  and  $\text{LPLB}$  for sufficiently large  $G$ . For example, for Brazzers dataset when  $G = 1.04433 \times 10^{16}$  our model based lower bound using neural network attack results ( $\text{ExtendedLB} : \text{NN}$ ) shows  $\lambda_G \geq 96.47\%$  while the best of our other lower bounds is only 58.62%.

Figure 1i plots the best upper/lower bounds (denoted as  $\text{ub/lb}$  in the figure) for the Yahoo!, RockYou, 000webhost, and Neopets datasets — to avoid overcrowding we ploy the best upper/lower bounds for Battlefield Heroes, Brazzers, Clixsense, and CSDN datasets in Appendix C. In particular, we plot  $\text{UB}(G, S) = \min\{\text{LPUP}(G, S), \text{FrequencyUB}(G, S)\}$  (solid curves) and  $\text{LB}(G, S) = \max\{\text{LPLB}(G, S),$

$\text{SamplingLB}(G, S), \text{ExtendedLB}\}$  (dotted curves). Notice that the lower bound appears to initially plateau before it starts to increase again e.g., for 000webhost dataset the lower bound plateaus at 55.55% when  $G = 3.35544 \times 10^7$ , barely increases when  $3.35544 \times 10^7 \leq G \leq 4.33431 \times 10^{10}$  and then starts to significantly increase again when  $G \geq 4.33431 \times 10^{10}$ . The initial plateau point is occurs when the lower  $\text{LPLB}(G, S)$  plateaus and once the empirical guessing curves “catch up” the curve starts to increase again.

The real (unknown) guessing curve  $\lambda_G$  lies somewhere in between  $\text{LB}(G, S)$  and  $\text{UB}(G, S)$ . While our work substantially tightens the gap  $\text{UB}(G, S) - \text{LB}(G, S)$ , there is still an uncertain region between the solid/dotted curves. We conjecture that improved password cracking models may be able to tighten this gap.

**The LinkedIn Dataset is Not IID** When we attempted to generate LP based upper/lower bounds for the LinkedIn frequency corpus [31] we discovered that there was no feasible solution that satisfies all of the constraints in  $\text{LP}_{\text{lower}}$  and  $\text{LP}_{\text{upper}}$ . This indicates that we cannot view the LinkedIn frequency corpus  $S$  as consisting of  $N$  independent samples from an unknown distribution i.e., the LinkedIn frequency corpus is inconsistent with any proposed password distribution  $\mathcal{D}$  if the samples are drawn independently. Indeed, according to [50] the dataset contains 177,500,189 passwords, but only 164,590,819 unique e-mail addresses. It is likely that many of the passwords were duplicated which would clearly violate our i.i.d. assumption. The ability to identify datasets which are inconsistent with an i.i.d. assumption (e.g., due to data duplication) can be viewed as a useful side benefit of our model. We can still generate upper/lower bounds for the LinkedIn frequency corpus using  $\text{FrequencyUB}(G, S)$  and  $\text{SamplingLB}(G, S)$  — see Figure 2 in Appendix C. However, these results should be interpreted with caution given the above observations.

**Zipf’s Law in Passwords** Zipf’s law [42, 57, 58] has been proposed as reasonable model for the password distribution  $\lambda_G$ . For example, CDF Zipf’s law estimates that  $\lambda_G \approx yG^r$  where the constant  $y, r > 0$  are tuned based on the sample  $S$ . Several papers [9, 57, 58] find that CDF Zipf’s law closely fits all known empirical password distributions. However, there is no guarantee theoretical guarantee that the estimate  $yG^r$  is close to  $\lambda_G$ . In Figure 5 we compare our upper/lower bounds with the CDF-Zipf curves using the parameters  $y, r$  from [57] for the datasets RockYou, Battlefield Heroes, 000webhost, CSDN and [9] for the more recent Yahoo! dataset [8, 13] — see Table 2. In all of the plots the CDF-Zipf plot (green) is close to our best upper bound (red). For the Battlefield, CSDN and 000webhost datasets the CDF-Zipf plot (green) lies in between our best upper bound (red) and our best lower bound (blue) indicating that the curve  $yG^r$  is consistent with our statistical bounds. For the RockYou and Yahoo! datasets the CDF-Zipf plots (green) often lie above the red upper bound e.g., for the RockYou (resp. Yahoo!) dataset when  $G = 33554432$  (resp.  $G = 134217728$ ) we have  $yG^r \geq 0.95$  (resp.  $yG^r \geq 0.98$ ) while our upper bounds imply that  $\lambda_G \leq 0.87$  (resp.  $\lambda_G \leq 0.86$ ). In such cases we can confidently state that the CDF-Zipf curve overestimates  $\lambda_G$ .

**Comparing LP1 with  $\text{LP}_{\text{lower}}$  and  $\text{LP}_{\text{upper}}$**  In Section 3.5 we first presented a linear program  $\text{LP1}$  to upper/lower bound  $\lambda_G$  under the idealized assumption that the real probability distribution is consistent with a fixed probability mesh. We later showed

how to eliminate this idealized assumption by adding a small slack terms in our constraints. In Figure 4 in Appendix C we compare the upper/lower bounds derived using LP1 (i.e., under idealized assumptions) with the upper/lower bounds generated using LP<sub>lower</sub> and LP<sub>upper</sub> respectively using the same parameter settings. We find that the bounds are very close indicating that the small slack terms we added to eliminate the idealized assumption does not negatively impact the quality of the upper/lower bounds.

## 5 RELATED WORK

**Password Hashing and Memory Hard Functions:** Key-stretching was proposed as early as 1979 [46] as a way to protect lower-entropy passwords against offline brute force attacks by making the password hash function moderately expensive to compute. Password hashing algorithms such as BCrypt [49] and PBKDF2 [35] use hash iteration to control guessing costs, but are potentially vulnerable to an attacker who uses FPGAs or Application Specific Integrated Circuits (ASICs) to dramatically reduce guessing costs. Blocki et al. [9] argued that hash iteration alone cannot provide sufficient protection for user password without introducing an unacceptably long delay during the authentication process i.e., minutes. Memory hard functions such as scrypt [48], Argon2 [5] or DRSample [2] are designed to force an attacker to allocate large amounts of memory for the duration of computation and are believed to be ASIC resistant. When attempting to tune the cost parameters of our key-stretching algorithm it will be important to understand the password distribution  $\lambda_G$ . We also note that as organizations start to use moderately expensive memory hard password hash functions like scrypt [48], Argon2 [5] or DRSample [2] offline attackers will have stronger incentives to develop guess efficient password cracking models.

**Offline Password Cracking Models and Defenses:** Offline password cracking has been studied for decades. Researchers have proposed many password cracking algorithms based on probabilistic password models such as Probabilistic Context-free Grammars (PCFG) [55, 59], Markov models [18, 25, 41, 47], and neural networks [45]. Monte-Carlo strength estimation [24] is a tool which allows us to efficiently approximate the guessing number of a given without requiring the defender to simulate the full attack. Liu et al. [40] developed tools to estimate guessing numbers for software tools such as John the Ripper (JtR) [33] and Hashcat [32] which are used more frequently by real world attackers. Juels and Rivest [34] suggested the use of honeywords (fake passwords) to help detect offline attacks. Compromised Credential Checking services such as HaveIBeenPwned and Google Password Checkup can be used to help alert users when one of their passwords have been breached [39, 52]. Distributed password hashing e.g., [27] ensures that the information needed to evaluate the password hash function is distributed across multiple servers so that a hacker who breaks into any individual server will not be able to mount an offline attack. Multifactor authentication provides another defense against password cracking attacks [16, 43].

**Strengthening the Password Distribution:** A large body of research has focused on encouraging (or forcing) users to pick

stronger passwords. Password composition policies [17, 38, 51] require users to pick passwords that comply with particular requirements e.g., passwords must at least one number and one upper case letter or passwords must be at least 8 characters long. Password composition policies often induce a substantial usability burden [1, 51] and can often be counter-productive [11, 38]. Telepathwords [37] is a password meter which encourages users to select stronger passwords by displaying realtime predictions for the next character that the user will type. Bonneau and Schecter [15] introduced the notion of incremental password strengthening where users are continually nudged to memorize one more character of a strong 56-bit password. Another line of work has focused on developing strategies for users to generate stronger passwords e.g., see [6, 10, 60, 61].

**Empirical Distribution:** The empirical password distribution has been used to evaluate many password research ideas. Harsha et al [31] used empirical distributions derived from the LinkedIn and RockYou datasets to quantify the advantage of an attacker after learning the length of the user’s password. The empirical password distribution has also been used to tune and evaluate distribution-aware password throttling mechanisms [12, 53] to defend against online attacks, distribution-aware mechanisms to tune relevant cost parameters for password hashing [3, 4, 7], achieve (personalized) password typo correction [19, 20] and evaluate the security of Compromised Credential Checking protocols [39].

**Estimating Properties of (Password) Distributions:** Valiant and Valiant [54] proposes an approach to accurately estimate key properties of *any* distribution over at most  $k$  distinct elements using  $O(k/\log k)$  independent and identically distributed (i.i.d.) samples. However, we cannot directly apply the results of Valiant and Valiant [54] to bound  $\lambda_G$  in our password setting as the number of distinct passwords  $k$  in the support of the password distribution  $\mathcal{P}$  is unknown to us and we almost certainly have  $k \gg N^2$  where  $N$  denotes the size of our largest password dataset. However, we are able to adapt the techniques of Valiant and Valiant [54] when developing the linear program that we use to upper/lower bound  $\lambda_G$ .

## 6 CONCLUSION

We introduced several statistical techniques to upper and lower bound  $\lambda_G$  the performance of a password cracker who knows the distribution from which passwords were sampled. Our upper/lower bounds hold with high confidence and can be derived from a password dataset even when the real password distribution is unknown to us. We applied our technique to analyze several large empirical password datasets. Our analysis demonstrates that the empirical guessing curve closely matches the real guessing curve as long as  $G$  is not too large, and highlights that state-of-the-art password cracking models are often far less guess efficient than a perfect knowledge attacker. For example, with the 000webhost dataset the state-of-the-art password cracking models indicate the attacker making 8390551 guesses would crack any password with probability 14% while our lower bound shows the probability is at least 39.16% with high confidence (99%). This shows that even the most sophisticated password cracking models still have a large room for improvement. While our results significantly narrows the uncertain region for  $\lambda_G$ , there are regions where our best upper/lower bounds

diverge significantly. Reducing this gap will help us to better understand the distribution over user chosen passwords and is an important challenge for future research.

## ACKNOWLEDGMENTS

Jeremiah Blocki was supported in part by the National Science Foundation under awards CNS #1755708 and CNS #2047272, a gift from Protocol Labs, and by a Purdue Big Ideas award. Peiyuan Liu was supported in part by a Purdue Big Ideas award and by NSF CNS #1755708.

## REFERENCES

- [1] Anne Adams and Martina Angela Sasse. 1999. Users are not the enemy. *Commun. ACM* 42, 12 (1999), 40–46.
- [2] Joël Alwen, Jeremiah Blocki, and Ben Harsha. 2017. Practical Graphs for Optimal Side-Channel Resistant Memory-Hard Functions. In *ACM CCS 2017*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, 1001–1017. <https://doi.org/10.1145/3133956.3134031>
- [3] Wenjie Bai and Jeremiah Blocki. 2021. DAHash: Distribution Aware Tuning of Password Hashing Costs. *arXiv preprint arXiv:2101.10374* (2021).
- [4] Wenjie Bai, Jeremiah Blocki, and Ben Harsha. 2020. Information Signaling: A Counter-Intuitive Defense Against Password Cracking. *arXiv preprint arXiv:2009.10060* (2020).
- [5] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. 2016. Argon2: new generation of memory-hard functions for password hashing and other applications. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 292–302.
- [6] Jeremiah Blocki, Manuel Blum, and Anupam Datta. 2013. Naturally Rehearsing Passwords. In *ASIACRYPT 2013, Part II (LNCS)*, Kazuo Sako and Palash Sarkar (Eds.), Vol. 8270. Springer, Heidelberg, 361–380. [https://doi.org/10.1007/978-3-642-42045-0\\_19](https://doi.org/10.1007/978-3-642-42045-0_19)
- [7] Jeremiah Blocki and Anupam Datta. 2016. CASH: A Cost Asymmetric Secure Hash Algorithm for Optimal Password Protection. In *CSF 2016 Computer Security Foundations Symposium*, Michael Hicks and Boris Köpf (Eds.). IEEE Computer Society Press, 371–386. <https://doi.org/10.1109/CSF.2016.33>
- [8] Jeremiah Blocki, Anupam Datta, and Joseph Bonneau. 2016. Differentially Private Password Frequency Lists. In *NDSS 2016*. The Internet Society.
- [9] Jeremiah Blocki, Benjamin Harsha, and Samson Zhou. 2018. On the Economics of Offline Password Cracking. In *2018 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 853–871. <https://doi.org/10.1109/SP.2018.00009>
- [10] Jeremiah Blocki, Saranga Komanduri, Lorrie Faith Cranor, and Anupam Datta. 2015. Spaced Repetition and Mnemonics Enable Recall of Multiple Strong Passwords. In *NDSS 2015*. The Internet Society.
- [11] Jeremiah Blocki, Saranga Komanduri, Ariel Procaccia, and Or Sheffet. 2013. Optimizing Password Composition Policies. In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce (EC '13)*. Association for Computing Machinery, New York, NY, USA, 105–122. <https://doi.org/10.1145/2482540.2482552>
- [12] Jeremiah Blocki and Wuwei Zhang. 2020. Dalock: Distribution aware password throttling. *arXiv preprint arXiv:2005.09039* (2020).
- [13] Joseph Bonneau. 2012. The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords. In *2012 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 538–552. <https://doi.org/10.1109/SP.2012.49>
- [14] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. 2012. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *2012 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 553–567. <https://doi.org/10.1109/SP.2012.44>
- [15] Joseph Bonneau and Stuart E. Schechter. 2014. Towards Reliable Storage of 56-bit Secrets in Human Memory. In *USENIX Security 2014*, Kevin Fu and Jaeyeon Jung (Eds.). USENIX Association, 607–623.
- [16] John Brainard, Ari Juels, Ronald L. Rivest, Michael Szydlo, and Moti Yung. 2006. Fourth-factor authentication: Somebody you know. In *ACM CCS 2006*, Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati (Eds.). ACM Press, 168–178. <https://doi.org/10.1145/1180405.1180427>
- [17] John Campbell, Wanli Ma, and Dale Kleeman. 2011. Impact of restrictive composition policy on user password choices. *Behaviour & Information Technology* 30, 3 (2011), 379–388.
- [18] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. 2012. Adaptive Password-Strength Meters from Markov Models. In *NDSS 2012*. The Internet Society.
- [19] Rahul Chatterjee, Anish Athayle, Devdatta Akhawe, Ari Juels, and Thomas Ristenpart. 2016. pASSWORD iYOPS and How to Correct Them Securely. In *2016 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 799–818. <https://doi.org/10.1109/SP.2016.53>
- [20] Rahul Chatterjee, Joanne Woodage, Yuval Pnueli, Anusha Chowdhury, and Thomas Ristenpart. 2017. The TypTop System: Personalized Typo-Tolerant Password Checking. In *ACM CCS 2017*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, 329–346. <https://doi.org/10.1145/3133956.3134000>
- [21] Joseph Cox. May 5, 2016. Another Day, Another Hack: Tens of Millions of Neopets Accounts. [https://motherboard.vice.com/en\\_us/article/ezpvw7/neopets-hack-another-day-another-hack-tens-of-millions-of-neopets-accounts](https://motherboard.vice.com/en_us/article/ezpvw7/neopets-hack-another-day-another-hack-tens-of-millions-of-neopets-accounts). (May 5, 2016).
- [22] Joseph Cox. September 5, 2016. Nearly 800,000 Brazzers Porn Site Accounts Exposed in Forum Hack. [https://motherboard.vice.com/en\\_us/article/vv7pgd/nearly-800000-brazzers-porn-site-accounts-exposed-in-forum-hack](https://motherboard.vice.com/en_us/article/vv7pgd/nearly-800000-brazzers-porn-site-accounts-exposed-in-forum-hack). (September 5, 2016).
- [23] Nik Cubrilovic. December 15, 2009. RockYou Hack: From Bad To Worse. <https://techcrunch.com/2009/12/14/rockyou-hack-security-myspace-facebook-passwords/>. (December 15, 2009).
- [24] Matteo Dell’Amico and Maurizio Filippone. 2015. Monte Carlo Strength Evaluation: Fast and Reliable Password Checking. In *ACM CCS 2015*, Indrajit Ray, Ninghui Li, and Christopher Kruegel (Eds.). ACM Press, 158–169. <https://doi.org/10.1145/2810103.2813631>
- [25] Markus Dürmuth, Fabian Angelstorf, Claude Castelluccia, Daniele Perito, and Abdelber Chaabane. 2015. OMEN: Faster password guessing using an ordered markov enumerator. In *International Symposium on Engineering Secure Software and Systems*. Springer, 119–132.
- [26] Jean-Philippe Aumasson et al. 2015. Password Hashing Competition. (2015). <https://password-hashing.net/>.
- [27] Adam Everspaugh, Rahul Chatterjee, Samuel Scott, Ari Juels, and Thomas Ristenpart. 2015. The Pythia PRF Service. In *USENIX Security 2015*, Jaeyeon Jung and Thorsten Holz (Eds.). USENIX Association, 547–562.
- [28] Dan Goodin. October 28, 2015. 13 million plaintext passwords belonging to webhost users leaked online. <https://arstechnica.com/information-technology/2015/10/13-million-plaintext-passwords-belonging-to-webhost-users-leaked-online/>. (October 28, 2015).
- [29] Dan Goodin. September 13, 2016. 6.6 million plaintext passwords exposed as site gets hacked to the bone. <https://arstechnica.com/information-technology/2016/09/6-million-plaintext-passwords-exposed-as-site-gets-hacked-to-the-bone/>. (September 13, 2016).
- [30] LLC Gurobi Optimization. 2021. Gurobi Optimizer Reference Manual. (2021). <http://www.gurobi.com>
- [31] Benjamin Harsha, Robert Morton, Jeremiah Blocki, John Springer, and Melissa Dark. 2021. Bicycle attacks considered harmful: Quantifying the damage of widespread password length leakage. *Computers & Security* 100 (2021), 102068. <https://doi.org/10.1016/j.cose.2020.102068>
- [32] Hashcat. Hashcat. <https://hashcat.net/hashcat/>. (????). Accessed March 15, 2021.
- [33] John the Ripper. John the Ripper. <https://www.openwall.com/john/>. (????). Accessed March 15, 2021.
- [34] Ari Juels and Ronald L. Rivest. 2013. Honeywords: making password-cracking detectable. In *ACM CCS 2013*, Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung (Eds.). ACM Press, 145–160. <https://doi.org/10.1145/2508859.2516671>
- [35] Burt Kaliski. 2000. PKCS# 5: Password-based cryptography specification version 2.0. (2000).
- [36] Saranga Komanduri. 2016. Modeling the adversary to evaluate password strength with limited samples. *Ph.D. dissertation* (2016).
- [37] Saranga Komanduri, Richard Shay, Lorrie Faith Cranor, Cormac Herley, and Stuart E. Schechter. 2014. Telepathwords: Preventing Weak Passwords by Reading Users’ Minds. In *USENIX Security 2014*, Kevin Fu and Jaeyeon Jung (Eds.). USENIX Association, 591–606.
- [38] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman. 2011. Of passwords and people: measuring the effect of password-composition policies. In *CHI 2011*. ACM Press, 2595–2604. <http://dl.acm.org/citation.cfm?id=1979321>
- [39] Lucy Li, Bijeeta Pal, Junade Ali, Nick Sullivan, Rahul Chatterjee, and Thomas Ristenpart. 2019. Protocols for Checking Compromised Credentials. In *ACM CCS 2019*, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM Press, 1387–1403. <https://doi.org/10.1145/3319535.3354229>
- [40] Enze Liu, Amanda Nakanishi, Maximilian Golla, David Cash, and Blase Ur. 2019. Reasoning Analytically about Password-Cracking Software. In *2019 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 380–397. <https://doi.org/10.1109/SP.2019.00070>
- [41] Jerry Ma, Weining Yang, Min Luo, and Ninghui Li. 2014. A Study of Probabilistic Password Models. In *2014 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 689–704. <https://doi.org/10.1109/SP.2014.50>
- [42] David Malone and Kevin Maher. 2012. Investigating the distribution of password choices. In *Proceedings of the 21st international conference on World Wide Web*.

- 301–310.
- [43] Mohammad Mannan and P. C. van Oorschot. 2011. Leveraging Personal Devices for Stronger Password Authentication from Untrusted Computers. *J. Comput. Secur.* 19, 4 (Dec. 2011), 703–750.
- [44] Colin McDiarmid. 1989. On the method of bounded differences. *Surveys in combinatorics* 141, 1 (1989), 148–188.
- [45] William Melicher, Blase Ur, Sean M. Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2016. Fast, Lean, and Accurate: Modeling Password Guessability Using Neural Networks. In *USENIX Security 2016*, Thorsten Holz and Stefan Savage (Eds.). USENIX Association, 175–191.
- [46] Robert Morris and Ken Thompson. 1979. Password security: A case history. *Commun. ACM* 22, 11 (1979), 594–597.
- [47] Arvind Narayanan and Vitaly Shmatikov. 2005. Fast Dictionary Attacks on Passwords Using Time-Space Tradeoff. In *ACM CCS 2005*, Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels (Eds.). ACM Press, 364–372. <https://doi.org/10.1145/1102120.1102168>
- [48] Colin Percival. 2009. Stronger key derivation via sequential memory-hard functions. (2009).
- [49] Niels Provos and David Mazieres. 1999. Bcrypt algorithm. In *USENIX*.
- [50] Rick Redman. May 19, 2016. LinkedIn Revisited – Full 2012 Hash Dump Analysis. [https://blog.korelogic.com/blog/2016/05/19/linkedin\\_passwords\\_2016](https://blog.korelogic.com/blog/2016/05/19/linkedin_passwords_2016). (May 19, 2016).
- [51] Richard Shay, Saranga Komanduri, Patrick Gage Kelley, Pedro Giovanni Leon, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2010. Encountering stronger password requirements: user attitudes and behaviors. In *Proceedings of the Sixth Symposium on Usable Privacy and Security (SOUPS '10)*. ACM, New York, NY, USA, Article 2, 20 pages. <https://doi.org/10.1145/1837110.1837113>
- [52] Kurt Thomas, Jennifer Pullman, Kevin Yeo, Ananth Raghunathan, Patrick Gage Kelley, Luca Invernizzi, Borbala Benko, Tadek Pietraszek, Sarvar Patel, Dan Boneh, and Elie Bursztein. 2019. Protecting accounts from credential stuffing with password breach alerting. In *USENIX Security 2019*, Nadia Heninger and Patrick Traynor (Eds.). USENIX Association, 1556–1571.
- [53] Yuan Tian, Cormac Herley, and Stuart Schechter. 2019. StopGuessing: Using guessed passwords to thwart online guessing. In *2019 IEEE European Symposium on Security and Privacy (EuroSecP)*. IEEE, 576–589.
- [54] Gregory Valiant and Paul Valiant. 2017. Estimating the unseen: Improved estimators for entropy and other properties. *Journal of the ACM (JACM)* 64, 6 (2017), 1–41.
- [55] Rafael Veras, Christopher Collins, and Julie Thorpe. 2014. On Semantic Patterns of Passwords and their Security Impact. In *NDSS 2014*. The Internet Society.
- [56] John Walker. June 26, 2011. LulzSec Over, Release Battlefield Heroes Data. <https://www.rockpapershotgun.com/2011/06/26/lulzsec-over-release-battlefield-heroes-data/>. (June 26, 2011).
- [57] Ding Wang, Haibo Cheng, Ping Wang, Xinyi Huang, and Gaopeng Jian. 2017. Zipf’s Law in Passwords. *IEEE Transactions on Information Forensics and Security* 12, 11 (2017), 2776–2791. <https://doi.org/10.1109/TIFS.2017.2721359>
- [58] Ding Wang and Ping Wang. 2016. On the implications of Zipf’s law in passwords. In *European Symposium on Research in Computer Security*. Springer, 111–131.
- [59] Matt Weir, Sudhir Aggarwal, Breno de Medeiros, and Bill Glodek. 2009. Password Cracking Using Probabilistic Context-Free Grammars. In *2009 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 391–405. <https://doi.org/10.1109/SP.2009.8>
- [60] Jeff Yan, Alan Blackwell, Ross Anderson, and Alasdair Grant. 2004. Password memorability and security: Empirical results. *IEEE Security & privacy* 2, 5 (2004), 25–31.
- [61] Weining Yang, Ninghui Li, Omar Chowdhury, Aiping Xiong, and Robert W. Proctor. 2016. An Empirical Study of Mnemonic Sentence-based Password Generation Strategies. In *ACM CCS 2016*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM Press, 1216–1229. <https://doi.org/10.1145/2976749.2978346>
- [62] Xue Yang. December 26, 2011. Chinese Internet Suffers the Most Serious User Data Leak in History. <https://blogs.forcepoint.com/security-labs/chinese-internet-suffers-most-serious-user-data-leak-history>. (December 26, 2011).

## A MISSING PROOFS

### A.1 Missing Proofs in Section 3.1

Here we present the missing proofs of Theorem 3.3 and Corollary 3.4 in Section 3.1.

Consider an arbitrary sample set  $S = \{s_1, \dots, s_N\} \leftarrow \mathcal{P}^N$  with size  $N$ . Same to Blocki et al [9], We define  $B_i$  to be an indicator random variable such that  $B_i = 1$  if and only if  $f_i^S \geq j$  while  $p_i < \frac{1}{NL}$ ;

otherwise,  $B_i = 0$ . We call a sample  $s_i$  is  $(j, L)$  – *bad overestimate* if  $B_i = 1$ . To analyzing the lower bound of the number of cracked user passwords in the sample set  $S$ , we first bound the number of samples in  $S$  that are  $(j, L)$  – *bad overestimate*.

LEMMA A.1. For any  $t \geq 0$ , we have:

$$\Pr\left[\sum_i f_i^S \times B_i \geq \frac{N}{(j-1)!L^{j-1}} + t\right] \leq \exp\left(\frac{-2t^2}{Nj^2}\right).$$

where the randomness is taken over the sample set  $S$  with size  $N$ .

PROOF. We let  $g(s_1, s_2, \dots, s_N)$  be a function that outputs the number of  $(j, L)$  – *bad overestimate* samples, i.e.  $\sum_i f_i^S \times B_i$ . Then for any two different input  $s_1, \dots, s_i, \dots, s_N$  and  $s_1, \dots, s'_i, \dots, s_N$  which only differ on the  $i$  – *th* input, the difference between two outputs of  $g$  will be no greater than  $j$ , i.e.,  $\sup_{s_1, \dots, s_i, \dots, s_N, s'_i} |g(s_1, \dots, s_i, \dots, s_N) - g(s_1, \dots, s'_i, \dots, s_N)| \leq j$ . Claim 6 in Blocki et al [9] proves that  $\mathbb{E}(\sum_i f_i^S \times B_i) \leq \frac{N}{(j-1)!L^{j-1}}$ .

Using Theorem 3.1, we have:

$$\begin{aligned} \Pr\left[\sum_i f_i^S \times B_i \geq \frac{N}{(j-1)!L^{j-1}} + t\right] \\ \leq \Pr[g(s_1, \dots, s_N) \geq \mathbb{E}(g(s_1, \dots, s_N)) + t] \leq \exp\left(\frac{-2t^2}{Nj^2}\right) \end{aligned}$$

□

**Reminder of Theorem 3.3.** For any  $L \geq 1, t_1 \geq 0$  and any integers  $N \geq 1$  and  $j \geq 2$  setting  $G = NL$  we have:

$$\Pr\left[\lambda(S, G) \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right)\right] \geq 1 - \exp\left(\frac{-2t^2}{Nj^2}\right)$$

where the randomness is taken over the sample set  $S \leftarrow \mathcal{P}^N$ . *Proof of Theorem 3.3.* We consider a perfect knowledge attacker who makes  $G = N \times L$  guesses for any  $L \geq 1$ . The  $G$  guesses contains the  $G$  passwords with the top  $G$  highest probabilities. Since there are at most  $NL$  passwords with probability no less than  $\frac{1}{NL}$ , the perfect knowledge attacker making  $G = NL$  guesses must crack all of them. Note that for any password  $pwd_i$  satisfying  $f_i^S \geq j$ , we have either  $p_i \geq \frac{1}{NL}$  or  $B_i = 1$ . Therefore, at least  $\sum_{i: f_i^S \geq j} f_i^S - \sum_i f_i^S \times B_i$  will be cracked by a perfect knowledge attacker making  $G = NL$  guesses, i.e.  $N\lambda(S, G) \geq \sum_{i: f_i^S \geq j} f_i^S - \sum_i f_i^S \times B_i$ . Using Lemma A.1 we have:

$$\begin{aligned} \Pr[\lambda(S, G) \geq \frac{1}{N} (\sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t)] \\ \geq \Pr[\sum_{i: f_i^S \geq j} f_i^S - \sum_i f_i^S \times B_i \geq \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t] \\ = \Pr[\sum_i f_i^S \times B_i \leq \frac{N}{(j-1)!L^{j-1}} + t] \geq 1 - \exp\left(\frac{-2t^2}{Nj^2}\right). \end{aligned}$$

**Reminder of Corollary 3.4.** Given a sample set  $S \leftarrow \mathcal{P}^N$  with size  $N$ , for any  $L \geq 1, t \geq 0, 0 \leq \epsilon \leq 1$ , any integer  $j \geq 2$ , the

expected percentage of passwords cracked by a perfect knowledge attacker making  $G = N \times L$  guesses is bounded as below:

$$\Pr[\lambda_G \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right) - \epsilon] \geq 1 - \delta$$

where  $\delta = \exp\left(\frac{-2t^2}{Nj^2}\right) + \exp(-2N\epsilon^2)$ , and the randomness is taken over the sample set  $S \leftarrow \mathcal{P}^N$ . Proof of Corollary 3.4. Using Theorem 3.2 we have:

$$\begin{aligned} & \Pr[\lambda_G \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right) - \epsilon] \\ & \geq \Pr[\lambda_G \geq \lambda(S, G) - \epsilon \wedge \lambda(S, G) \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right)] \\ & \geq \Pr[\lambda_G \geq \lambda(S, G) - \epsilon] \\ & \quad - \Pr[\lambda(S, G) \geq \frac{1}{N} \left( \sum_{i: f_i^S \geq j} f_i^S - \frac{N}{(j-1)!L^{j-1}} - t \right)] \\ & \geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{Nj^2}\right). \end{aligned}$$

## A.2 Other Missing Proofs

**Reminder of Lemma 3.12.** Fix any  $i \geq 0$  and  $0 \leq \epsilon_{2,i} \leq 1$  and assume that  $x_l < \frac{1}{N}$  and that  $\mathcal{P}$  is  $l$ -partially consistent with our mesh  $X$ . If  $i = 0$  let  $W_0$  be an indicator random variable which is 1 if and only if  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^l h_j x_j \cdot \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^l h_j x_j \cdot \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l)$ . Similarly, if  $i > 0$  let  $W_i$  be an indicator random variable which is 1 if and only if  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) \leq \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \leq \epsilon_{2,i}$ . Then the constraints hold with probability

$$\Pr[W_i = 1] \geq 1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$$

where the randomness is taken over the selection of  $S \leftarrow \mathcal{P}^N$ . Proof of Lemma 3.12. Note that for any  $x < x_l < \frac{1}{N}$  we have  $\text{bpdf}(0, N, x_l) \leq \text{bpdf}(0, N, x) \leq 1$  for  $i = 0$  and  $0 \leq \text{bpdf}(i, N, x) \leq \text{bpdf}(i, N, x_l)$  for any  $i \geq 1$ . Then when  $i = 0$  we have the following bounds on  $\sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j)$ :

$$\begin{aligned} & \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_j) \\ & \leq \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_l) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - p \times \text{bpdf}(i, N, x_l), \end{aligned}$$

and similarly, for our lower bound we have

$$\begin{aligned} & \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_j) \\ & \geq \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - p; \end{aligned}$$

Similarly, when  $i > 0$  we can derive the following upper/lower bounds

$$\begin{aligned} & \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_j) \\ & \leq \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j), \end{aligned}$$

and

$$\begin{aligned} & \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_j) \\ & \geq \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - \sum_{j > l} h_j x_j \times \text{bpdf}(i, N, x_l) \\ & = \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) - p \times \text{bpdf}(i, N, x_l). \end{aligned}$$

Applying Lemma 3.10, for any  $i \geq 0$  and  $0 \leq \epsilon_{2,i} \leq 1$ , we have  $\frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \leq \sum_{j \geq 1} h_j x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}$  with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . Therefore, we have:

$$\begin{aligned} & \text{if } i = 0, \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \\ & \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l); \\ & \text{if } i \geq 1, \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) \\ & \leq \sum_{j=1}^l h_j x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i}. \end{aligned}$$

where for any  $i \geq 0$  and  $0 \leq \epsilon_{2,i} \leq 1$  the inequality holds with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ .

## B LINEAR PROGRAM AND MISSING PROOFS

In this section, we provide a complete description of the linear programming approach we describe in Section 3.5.3 and a complete proof of Theorem 3.14.

Recall that we fix a particular mesh  $X_{l,q} = \{x_1, \dots, x_l\}$  where where  $x_l = \frac{1}{10^4 N}$ ,  $l = \lfloor \frac{\ln(\frac{1}{x_l})}{\ln q} \rfloor + 1$  and for each  $1 \leq i < l$  we have  $x_i = q \cdot x_{i+1} = q^{l-i} x_l$ . Here,  $q > 1$  is a parameter that determines how fine-grained our mesh values are. We also consider an ideal mesh  $X^r = \{x_1^r, x_2^r, \dots\} = \{\Pr[pwd] : pwd \in \mathcal{P}\}$  for the real distribution  $\mathcal{P}$  along with a corresponding histogram  $H^r = \{h_1^r, h_2^r, \dots\}$  where  $h_i^r \triangleq |\{pwd \in \mathcal{P} : \Pr[pwd] = x_i^r\}|$ . Given the number of guesses  $G > 0$  we have  $\lambda_G = X_{i(G,H)} c(G, H) + \sum_{j=1}^{i(G,H)-1} x_j^r h_j^r$  where we define  $i(G, H) = \max\{j : \sum_{i=1}^{j-1} h_i \leq G\}$  and  $c(G, H) = G - \sum_{i=1}^{i(G,H)-1} h_i$  as before.

The LP to generate lower bound is not exactly the same as the LP to generate upper bound. We discuss them separately in the following sections. We present LP<sub>lower</sub> for lower bounds and prove the lower bound in Theorem 3.14 in Section B.1, and present LP<sub>upper</sub> for upper bounds and prove the upper bound in Theorem 3.14 in Section B.2.

### B.1 Lower Bound

Let  $x_k^r = \min_{i \geq 1} \{x_i^r : x_i^r \geq x_l\}$  be the smallest probability in  $P$  no less than  $x_l$ . We will use  $X_l$  to estimate  $X_k^r = \{x_1^r, \dots, x_k^r\}$ , and let  $p = \sum_{i>k} x_i^r \times h_i^r$  be the remaining probability. Specifically, for any  $1 \leq i \leq k$ , we map  $x_i^r \in X_k^r$  to its closest value  $x_{b_i}$  where  $b_i = \arg \max_{1 \leq j \leq l} \{x_j : x_j \leq x_i^r\}$  in  $X_l$  that are equal to or smaller than  $x_i^r$ . Then we have  $x_{b_i} \leq x_i^r \leq q \times x_{b_i}$ . Therefore, for any  $1 \leq j \leq l$ , the histogram  $h_j$  of  $x_j$  is the sum of all values in  $H^r$  with the corresponding mesh values being mapped to  $x_j$ , i.e.,  $h_j = \sum_{i:b_i=j} h_i^r$ . Recall that for any  $G \geq 0$ ,  $G = \sum_{i=1}^{i(G,H^r)-1} h_i^r + c(G, H^r)$ . We can use  $H$  instead of  $H^r$  to represent  $G$  by defining  $i(G, H) = b_{i(G,H^r)}$  and  $c(G, H) = \sum_{j:b_j=i(G,H)} h_j^r + c(G, H^r)$  such that  $G = c(G, H) + \sum_{i=1}^{i(G,H)-1} h_i$ . Then we have

$$\begin{aligned} \lambda_G &= c(G, H^r) x_{i(G,H^r)} + \sum_{i=1}^{i(G,H^r)-1} x_i h_i^r \\ &\geq c(G, H^r) x_{b_{i(G,H^r)}} + \sum_{i=1}^{i(G,H^r)-1} x_{b_i} h_i^r \\ &= c(G, H) x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i \end{aligned}$$

Therefore, given a set of fine-grained mesh values  $X_l$  and a guessing number  $G$ , as long as we can find a lower bound of  $c(G, H) x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i$  using linear programming, it will also be a lower bound of  $\lambda_G$ .

The linear programming task for bounding  $c(G, H) x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i$  in this section is similar to LP1a we described in Section 3.5.2, except that Constraint (2) and (3) need to be modified considering the difference between our fine-grained mesh  $X_l$  and the real probabilities  $X_k^r$ . For Constraint (3), previously we have

$\sum_{j=1}^k h_j^r \times x_j^r = 1 - p$ . After mapping  $x_j^r$  to  $x_{b_j}$  with  $x_{b_j} \leq x_j^r \leq q \times x_{b_j}$  for each  $1 \leq j \leq k$ , we have:

$$\begin{aligned} \sum_{j=1}^l h_j \times x_j &= \sum_{j=1}^k h_j^r \times x_{b_j} \leq \sum_{j=1}^k h_j^r \times x_j^r = 1 - p \\ \sum_{j=1}^l h_j \times x_j &= \sum_{j=1}^k h_j^r \times x_{b_j} \geq \sum_{j=1}^k h_j^r \times \left(\frac{1}{q} \times x_j^r\right) = \frac{1-p}{q} \end{aligned}$$

Constraint (2) that bounds  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  also need to be changed considering the difference between  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)$ . We will present the change and the proof in Lemma B.1 later in this section.

We use the final linear program LP<sub>lower</sub> to lower bound  $\lambda_G$ . This linear program LP<sub>lower</sub> can be considered as an extension from LP1 and LP1a removing all ideal settings.

LP<sub>lower</sub> takes  $h_1, \dots, h_l, c, p$  as variables, and minimize  $\sum_{j<idx} (h_j \times x_j + c \times x_{idx})$  with the constraint  $\sum_{j<idx} h_j + c = G$ . Given an arbitrary guessing number  $G$ , fixing other input parameters, we can lower bound  $\lambda_G$  as  $\min_{1 \leq idx \leq l+1} \text{LP}_{\text{lower}}(X_l, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})$  by running LP<sub>lower</sub>( $X_l, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3}$ ) for  $l+1$  times with  $idx \in \{1, 2, \dots, l+1\}$ .

The following lemma proves Constraint (2) in LP<sub>lower</sub> holds with high probability:

LEMMA B.1. Given  $F^S = \{F_1, \dots, F_N\}$  from a sample set  $S \leftarrow \mathcal{P}^N$ , for any  $q > 1$ ,  $i \geq 0$ ,  $0 \leq \epsilon_{2,i} \leq 1$ ,  $\frac{i+1}{N+1} \leq \hat{x}_{\epsilon_{3,i}} \leq 1$ , and  $\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1 \in (0, 1)$ , let  $X_l = X_l^q$ , then we have:

if  $i = 0$ ,

$$\begin{aligned} \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \right) &\leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\ &\leq (1 + \epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, qx_l) \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) \end{aligned}$$

if  $i \geq 1$ ,

$$\begin{aligned} \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, qx_l) \right) \\ \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\ \leq (1 + \epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) \end{aligned}$$

where for any  $i$  the two inequalities hold with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ .

PROOF. Recall that for any  $x_l < \frac{1}{N}$ , any individual  $i \geq 0$ , and any  $0 \leq \epsilon_{2,i} \leq 1$ , the following constraint holds with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ , Lemma 3.12 proves the following



constraint holds with probability at least  $1 - 2 \times \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ :

if  $i = 0$ ,

$$\begin{aligned} \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p &\leq \sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) \\ &\leq \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_k^r); \end{aligned}$$

if  $i \geq 1$ ,

$$\begin{aligned} \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p &\times \text{bpdf}(i, N, x_l) \\ &\leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i}. \end{aligned}$$

Note that  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) = \sum_{j=1}^k h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})$ .

To prove that the difference between  $\sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)$  and  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  is small, we look into  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})}$  for each  $1 \leq j \leq k$ .

One the one side, since  $x_{b_j} \leq x_j^r \leq qx_{b_j}$ , we bound it as  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})} = \frac{(x_j^r)^{i+1} (1-x_j^r)^{N-i}}{x_{b_j}^{i+1} (1-x_{b_j})^{N-i}} \leq q^{i+1}$ . Therefore, with probability at least  $1 - \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ , we have

$$\begin{aligned} &\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\ &= \sum_{j: x_j^r \geq x_l} h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j}) \\ &\geq \frac{1}{q^{i+1}} \sum_{j: x_j^r \geq x_l} h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) \\ &\geq \begin{cases} \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \right) & i = 0 \\ \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_k^r) \right) & i > 0 \end{cases} \\ &\geq \begin{cases} \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \right) & i = 0 \\ \frac{1}{q^{i+1}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, qx_l) \right) & i > 0 \end{cases} \end{aligned}$$

The left side of the inequality in this lemma is proved. On the other side, we consider two cases. Define a function  $f(x) = x^{i+1}(1-x)^{N-i}$  for any  $i \geq 0$ . Since

$$\begin{aligned} f'(x) &= (i+1)x^i(1-x)^{N-i} - (N-i)x^{i+1}(1-x)^{N-i-1} \\ &= x^i(1-x)^{N-i-1}((i+1)(1-x) - (N-i)x) \\ &= x^i(1-x)^{N-i-1}(-(N+1)x + (i+1)) \end{aligned}$$

$f(x)$  monotonically increases when  $0 \leq x \leq \frac{i+1}{N+1}$ , and monotonically decreases when  $\frac{i+1}{N+1} \leq x \leq 1$ . Therefore, for  $x_j^r \leq \frac{i+1}{N+1}$ , we have:

$$\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})} = \frac{(x_j^r)^{i+1} (1-x_j^r)^{N-i}}{x_{b_j}^{i+1} (1-x_{b_j})^{N-i}} \geq 1;$$

for  $x_j^r > \frac{i+1}{N+1}$ , we have:

$$\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})} = \frac{(x_j^r)^{i+1} (1-x_j^r)^{N-i}}{x_{b_j}^{i+1} (1-x_{b_j})^{N-i}} \geq q^{i+1} \left( \frac{1-qx_{b_j}}{1-x_{b_j}} \right)^{N-i}.$$

Picking a parameter  $\hat{x}_{\epsilon_{3,i}} \geq \frac{i+1}{N+1}$  we can find the corresponding  $\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1-\hat{x}_{\epsilon_{3,i}}}{1-q\hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1 \in (0, 1)$  such that for any  $x_j^r \leq qx_{b_j} \leq q\hat{x}_{\epsilon_{3,i}}$  we have  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j})} \geq q^{i+1} \left( \frac{1-qx_{b_j}}{1-x_{b_j}} \right)^{N-i} = \frac{1}{1+\epsilon_{3,i}}$ . Therefore, we can get the right side of the inequality as below:

$$\begin{aligned} &\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\ &= \sum_{j: q\hat{x}_{\epsilon_{3,i}} \geq x_j^r \geq x_l} h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j}) \\ &\quad + \sum_{j: x_j^r > q\hat{x}_{\epsilon_{3,i}}} h_j^r \times x_{b_j} \times \text{bpdf}(i, N, x_{b_j}) \\ &\leq (1+\epsilon_{3,i}) \sum_{j: x_j^r \geq x_l} h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) \\ &\leq \begin{cases} (1+\epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_k^r) \right) \\ \quad + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) & i = 0 \\ (1+\epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) & i > 0 \end{cases} \\ &\leq \begin{cases} (1+\epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, qx_l) \right) \\ \quad + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) & i = 0 \\ (1+\epsilon_{3,i}) \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} \right) + \text{bpdf}(i, N, \hat{x}_{\epsilon_{3,i}}) & i > 0 \end{cases} \end{aligned}$$

The inequality holds with probability at least  $1 - \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ .  $\square$

Therefore, the lower bound in Theorem 3.14 is proved, i.e.  $\lambda_G \geq \min_{1 \leq id_x \leq l+1} \text{LP}(\text{lower}(X_l, F^S, id_x, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3}))$  with probability at least  $1 - 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ , where the randomness is taken over the sample set  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

## B.2 Upper Bound

Let  $x_k^r = \min_{i \geq 1} \{x_i^r : x_i^r \geq x_l/q\}$  be the smallest probability in  $P$  no less than  $x_l/q$ . Note that for upper bounds the definition of  $x_k^r$  is slightly different from the  $X_k^r$  for lower bounds in Section B.1 above. We will use  $X_l$  to estimate  $X_k^r = \{x_1^r, \dots, x_k^r\}$ , and let  $p = \sum_{i>k} x_i^r \times h_i^r$  be the remaining probability. Specifically, for any  $1 \leq i \leq k$ , we map  $x_i^r \in X_k^r$  to its closest value  $x_{a_i}$  where  $a_i = \arg \min_{1 \leq j \leq l} \{x_j : x_j \geq x_i^r\}$  in  $X_l$  that are equal to or greater than  $x_i^r$ . Then we have  $\frac{x_{a_i}}{q} \leq x_i^r \leq x_{a_i}$ . Therefore, for any  $1 \leq j \leq l$ , the histogram  $h_j$  of  $x_j$  is the sum of all values in  $H^r$  with the corresponding mesh values being mapped to  $x_j$ , i.e.,  $h_j = \sum_{i: a_i=j} h_i^r$ . Recall that for any  $G \geq 0$ ,  $G = \sum_{i=1}^{i(G, H^r)-1} h_i^r + c(G, H^r)$ . We can use  $H$  instead of  $H^r$  to represent  $G$  by defining  $i(G, H) = a_{i(G, H^r)}$  and  $c(G, H) =$

$\sum_{j:a_j=i(G,H)} h_j^r + c(G, H^r)$  such that  $G = c(G, H) + \sum_{i=1}^{i(G,H)-1} h_i$ . Then we have

$$\begin{aligned}\lambda_G &= c(G, H^r)x_{i(G,H^r)} + \sum_{i=1}^{i(G,H^r)-1} x_i h_i^r \\ &\leq c(G, H^r)x_{a_i(G,H^r)} + \sum_{i=1}^{i(G,H^r)-1} x_{a_i} h_i^r \\ &= c(G, H)x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i\end{aligned}$$

Therefore, given a set of fine-grained mesh values  $X_l$  and a guessing number  $G$ , as long as we can find an upper bound of  $c(G, H)x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i$  using linear programming, it will also be an upper bound of  $\lambda_G$ .

The linear programming task for bounding  $c(G, H)x_{i(G,H)} + \sum_{i=1}^{i(G,H)-1} x_i h_i$  in this section is similar to LP1a we described in Section 3.5.2, except that Constraint (2) and (3) need to be modified considering the difference between our fine-grained mesh  $X_l$  and the real probabilities  $X_k^r$ . For Constraint (3), previously we have  $\sum_{j=1}^k h_j^r \times x_j^r = 1 - p$ . After mapping  $x_j^r$  to  $x_{a_j}$  with  $\frac{x_{a_j}}{q} \leq x_i^r \leq x_{a_i}$  for each  $1 \leq j \leq k$ , we have:

$$\begin{aligned}\sum_{j=1}^l h_j \times x_j &= \sum_{j=1}^k h_j^r \times x_{a_j} \geq \sum_{j=1}^k h_j^r \times x_j^r = 1 - p \\ \sum_{j=1}^l h_j \times x_j &= \sum_{j=1}^k h_j^r \times x_{a_j} \leq \sum_{j=1}^k h_j^r \times (q \times x_j^r) = q(1 - p)\end{aligned}$$

Constraint (2) that bounds  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  also need to be changed considering the difference between  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  and  $\sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)$ . We will present the change and the proof in Lemma B.2 later in this section.

We use the final linear program LPupper to upper bound  $\lambda_G$ . This linear program LPupper can be considered as an extension from LP1 and LP1a removing the two ideal settings.

LPupper takes  $h_1, \dots, h_l, c, p$  as variables, and maximize  $\sum_{j < \text{id}_x} (h_j \times x_j + c \times x_{\text{id}_x})$  with the constraint  $\sum_{j < \text{id}_x} h_j + c = G$ . Given an arbitrary guessing number  $G$ , fixing other input parameters, we can upper bound  $\lambda_G$  as  $\max_{1 \leq \text{id}_x \leq l+1} \text{LPupper}(X_l, F^S, \text{id}_x, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})$

by running  $\text{LPupper}(X_l, F^S, \text{id}_x, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})$  for  $l+1$  times with  $\text{id}_x \in \{1, 2, \dots, l+1\}$ .

The following lemma proves Constraint (2) in LPupper holds with high probability:

LEMMA B.2. Given  $F^S = \{F_1, \dots, F_N\}$  from a sample set  $S \leftarrow \mathcal{P}^N$ , for any  $q > 1$ ,  $i \geq 0$ ,  $0 \leq \epsilon_{2,i} \leq 1$ ,  $\frac{i+1}{N+1} \leq \hat{x}_{\epsilon_{3,i}} \leq 1$ , and  $\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q\hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1 \in (0, 1)$ , let  $X_l = X_l^q$ , then we have:

if  $i = 0$ ,

$$\begin{aligned}&\frac{1}{1 + \epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) \\ &\leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l) \right)\end{aligned}$$

if  $i \geq 1$ ,

$$\begin{aligned}&\frac{1}{1 + \epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} \right. \\ &\quad \left. - p \times \text{bpdf}(i, N, x_l) - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) \\ &\leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} \right)\end{aligned}$$

where for any  $i$  the two inequalities hold with probability at least  $1 - 2 \times \exp\left(-\frac{2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ .

PROOF. Recall that for any  $x_l < \frac{1}{N}$ , any individual  $i \geq 0$ , and any  $0 \leq \epsilon_{2,i} \leq 1$ , the following constraint holds with probability at least  $1 - 2 \times \exp\left(-\frac{2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . Lemma 3.12 proves the following constraint holds with probability at least  $1 - 2 \times \exp\left(-\frac{2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ :

if  $i = 0$ ,

$$\begin{aligned}&\frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \leq \sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) \\ &\leq \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_k^r);\end{aligned}$$

if  $i \geq 1$ ,

$$\begin{aligned}&\frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) \\ &\leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq \frac{(i+1)F_{i+1}}{N-i} + \epsilon_{2,i}.\end{aligned}$$

Note that  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) = \sum_{j=1}^k h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})$ . To prove that the difference between  $\sum_{j=1}^k h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)$  and  $\sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j)$  is small, we look into  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})}$  for each  $1 \leq j \leq k$ .

One the one side, since  $\frac{x_{a_j}}{q} \leq x_j^r \leq x_{a_j}$ , we have  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})} = \frac{(x_j^r)^{i+1} (1 - x_j^r)^{N-i}}{x_{a_j}^{i+1} (1 - x_{a_j})^{N-i}} \geq \frac{1}{q^{i+1}}$ . Therefore, with probability at least  $1 -$

$\exp\left(\frac{-2(N-i)^2\epsilon_{2,i}^2}{N(i+1)^2}\right)$ , we have

$$\begin{aligned}
& \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\
&= \sum_{j: x_j^r \geq \frac{x_l}{q}} h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j}) \\
&\leq q^{i+1} \sum_{j: x_j^r \geq \frac{x_l}{q}} h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) \\
&\leq \begin{cases} q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_2 - p \times \text{bpdf}(i, N, x_k^r) \right) & i = 0 \\ q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_2 \right) & i > 0 \end{cases} \\
&\leq \begin{cases} q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_2 - p \times \text{bpdf}(i, N, x_l) \right) & i = 0 \\ q^{i+1} \left( \frac{(i+1)F_{i+1}}{N-i} + \epsilon_2 \right) & i > 0 \end{cases}
\end{aligned}$$

The right side of the inequality in this lemma is proved. On the other side, we consider two cases. Recall that for any  $i \geq 0$   $f(x) = x^{i+1}(1-x)^{N-i}$  monotonically increases when  $0 \leq x \leq \frac{i+1}{N+1}$ , and monotonically decreases when  $\frac{i+1}{N+1} \leq x \leq 1$ . Therefore, for  $x_j^r \leq \frac{i+1}{N+1}$ , we have:

$$\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})} = \frac{(x_j^r)^{i+1}(1-x_j^r)^{N-i}}{x_{a_j}^{i+1}(1-x_{a_j})^{N-i}} \leq 1;$$

for  $x_j^r > \frac{i+1}{N+1}$ . we have:

$$\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})} = \frac{(x_j^r)^{i+1}(1-x_j^r)^{N-i}}{x_{a_j}^{i+1}(1-x_{a_j})^{N-i}} \leq \frac{1}{q^{i+1}} \left( \frac{1-x_{a_j}}{1-x_{a_j}} \right)^{N-i}$$

Picking a parameter  $\hat{x}_{\epsilon_{3,i}} \geq \frac{i+1}{N+1}$  we can find the corresponding  $\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1-\hat{x}_{\epsilon_{3,i}}}{1-q\hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1 \in (0, 1)$  such that for any  $x_{a_j} \leq q\hat{x}_{\epsilon_{3,i}}$  we have  $\frac{h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r)}{h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j})} \leq \frac{1}{q^{i+1}} \left( \frac{1-x_{a_j}}{1-x_{a_j}} \right)^{N-i} = (1 +$

$\epsilon_{3,i}$ ). Therefore, we can get the right side of the inequality as below:

$$\begin{aligned}
& \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \\
&= \sum_{j: x_j^r \geq \frac{x_l}{q}} h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j}) \\
&\geq \sum_{j: q\hat{x}_{\epsilon_{3,i}} \geq x_j^r \geq \frac{x_l}{q}} h_j^r \times x_{a_j} \times \text{bpdf}(i, N, x_{a_j}) \\
&\geq \frac{1}{1+\epsilon_{3,i}} \left( \sum_{j: q\hat{x}_{\epsilon_{3,i}} \geq x_j^r \geq \frac{x_l}{q}} h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) \right) \\
&\geq \frac{1}{1+\epsilon_{3,i}} \left( \sum_{j: x_j^r \geq \frac{x_l}{q}} h_j^r \times x_j^r \times \text{bpdf}(i, N, x_j^r) - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) \\
&\geq \begin{cases} \frac{1}{1+\epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) & i=0 \\ \frac{1}{1+\epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_k^r) - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) & i>0 \end{cases} \\
&\geq \begin{cases} \frac{1}{1+\epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) & i=0 \\ \frac{1}{1+\epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) & i>0 \end{cases}
\end{aligned}$$

The inequality holds with probability at least  $1 - \exp\left(\frac{-2(N-i)^2\epsilon_{2,i}^2}{N(i+1)^2}\right)$ .  $\square$

Therefore, the upper bound in Theorem 3.14 is proved, i.e.  $\lambda_G \leq \max_{1 \leq i \leq l+1} \text{LPupper}(X_l, F^S, i, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3})$  with probability at least  $1 - 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2\epsilon_{2,i}^2}{N(i+1)^2}\right)$ , where the randomness is taken over the sample set  $S$  of size  $N$ .

### B.3 The Linear Program LPupper

Below we show the linear program LPupper that is used to generate the upper bound (LPUB(S, G)) in Theorem 3.14.

### Linear Programming Task 3:

LPupper( $G, X_l, F^S, idx, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3}$ )

**Input Parameters:**  $G, X_l = \{x_1, \dots, x_l\}, F^S = \{F_1^S, \dots, F_N^S\}, idx, i', \epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\}, \hat{x}_{\epsilon_3} = \{\hat{x}_{\epsilon_{3,0}}, \dots, \hat{x}_{\epsilon_{3,i'}}\}, \epsilon_3 = \{\epsilon_{3,i} = \frac{1}{q^{i+1}} \left( \frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}} \right)^{N-i} - 1, 0 \leq i \leq i'\}$

**Variables:**  $h_1, \dots, h_l, c, p$

**Objective:**  $\max \left( \sum_{j < idx} h_j \times x_j + c \times x_{idx} \right)$

**Constraints:**

$$(1) \sum_{j < idx} h_j + c = G$$

$$(2) \forall 0 \leq i \leq i':$$

$$(a) \text{ for } i = 0, \frac{1}{1 + \epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq q^{i+1} \left( \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} - p \times \text{bpdf}(i, N, x_l) \right)$$

$$(b) \text{ for } 1 \leq i \leq i', \frac{1}{1 + \epsilon_{3,i}} \left( \frac{(i+1)F_{i+1}^S}{N-i} - \epsilon_{2,i} - \frac{i+1}{N-i} - p \times \text{bpdf}(i, N, x_l) - \text{bpdf}(i, N, q\hat{x}_{\epsilon_{3,i}}) \right) \leq \sum_{j=1}^l h_j \times x_j \times \text{bpdf}(i, N, x_j) \leq q^{i+1} \left( \frac{(i+1)F_{i+1}^S}{N-i} + \epsilon_{2,i} \right)$$

$$(3) 1 - p \leq \sum_{j=1}^l h_j \times x_j \leq q \times (1 - p)$$

$$(4) 0 \leq c \leq h_{idx}$$

(Note: we consider  $idx = 1, 2, \dots, l+1$ . When  $idx = l+1$ , we define  $h_{l+1} = G$  and  $x_{l+1} = x_l$ .)

## C EXTRA FIGURES AND TABLES

The theoretical bounds we generated for LinkedIn dataset is in Figure 2.

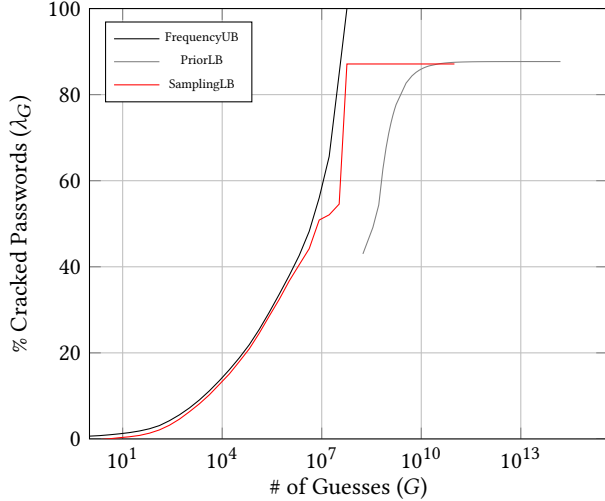


Figure 2: LinkedIn Guessing Curves (N=174292189)

We have shown the best upper/lower bounds for Yahoo!, Rock-You, 000webhost, and Neopets in Figure 1i in the main body of the paper. Here we plot the best upper/lower bounds for Battlefield Heroes, Brazzers, Clixsense, and CSDN datasets in Figure 3.

We compare the upper/lower bounds generated under idealized assumptions (i.e. LP1) with the regular high-confidence LP

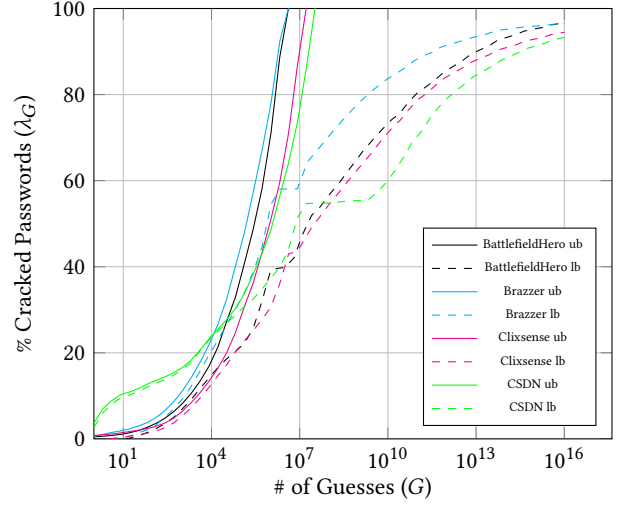


Figure 3: Best Upper/Lower Bounds for Battlefield Heroes, Brazzers, Clixsense, and CSDN

Table 2: CDF Zipf's Law Parameters [9, 58]

Dataset (S)	y	r
Yahoo! [8]	0.03315	0.1811
RockYou [23]	0.0374	0.1872
000webhost [28]	0.0059	0.2816
Battlefield Heroes [56]	0.0103	0.2949
CSDN [62]	0.0588	0.1486

upper/lower bounds (i.e. LP1lower and LP1upper) in Figure 4. Figure 4 shows the bounds are very close.

We plot CDF Zipf's Law estimates  $yG^r$  of  $\lambda_G$  for RockYou, Yahoo!, 000webhost, Battlefield Heroes, and CSDN dataset in Figure 5, using the parameters provided in Wang and Wang [58] and Blocki et al [9]. The parameters are shown in Table 2.

## D BOUNDING $\lambda(G, S)$

By Theorem 3.2 any high confidence upper/lower bound on  $\lambda_G$  immediately yields a high confidence upper/lower bound on  $\lambda(G, S)$  since  $\lambda(G, S)$  is tightly concentrated around its mean  $\mathbb{E}[\lambda(G, S)] = \lambda_G$ . We formally state each corollary below.

Corollary D.1 is a corollary of Theorem 3.7.

COROLLARY D.1. For any  $G \geq 1$  and any parameters  $0 < d < N$ ,  $0 \leq \epsilon \leq 1$ ,  $t \geq 0$ , we have:

$$\Pr[\lambda(S, G) \geq \frac{1}{d}(h(D_1, D_2, G) - t) - \epsilon] \geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right)$$

where the randomness is taken over the samples  $D_1 \leftarrow \mathcal{P}^{N-d}$  and  $D_2 \leftarrow \mathcal{P}^d$ .

PROOF. Using Theorem 3.2 we have  $\Pr[\lambda(S, G) \geq \lambda_G - \epsilon] \geq 1 - \exp(-2N\epsilon^2)$ . Then using Theorem 3.7, for any  $0 \leq \epsilon \leq 1$  and

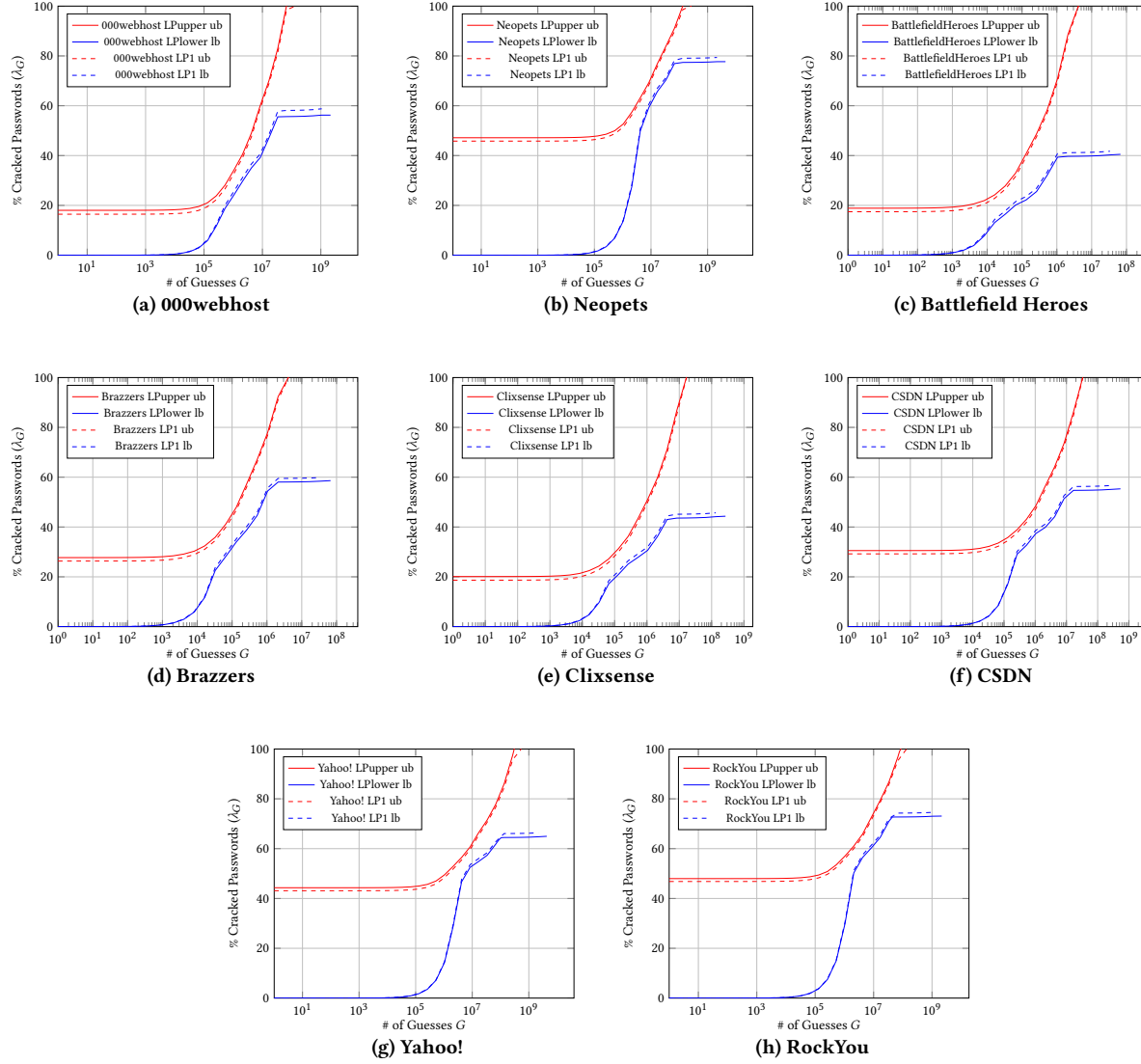


Figure 4: Comparison of LP1 and LP1lower LPupper

any  $t \geq 0$  we have:

$$\begin{aligned}
 & \Pr[\lambda(S, G) \geq \frac{1}{d}(h(D_2, G) - t) - \epsilon] \\
 & \geq \Pr[\lambda(S, G) \geq \lambda_G - \epsilon \wedge \lambda_G \geq \frac{1}{d}(h(D_2, G) - t)] \\
 & \geq \Pr[\lambda(S, G) \geq \lambda_G - \epsilon] - \Pr[\lambda_G < \frac{1}{d}(h(D_2, G) - t)] \\
 & \geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right).
 \end{aligned}$$

Therefore, for any guessing number  $G > 0$  and any parameter  $d, \epsilon_1, t_2$ , we can lower bound the percentage of cracked passwords as below:

$$\lambda(S, G) \geq \frac{1}{d}(h(D_2, G) - t) - \epsilon$$

where the left inequality holds with probability at least  $1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right)$ .  $\square$

Corollary D.2 follows from Theorem 3.8 by applying Theorem 3.2.

**COROLLARY D.2.** For any guessing number  $G > 0$  and any parameters  $0 < d < N$ ,  $0 \leq \epsilon \leq 1$ ,  $t \geq 0$ , we have

$$\begin{aligned}
 & \Pr[\lambda(S, G) \geq \frac{1}{d}(h'_M(D_1, D_2, G) - t) - \epsilon] \\
 & \geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right)
 \end{aligned}$$

where the randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

**PROOF.** Using Theorem 3.1, for any  $0 \leq \epsilon \leq 1$  we have  $\Pr[\lambda(S, G) \geq \sum_{i \leq G} P_i - \epsilon] \geq 1 - \exp(-2N\epsilon^2)$ . Then using Theorem 3.8, for any

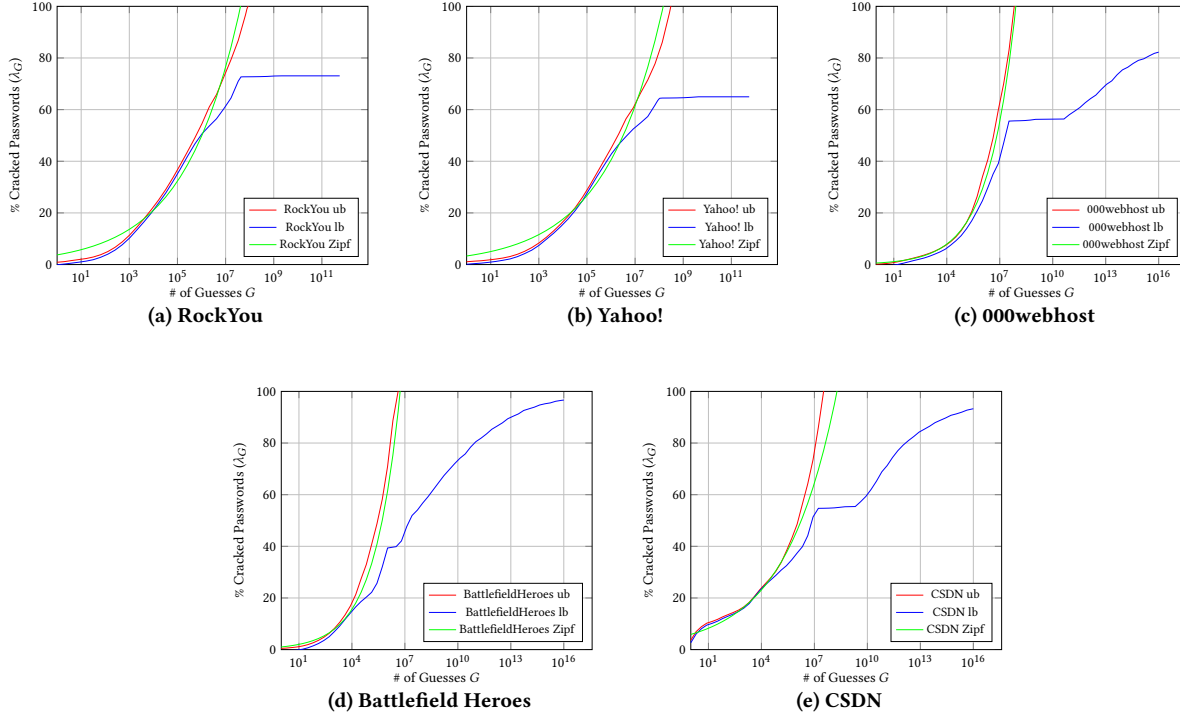


Figure 5: Comparison of CDF Zipf's Law Guessing Curves and Our Best Upper/Lower Bounds

$0 \leq \epsilon \leq 1, t \geq 0$  we have:

$$\begin{aligned} \Pr[\lambda(S, G) \geq \frac{1}{d}(h'_M(D_1, D_2, G) - t) - \epsilon] \\ \geq 1 - \exp(-2N\epsilon^2) - \exp\left(\frac{-2t^2}{d}\right). \end{aligned}$$

□

Corollary D.3 follows from Theorem 3.11 by applying Theorem 3.2.

**COROLLARY D.3.** *Given a probability distribution  $\mathcal{P}$  which is consistent with a finite mesh  $X_I = \{x_1, \dots, x_I\}$  for any  $G \geq 0$ , any  $i' \geq 0$ , any  $1 \leq \epsilon_1 > 0$  and any  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}$ , we have:*

$$\begin{aligned} \Pr\left[\lambda_G \geq \min_{1 \leq idx \leq l} \text{LP1}(X_I, F^S, idx, G, 1, i', \epsilon_2)\right] &\geq 1 - \delta \\ \Pr\left[\lambda_G \leq \max_{1 \leq idx \leq l} |\text{LP1}(X_I, F^S, idx, G, -1, i', \epsilon_2)|\right] &\geq 1 - \delta \end{aligned}$$

where  $\delta = \exp(-2N\epsilon_1^2) + 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . The randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

Corollary D.4 follows from Theorem 3.13 by applying Theorem 3.2.

**COROLLARY D.4.** *Suppose that the probability distribution  $\mathcal{P}$  is l-partially consistent with a mesh  $X = \{x_1, x_2, \dots\}$  then for any  $G \geq 0$ , any  $i' \geq 0$ , any  $0 \leq \epsilon_1 \leq 1$ , any  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}$  we*

have:

$$\begin{aligned} \Pr\left[\lambda_G \geq \min_{1 \leq idx \leq l} \text{LP1a}(X_I, F^S, idx, G, 1, i', \epsilon_2)\right] &\geq 1 - \delta \\ \Pr\left[\lambda_G \leq \max_{1 \leq idx \leq l} |\text{LP1a}(X_I, F^S, idx, G, -1, i', \epsilon_2)|\right] &\geq 1 - \delta \end{aligned}$$

where  $\delta = \exp(-2N\epsilon_1^2) + 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . The randomness is taken over the sample  $S \leftarrow \mathcal{P}^N$  of size  $N$ .

Corollary D.5 follows from Theorem 3.14 by applying Theorem 3.2.

**COROLLARY D.5.** *Given an unknown password distribution  $\mathcal{P}$ , a sample set  $S \leftarrow \mathcal{P}^N$ , for any  $G \geq 0$ , integer  $i' \geq 0$ ,  $\epsilon_2 = \{\epsilon_{2,0}, \dots, \epsilon_{2,i'}\} \in [0, 1]^{i'+1}$ ,  $\hat{x}_{\epsilon_3} = \{\hat{x}_{\epsilon_{3,0}}, \dots, \hat{x}_{\epsilon_{3,i'}}\}$ ,  $\epsilon_3 = \{\epsilon_{3,i} = \frac{1}{q^{i+1}}(\frac{1-\hat{x}_{\epsilon_{3,i}}}{1-q\hat{x}_{\epsilon_{3,i}}})^{N-i} - 1, 0 \leq i \leq i'\}$ , we can bound  $\lambda(S, G)$  using a set of fine-grained meshes  $X_{l,q}$  we generate as below:*

$$\begin{aligned} \Pr[\lambda(S, G) \geq \min_{1 \leq idx \leq l+1} \text{LP1lower}(X_{l,q}, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3}) - \epsilon_1] &\geq 1 - \delta \\ \Pr[\lambda(S, G) \leq \max_{1 \leq idx \leq l+1} \text{LP1upper}(X_{l,q}, F^S, idx, G, i', \epsilon_2, \epsilon_3, \hat{x}_{\epsilon_3}) - \epsilon_1] &\geq 1 - \delta \end{aligned}$$

where each inequality holds with probability at least  $\delta = \exp(-2N\epsilon_1^2) + 2 \times \sum_{0 \leq i \leq i'} \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$ . The randomness is taken over the sample  $S$  of size  $N$ .

## E PARAMETER SETTING

To generate high-confidence theoretical bounds of  $\lambda_G$  and  $\lambda(S, G)$ , we assign the parameters values to guarantee every bound in Section 3 holds with probability *at least* 0.99.

Recall that the parameter  $\epsilon$  in Corollary 3.4, Corollary 3.6, Theorem 3.7, Corollary 3.9 and  $\epsilon_1$  in Theorem 3.14 bound the difference between  $\lambda_G$  and  $\lambda(S, G)$ . We fix this parameter as  $\epsilon = \epsilon_1 = \left(\frac{\ln(\delta_1)}{-2N}\right)^{\frac{1}{2}}$  where we set  $\delta_1 = 0.00009$  for all bounds of  $\lambda_G$  and  $\lambda(S, G)$ . Then the upper bound in Corollary 3.6 (FrequencyUB( $S, G$ )) holds with probability at least  $1 - \delta_1 \geq 0.99$ .

We denote  $\delta_{2,j} = \exp\left(\frac{-2t^2}{Nj^2}\right)$  in Corollary 3.4. We set  $\delta_{2,j} = 0.01 - \delta_1$  (i.e.  $t = \left(\frac{Nj^2 \ln(\delta_{2,j})}{-2}\right)^{\frac{1}{2}}$ ) such that the lower bound in Corollary 3.4 (PriorLB( $S, G, j$ )) holds with probability at least  $1 - \delta_1 - \delta_{2,j} \geq 0.99$  for any  $j \geq 2$ .

We denote  $\delta_3 = \exp\left(\frac{-2t^2}{d}\right)$  in both Theorem 3.7 and Corollary 3.9. We set  $d = 25000$  and  $\delta_3 = 0.01 - \delta_1$  (i.e.  $t = \left(\frac{d \ln(\delta_3)}{-2}\right)^{\frac{1}{2}}$ ) such that the lower bound in Corollary 3.9 (ExtendedLB( $S, G$ )) extends the lower bound in Theorem 3.7 (SamplingLB( $S, G$ )), and both of them hold with probability at least  $1 - \delta_3 \geq 0.99$ .

We denote  $\delta_{4,i} = \exp\left(\frac{-2(N-i)^2 \epsilon_{2,i}^2}{N(i+1)^2}\right)$  in Theorem 3.14. we set  $q = 1.002$ ,  $i' = 4$ ,  $\{\delta_{4,0}, \delta_{4,1}, \delta_{4,2}, \delta_{4,3}, \delta_{4,4}\} = \{0.00009, 0.000165, 0.00175, 0.00175, 0.0012\}$ , and  $\{\hat{x}_{\epsilon_{3,0}}, \hat{x}_{\epsilon_{3,1}}, \hat{x}_{\epsilon_{3,2}}, \hat{x}_{\epsilon_{3,3}}, \hat{x}_{\epsilon_{3,4}}\} = \{7.0/N, 11.0/N, 14.0/N, 16.3/N, 18.5/N\}$ . Note that  $\epsilon_{2,i} = \left(\frac{N(i+1)^2 \ln(\delta_{4,i})}{-2(N-i)^2}\right)^{\frac{1}{2}}$  and  $\epsilon_{3,i} = \frac{1}{q^{i+1}} \left(\frac{1 - \hat{x}_{\epsilon_{3,i}}}{1 - q \hat{x}_{\epsilon_{3,i}}}\right)^{N-i} - 1$  for any  $0 \leq i \leq i'$ . Both of the upper and lower bounds in Theorem 3.14 (LPUB( $S, G$ ) and LPLB( $S, G$ )) hold with probabilities at least  $1 - \sum_{i=0}^{i'} \delta_{4,i} \geq 0.99$ .