The Pennsylvania State University

The Graduate School

College of Engineering

# ADAPTIVE PASSWORD METERS WITH UNRELIABLE STORAGE

A Thesis in

Computer Science and Engineering

by

Megan Heysham

©2013 Megan Heysham

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2013

The thesis of Megan Heysham was reviewed and approved* by the following:

Adam Smith
Associate Professor of Computer Science and Engineering
Thesis Adviser

Trent Jaeger
Professor of Computer Science and Engineering

Lee Coraor
Associate Professor of Computer Science and Engineering
Chair of Graduate Program of the Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.

## ABSTRACT

User-chosen passwords continue to be a primary means of authorization on computerized systems. They are a cheap solution that is easy to use and maintain. Unfortunately, when asked to pick a password, most users will choose from the same relatively small pool of all possible strings. Efforts to encourage better selection of passwords include providing guidelines (which are often fulfilled in a predictable manner) and password meters (which are often inaccurate).

Password meters should not be seen as a "one size fits all" solution, as different systems see different password distributions, and what is secure on one system may be a very popular choice on another. Due to this variability, password meters that base score on the existing set of passwords may perform better. For such an adaptive password meter to function, it must have access to some information about the passwords other than their hashed values. This means that the system must store this additional information. This information, if leaked, may compromise users' passwords.

In this thesis, we introduce a security model for adaptive password meters, reveal a vulnerability in previous work, and propose and analyze a new scheme for an adaptive password meter. There is currently little published work on adaptive password meters or on their security concerns.

## TABLE OF CONTENTS

## LIST OF FIGURES

# 1 Introduction

User-chosen passwords continue to be a primary means of authorization on computerized systems. They are a cheap solution that is easy to use and maintain. Unfortunately, when asked to pick a password, most users will choose from the same relatively small pool of all possible strings. Efforts to encourage better selection of passwords include providing guidelines (which are often fulfilled in a predictable manner) and password meters (which are often inaccurate).

Password meters should not be seen as a "one size fits all" solution, as different systems see different password distributions, and what is secure on one system may be a very popular choice on another. Due to this variability, password meters that base score on the existing set of passwords may perform better. For such an adaptive password meter to function, it must have access to some information about the passwords other than their hashed values. This means that the system must store this additional information. This information, if leaked, may compromise users' passwords.

In this thesis, we introduce a security model for adaptive password meters, reveal a vulnerability in previous work, and propose and analyze a new scheme for an adaptive password meter. There is currently little published work on adaptive password meters or on their security concerns. In the next section, we summarize previous work to put our contributions in context. In Section 1.2 we list our contributions.

## 1.1 Previous work

Passwords are unlikely to be replaced any time soon. They are cheap and easy to maintain and use [8]. The most secure passwords are those that are randomly-generated by the system. However, these passwords are difficult for users to remember, and users tend not to like them [12, 16, 21]. For these reasons, many systems ask users to select their own passwords.

Unfortunately, many studies have shown that human-chosen passwords tend to be insecure. A simple dictionary of commonly-chosen passwords can be used to crack especially weak passwords. Markov-chain-based password crackers take advantage of the fact that users tend to choose passwords with similar character distributions to their native language [14].

In hopes of helping users make smarter choices when selecting passwords, systems often give guidelines for creating strong passwords. For example, www.google.com suggests (but does not require) that passwords contain a mix of letters, digits, and symbols. When these guidelines are made explicit, users are more likely to follow them [7]. Unfortunately, even when these password composition policies are in place, most users follow predictable patterns. For example, if users are told that passwords must contain a digit, many will simply append a digit to the end of a weaker password. These predictable password structures can be exploited by crackers [20]. When given advice to create a password from a memorable phrase, users tend to choose common phrases from pop culture enabling dictionary attacks against these passwords [11].

In addition to guidelines, password meters can be used to give users feedback on the strength of their passwords. A password meter takes as input a password and outputs a "score" that estimates how secure the password is. A password meter should estimate the ease with which an adversary would be able to guess the password. The mere presence of a password meter will influence users toward passwords that receive a higher score [18], especially if the password is protecting an important account [6]. Because password meters can have influence on user choice, it is important that they are strict enough and accurate so users have correct information [18].

Password meters may simply differentiate between "strong" and "weak" passwords, using presence in a dictionary to determine that a password is weak. A more space-efficient and flexible solution is to store the dictionary as a Markov model [3]. More commonly, password meters give a score based on length, estimated Shannon entropy of character composition, and a lack dictionary words. These scores are based on password guidelines proposed by NIST [1].

[19, 13, 9, 17, 10] show that the use of entropy as defined by NIST does not accurately measure password strength, assigning some weak passwords a high score while assigning some strong passwords a low score. Features such as requiring certain character classes and setting a minimum password length do not guarantee that passwords will be secure. In part, this is because users follow predictable patterns when including the various character classes, as noted above.

One reason for the difficulty in estimating password strength is that password distributions tend to be site-specific [2]. For instance, a user is much more likely use "faithwriters" in a password for an account on www.faithwriters.com than for an account on www.myspace.com. These different distributions mean that a password might be strong on one website while being weak on another. From this observation, [2] defines an adaptive password meter, which uses previously-created passwords in determining the strength of a newly-created password. [15] uses a count-min sketch to detect passwords that are already common within the system. However, the meter from [2] uses Markov models and is capable of detecting not only common passwords but also variations on common passwords. This adaptive password meter outperforms many existing password meters in terms of accuracy.

Adaptive password meters must store information about passwords other than their hashed values. Although there is evidence that adaptive password meters give good strength estimates, they have not been thoroughly studied, and in particular, their security requires a much closer examination.

## 1.2 Contributions

This thesis makes the following contributions:

- We define a security model for adaptive password meters with unreliable storage. There is currently little in the literature about adaptive password meters or how access to their storage may aid password cracking. Repeated access to this storage may compromise systems that are otherwise secure.

- We identify a vulnerability in a published adaptive password meter scheme. This vulnerability arises from developing the scheme under a different security model, which we believe to be less realistic than our security model.

- We define an adaptive password meter scheme in which we attempt to alleviate the security concerns in the previous scheme.

- We define a framework for analyzing security under repeated leakage and analyze our password meter. With this framework, we are able to loosely bound the advantage that repeated leakage gives an adversary.

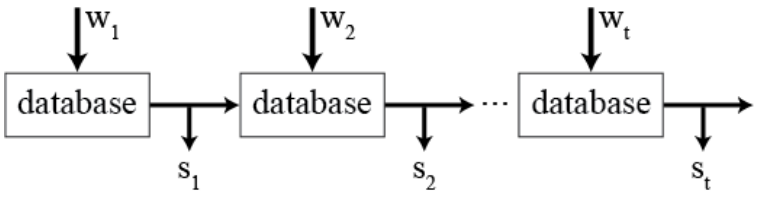


Figure 1: Repeated leakage of the internal state of an adaptive password meter.

### 1.3 Organization

The remainder of this thesis is organized as follows. In Section 2 we define a security model for adaptive password meters and discuss a vulnerability in previous work. In Section 3 we define a new scheme for an adaptive password meter. In Section 4 we define a framework for analyzing adaptive password meter security and analyze our scheme in this framework. In Section 5 we discuss the effects of password meters on password distributions and attack time. Finally, we conclude with Section 6.

## 2 Security model

For an adaptive password meter to function, it must have access to some information about the passwords other than their hashed values. This means that the system must store this additional information. We assume that all internal state relevant to the password meter is at risk for being leaked to an adversary [5]. In addition, systems aren't hacked in a 'one-shot' way. That is, once an adversary gains access to a system, he is able to continue observing changes to the system. Let $w_i$ be the $i$th password chosen in the system, and let $s_i$ be the internal state of the password meter after $w_i$ has been added. We assume that an adversary has access to $s_1, s_2, ...s_t$, and define $z_i = s_i - s_{i-1}$ to be the change in state when then $i$th password is added. Therefore, security requires that access to $s_1, s_2, ...s_t$ not significantly improve an adversary's ability to crack passwords. We also assume that there is a range of acceptable passwords lengths and that the adversary knows this range.

### 2.1 The CDP adaptive password meter

Castelluccia et al. [2] designed an adaptive password meter (henceforth called the CDP meter) using Markov models. The CDP meter takes as input passwords composed of letters from some alphabet $\Sigma$. For some chosen value $n$, the meter maintains a database containing a count of each string of $n$ characters, or $n$-gram, within the set of existing passwords. Thus the internal state is $s_i = \{count_i(x_1, ...x_n) \mid x_1, ...x_n \in \Sigma^n\}$.

The CDP meter was designed to be secure against one instance of the $n$-gram database being leaked. If a completely accurate count was stored (and leaked), then strong passwords, containing only rare $n$-grams, could be reconstructed with high probability. For this reason, the CDP meter also adds noise to the database each time a password is created.

The database is initialized according to Algorithm 1.

**foreach** *n-gram* $x_1, ...x_n$ **do**
  $count_0(x_1, ...x_n) = 0$;
**end**

**Algorithm 1:** CDP initialization

When password $w_i = c_1, c_2, ..., c_m$ is created in the system, the database is updated according to Algorithm 2.

**for** $k = 1, ..., m-n+1$ : **do**
  $count_i(c_k, ..., c_{k+n-1}) = count_{i-1}(c_k, ..., c_{k+n-1}) + 1$
**end**
**foreach** *n-gram* $x_1, ...x_n$ **do**
  $count_i(x_1, ...x_n) = count_i(x_1, ...x_n) + 1$ with probability $\gamma$;
**end**
Hash $w$ with salt and store.

**Algorithm 2:** CDP maintenance on creation of password $w_i = c_1, c_2, ..., c_m$

Castelluccia et al. consider the situation where the $n$-gram database is leaked a single time. They show that if the database is leaked when 5 million passwords have been created at $\gamma = 1e{-}6$, an average of 1.3 bits of Shannon entropy are leaked per password. They conclude that the guessing entropy remains high.

In addition, they note that when few passwords have been created, a leaked database allows an adversary to see which $n$-grams are not present in any password, limiting the search space. To prevent this, they begin by executing the noise-addition step of their algorithm $k$ times. Then, for the first $k$ passwords, noise is not added. After $k$ passwords have been created, the system follows all steps of Algorithm 2 for the remaining passwords.

## 2.2 Password strength estimator

Castelluccia et al. estimate the strength of a new password $w_i = c_1, c_2, ..., c_m$ according to the following algorithm:

**for** $k = 1, ..., m$ **do**
  $P(c_k | c_{k-n+1}, ..., c_{k-1}) = \frac{count_{i-1}(c_{k-n+1}, ...c_k)}{\sum_{x \in \Sigma} count_{i-1}(c_{k-n+1}, ..., c_{k-1}, x)}$;
**end**
$P(w_i) = \prod_{k=0}^{m} P(c_k | c_{k-n+1}, ..., c_{k-1})$;
Classify $w_i$ as "strong" if $P(w_i) < 2^{-20}$ and "weak" otherwise;

**Algorithm 3:** CDP password strength estimator

Castellucia et al. chose $2^{-20}$ as the threshold for determining strength because it guarantees an expected brute-force attack time of at least $2^{20}$ guesses.

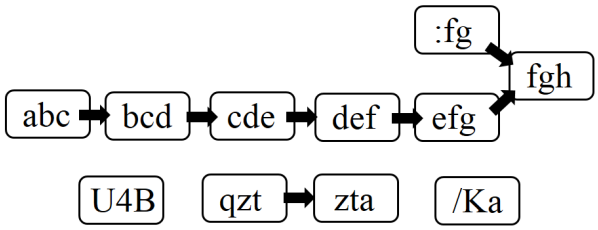


Figure 2: De Bruijn graph

### 2.3 Vulnerability

Although Castelluccia et al. proved security against a single leak of the database, under our security model, we must analyze the case where the database is leaked at multiple time steps. Specifically, suppose the adversary obtained $s_i$ and $s_{i+k}$ for some values $i$ and $k$. By taking the difference between these two, the adversary could calculate the $n$-gram counts for the $k$ passwords and $k$ rounds of noise. That is, the adversary can create the exact situation Castelluccia et al. attempted to prevent by inserting noise in bulk before any passwords were created.

For example, if $k = 1$, the adversary can calculate $z_i$, which is non-zero only for the $n$-grams from $w_i$ and the set of $n$-grams for which noise was added. We will let this collection of $n$-grams be denoted by $N$. N is dependent on $n$, $\gamma$, and the size of the alphabet $\Sigma$. The adversary can recreate $w_i$ by forming a De Bruijn graph from $N$.

**Definition 1** (De Bruijn graph)**.** *A De Bruijn graph is formed from a set of n-grams N in the following way:*

*i. a vertex is created for each n-gram in N.*

*ii. a directed edge is added from n-gram $u = x_1, x_2, ..., x_n$ to n-gram $v = y_1, y_2, ..., y_n$ if $x_k = y_{k-1}$ for $k = 2...n$.*

The adversary can build a De Bruijn graph $G$ from $N$ and search $G$ for paths of appropriate length for a password. Since $N$ contains all $n$-grams from $w_i$, one of these paths will be $w_i$. With parameters $n = 5$ and $\gamma = 5e{-}6$, which were chosen by Castellucia et al., it is extremely likely that the only paths of the appropriate length are $w_i$ and substrings of $w_i$.

In experiments with these parameters and randomly generated passwords (containing any combination of upper and lowercase characters, digits, and special symbols), a 2.6GHz laptop with 8GB of RAM was able to reconstruct each password in less than 10 seconds. Figure 3 shows a visualization of one of these experiments. Note that there are 36,708 loose nodes in this De Bruijn graph, and that the path of 12 vertices represents the password of length 16.

## 3 New scheme

The key to the attack presented above is that the adversary knows that $N$ contains the $n$-grams from $w_i$ and the set of $n$-grams for which noise was added. Therefore, the number of paths of

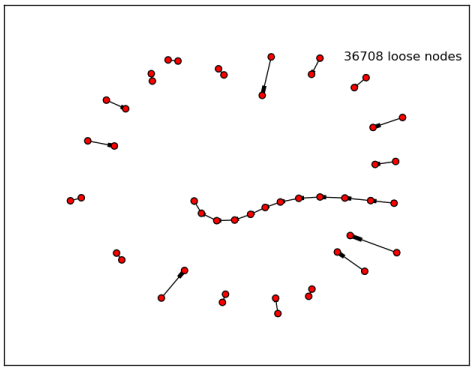


Figure 3: Visualization of an attack against the CDP meter

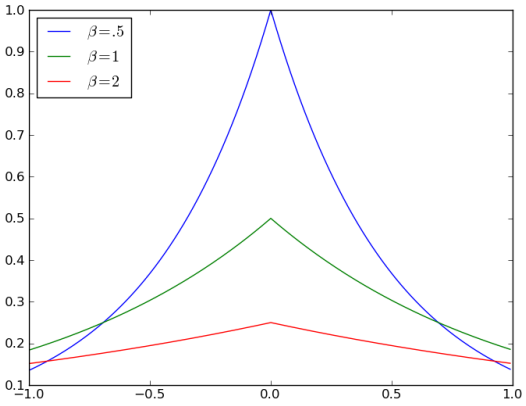


Figure 4: Probability density function for the Laplace distribution

the appropriate length is likely to be small with the chosen parameters. From this observation, we consider a modified scheme for maintaining the database of $n$-gram counts.

The specific modification we consider is inspired by differential privacy [4]. Let $A$ be an algorithm that takes as input a database containing information about a group of individuals. Then $A$ is said to be differentially private if the distribution of the output of $A$ does not change by much if a single individual from the database is replaced. One commonly used approach to constructing differentially private algorithms is to add noise drawn from a Laplace distribution.

**Definition 2** (Laplace distribution)**.** *The Laplace distribution with scale* $\beta$*, Laplace(*$\beta$*), has probability density function* $f(x) = \frac{1}{2\beta} exp(\frac{-|x|}{\beta})$.

As the goal of our security model is similar to that of differential privacy, we use Laplace noise in our password meter scheme. Thus, in our scheme, when password $w_i = c_1, c_2, ..., c_m$ is created in the system, the database is updated according to Algorithm 4.

**for** $k = 1, ..., m-n+1$ : **do**
  $count_i(c_k, ..., c_{k+n-1}) = count_{i-1}(c_k, ..., c_{k+n-1}) + 1$
**end**
**foreach** *n-gram* $x_1, ...x_n$ **do**
  $count_i(x_1, ...x_n) = count_i(x_1, ...x_n) + Laplace(\beta)$;
  **if** $count_i(x_1, ...x_n) < 0$ **then**
    $count_i(x_1, ...x_n) = 0$;
  **end**
**end**
Hash $w$ with salt and store.

**Algorithm 4:** Maintenance on creation of password $w_i = c_1, c_2, ..., c_m$ under new scheme

Passwords are then classified as "strong" or "weak" by comparing the estimated probability of the password to some threshold. The probability is calculated as in the CDP meter (Algorithm 3).

# 4 Analytical framework

In considering attacks against our scheme, we assume that passwords are chosen independently[1] from some distribution, and that the adversary knows that distribution. Note that implementation of our scheme does not require knowledge of the distribution, even though our analysis does.

**Proposition 1.** *If passwords are independent, then for cracking $w_i$, all database information except for $z_i$ can be ignored.*

*Proof.* When $w_i$ is added, the only change to the database is $z_i$, and $z_i$ is determined by $\beta$ and $w_i$. Thus, if passwords are independent, then $z_i$ is independent from all other passwords and changes to the database. Therefore, for cracking $w_i$, the rest of the database can be ignored. □

We also assume that there is a range of acceptable passwords lengths and that the adversary knows this range. For concreteness, we assume passwords must be at least 8 characters and no more than 16 characters long. We use $n$-grams of length $n = 3$, and we assume that passwords are composed of any combination of letters, digits, and special characters, for a total of 94 character options.

## 4.1 Attacks against small $\beta$

Depending on the value used for $\beta$ in our scheme, it is possible for an adversary to crack $w_i$ using a similar method to that described in Section 2.3. This modified attack works as follows:

- Based on the value of $\beta$, the adversary chooses some threshold $\tau$.
- The adversary keeps the set $N$ of $n$-grams whose count increases by at least $\tau$ according to $z_i$.

[1] If the passwords are not independent, the adversary's job is easier. For example, if the adversary knows that all passwords are identical (but doesn't know what they are), he can average the noisy counts to get (on average) a more accurate set of counts.

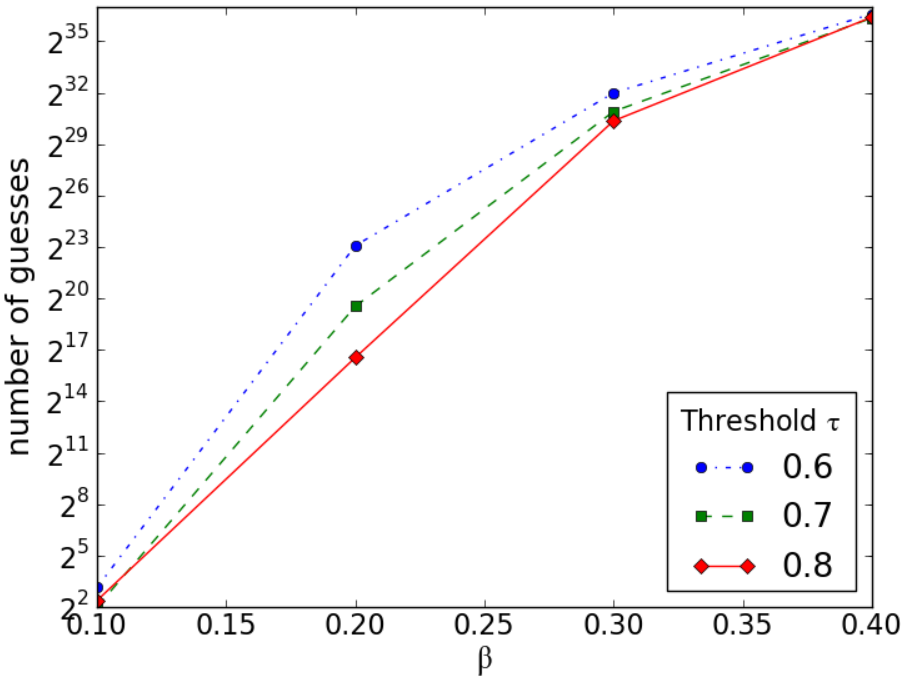


Figure 5: Number of guesses to have at least 50% chance of cracking a single password

- The adversary builds a De Bruijn graph $G$ from $N$ and searches for a path of appropriate length.
- If all $n$-grams from $w_i$ are in $N$, the adversary will find the password. Otherwise, the adversary will exhaust all paths and try again when $w_{i+1}$ is added.

The values of $\beta$ and $\tau$ determine the average number of paths the adversary must search through during each iteration, as well as the probability that all $n$-grams from $w_i$ are in $N$. These two factors determine the expected total number of guesses for an adversary to crack a password from the system. Figure 5 shows the number of guesses for the adversary to have at least a 50% chance of cracking a password of length 16. The probability that all $n$-grams from $w_i$ are present in $N$ determines the number of iterations required to have at least a 50% chance of at least one iteration succeeding. The number of iterations is then multiplied by the number of guesses per iteration.

It is important to note that while the number of guesses at each noise level does not change much with the choice of threshold, the total time to execute the attack is also dependent on waiting for new passwords to be created. Thus, if two attacks use the same number of guesses, but one uses fewer iterations, that attack will be more efficient. Figure 6 depicts the number of iterations and number of guesses per iteration for different threshold values at $\beta = 0.4$.

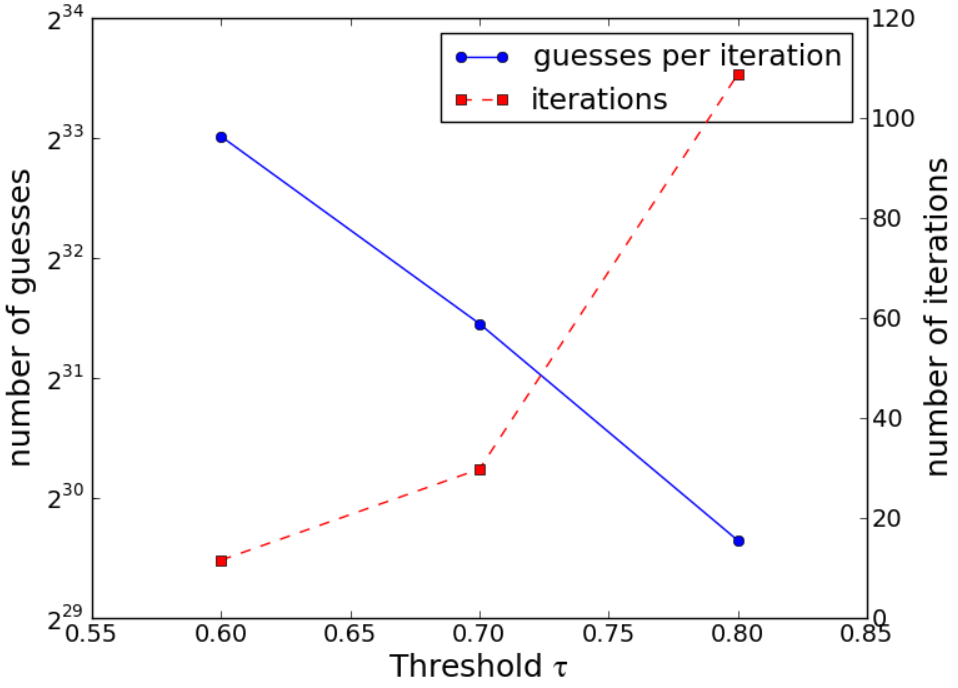


Figure 6: Number of iterations and guesses per iteration for $\beta = 0.4$

## 4.2 Theoretical bound on attacks

Under the assumption of independence of passwords, if the adversary has no additional information about a password $w_i$, the best attack is to guess passwords in decreasing order of probability. We use the following notation: $s_i$ is the internal state of the password meter after $w_i$ has been added (note that $s_i$ depends on Algorithm 4); $z_i = s_i - s_{i-1}$ is the change in state when $w_i$ is added; $z_i(g)$ is the change in count of $n$-gram $g = g_1, ..., g_n$ when $w_i$ is added;$p(w)$ is the probability that $w$ is chosen as a password; $p_{(j)}$ is the $j$th probability in sorted order; $p(w|z)$ is the conditional probability that $w$ is chosen as a password, given changes $z$; $p_{(j),z}$ is the $j$th probability in sorted order, given changes $z$.

From Proposition 1, all advantage in cracking $w_i$ with access to the database is gained from $z_i$. Whether or not an adversary has access to a database, we can bound the adversary's attack power. We now bound this advantage and eliminate subscripts $i$ for simplicity.

**Lemma 1.** *Suppose passwords are chosen independently from a distribution $p$.*

*i. Let $T$ be the number of guesses for the adversary to crack the password given the assumptions above. Then for any attack without access to a database, $E[T] \geq \sum j p_{(j)}$.*

*ii. Let $T_z$ be the number of guesses for the adversary to crack the password, given access to database changes $z$. Then for any attack, $E[T_z] \geq \sum j p_{(j),z}$.*

**Proposition 2.** *Let $z$ be any database. If there exists $c > 0$ such that, for all strings $w$, $p(w|z) \geq p(w) \cdot c$, then $p_{(i),z} \geq p_{(i)} \cdot c$ and $E[T_z] \geq E[T] \cdot c$.*

We can now bound the advantage in cracking $w$ that an adversary gains through $z$.

We begin by calculating $p(w|z)$ using Bayes' rule. Since under our scheme (Algorithm 4), each $z(g)$ is a continuous random variable, we use the probability density function $f(x) = \frac{1}{2\beta} exp(\frac{-|x|}{\beta})$ and let $f(z) = \prod_{n\text{-grams } g} z(g)$.

Thus we have

$$p(w|z) = \frac{f(z|w)}{f(z)} \cdot p(w) = \frac{f(z|w)}{\sum_{\text{strings } y} p(y) \cdot f(z|y)} \cdot p(w).$$

Plugging in the probability density function, this is equal to

$$\frac{\prod_{g \notin w} \exp(-|z(g)|/\beta) \prod_{g \in w} \exp(-|z(g)-1|/\beta)}{\sum_{\text{strings } y} p(y) \cdot \prod_{g \notin y} \exp(-|z(g)|/\beta) \prod_{g \in y} \exp(-|z(g)-1|/\beta)} \cdot p(w),$$

which then simplifies to

$$p(w) \cdot \left[ \sum_{y} p(y) \prod_{\substack{g \notin w, \\ g \in y}} \exp\left( \frac{|z(g)| - |z(g)-1|}{\beta} \right) \prod_{\substack{g \in w, \\ g \notin y}} \exp\left( \frac{|z(g)-1| - |z(g)|}{\beta} \right) \right]^{-1}.$$

Each term of each product is at most $\exp(1/\beta)$. This maximum occurs when $z(g) \geq 1$ for the first product and when $z(g) \leq 0$ for the second.

**Lemma 2.** *With noise parameter $\beta$, maximum password length $l$, and database changes $z$ for the set of all $n$-grams, $p(w|z) \geq p(w) \cdot \exp(1/\beta)^{-2(l-n+1)}$*

With our parameters of $l = 16$ and $n = 3$, Lemma 2 gives us

$$p(w|z) \geq p(w) \cdot \exp(1/\beta)^{-28}.$$

Therefore, we have a bound on the advantage to the adversary:

$$E[T_z] \geq E[T] \cdot exp(1/\beta)^{-28}.$$

Note that this is a loose upper bound on the advantage: The real reduction in attack time does not depend solely on this bound, but on a combination of the changes to all of the probabilities. In most cases, the ratio between the conditional and prior probabilities will be greater than $\exp(1/\beta)^{-2(l-n+1)}$. Empirical results suggest a much smaller advantage, indicating that a tighter analysis on the bound may be possible.

## 4.3 Empirical results

Figure 7 shows empirical results for the expected number of guesses from known password distributions, with and without the leaked database. These known distributions contain a relatively small

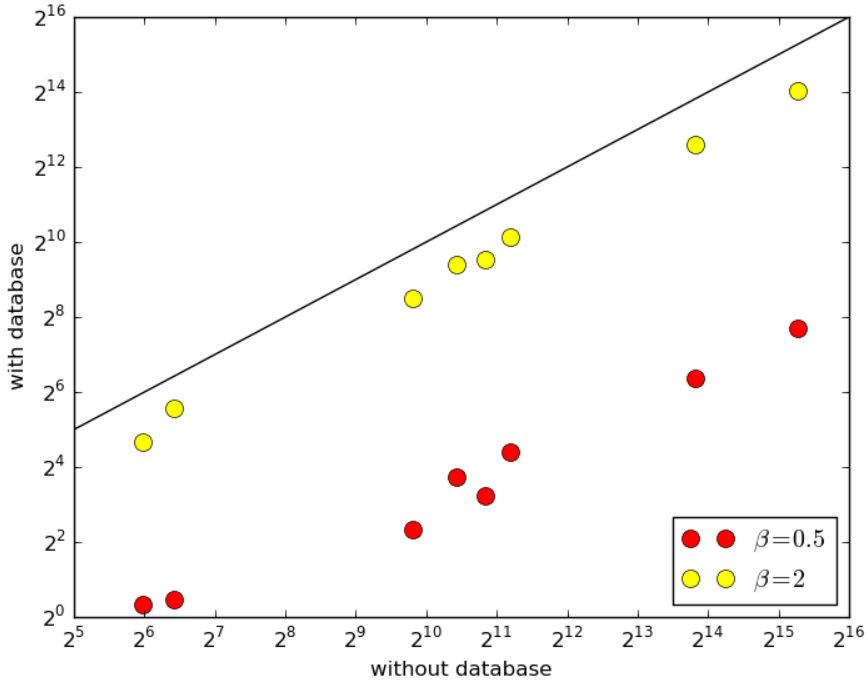


Figure 7: Empirical average number of guesses with and without database; each data point represents a single distribution

number of passwords each, chosen from publicly leaked password sets from www.phpbb.com and www.faithwriters.com. With $\beta = 0.5$, access to the database reduces the number of guesses by about a factor of 200, while a noise level of $\beta = 2$ only reduces the number of guesses to about 2/5 of its starting value. Both $\beta$ values seem to be approximately parallel to the $x = y$ line (in black). Because the password distributions contain few passwords, it is unclear how these values will trend for larger distributions.

There is also a gap between these empirical results and the theoretical bounds. This is to be expected, since (as discussed above) the theoretical bounds are loose.

## 4.4 Accuracy

In addition to being secure against leakage, it is important that any password meter be accurate. Suppose a user inputs a string that has a low probability of being chosen as a password, and yet the password meter gives a rating of "weak." If this occurs often, users become frustrated, and the usability of the password meter suffers. If, on the other hand, the password meter gives a rating of "strong" to commonly chosen passwords (that are in fact weak), then these weak passwords continue to be chosen, and the security of the system suffers.

To test the accuracy of our scheme at $\beta = 0.5$, we used a leaked and publicly available password set from www.faithwriters.com. The password set contains 4,035 passwords of length 8-16, along with the number of times each password was selected (ranging from 1 to 16). We began by filling

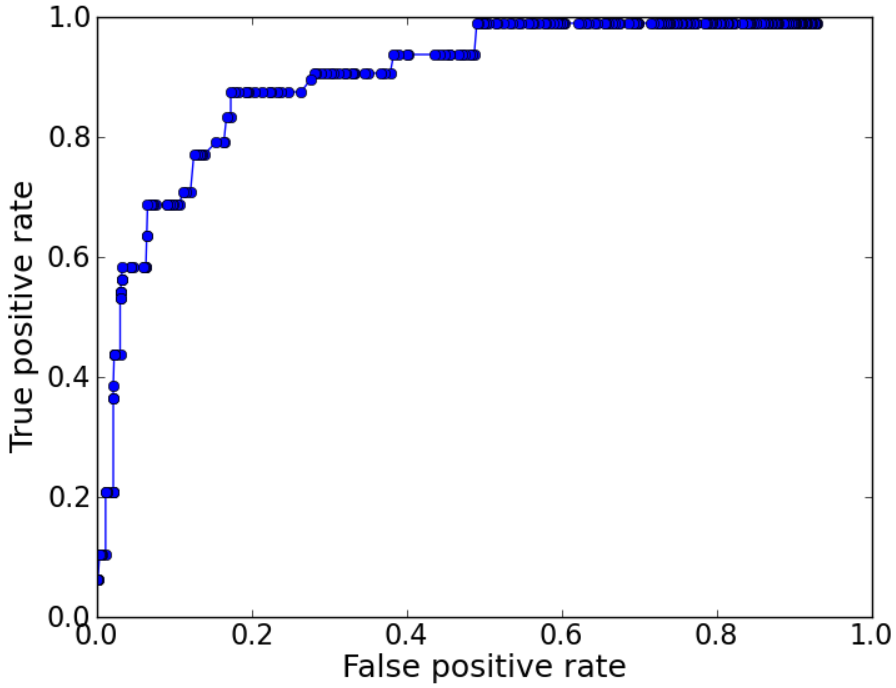


Figure 8: ROC curve for different threshold values

the database with 3-gram counts for 50,000 passwords chosen from this distribution. We then calculated the estimated probability of 1,000 of the passwords using true 3-gram counts and using noisy 3-gram counts. We took the estimated probabilities from the true counts to be the true probabilities, as Castellucia et al. showed good accuracy for true counts, and set the threshold to $2^{-20}$, as in Castellucia et al.

The threshold for classification based on estimated probabilities from noisy counts was flexible and determined the balance between false "strong" and false "weak" ratings. We gave each threshold a penalty for each false rating, with false "strong" ratings having more weight than false "weak" ratings to emphasize the importance of security. False "strong" ratings gave a penalty of 10, while false "weak" ratings gave a penalty of 1. Figure 8 shows the ROC curve for different classification threshold values, and Figure 9 shows the classification results using the threshold with the lowest penalty. (Two passwords had lower probabilities falling outside the range of Figure 9.)

Note that the estimated probabilities from noisy counts are consistently much lower than the estimated probabilities from true counts. This suggests that it may be possible to improve estimates through careful reduction of this bias.

# 5 Effects of meters on password distribution

We have not found any consideration in the literature of how password meters change the distribution of passwords (and thus the difficulty of an attack). Without intervention by a password

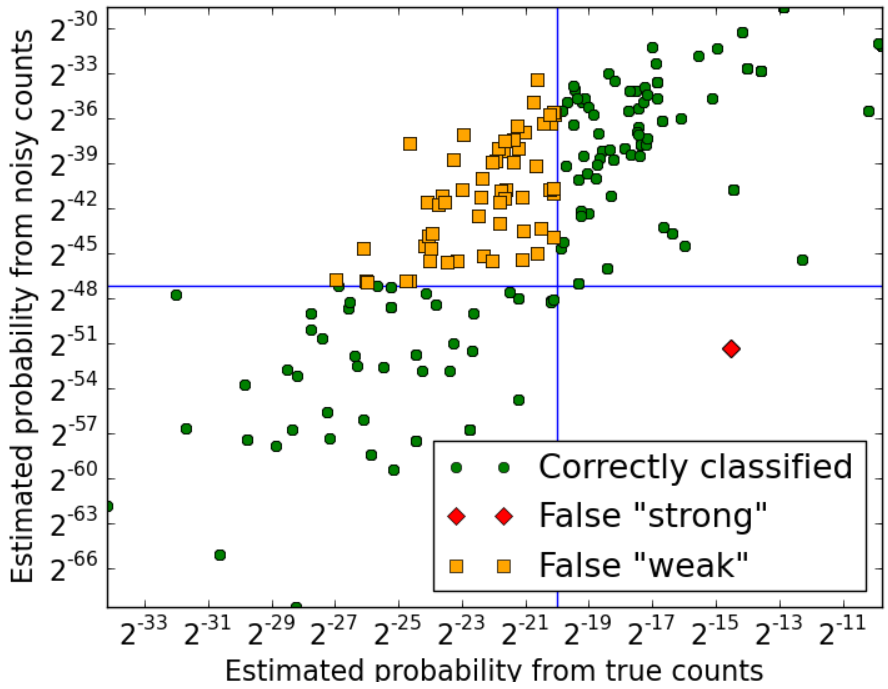


Figure 9: Classification with true vs. noisy counts; the vertical and horizontal lines indicate the thresholds for estimated probability from true counts and noisy counts, respectively.

meter, we assume that users choose passwords according to some distribution. A password meter attempts to push users away from the common, easily-guessed passwords from this distribution. The result is a new distribution, but what effect does this new distribution have on the security of passwords?

Assume we have a password meter that is completely accurate, according to the prior distribution on passwords; the meter gives a score of "weak" for all passwords above some probability and a score of "strong" for all passwords below that probability. Furthermore, assume that any password scored as "weak" is either prohibited from the system or that users always voluntarily avoid choosing it.

In the naïve model for distribution adjustment, we assume that the weak passwords are removed and the remaining probabilities are re-normalized. That is, if the $t$ most probable passwords are prohibited, the new sorted probabilities are $p'_{(j)} = \frac{p_{(j+t)}}{\sum_{k>t} p_{(k)}}$. This is equivalent to assuming that if the selected password is weak, the user selects a different password, again according to the prior distribution, repeating until the selected password is strong. If the password meter is public, an adversary will know which passwords are weak and thus will not guess them.

We now consider three cases of prior distributions and the effect the password meter has on the security of the posterior distribution.

- Case 1: Suppose the prior distribution was such that $p_{(j)} = c^{j-1}(1-c)$ for some constant $0 < c < 1$. If the $t$ most probable passwords are all eliminated, the posterior distribution

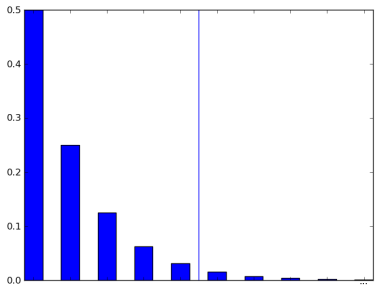


Figure 10: Case 1 prior

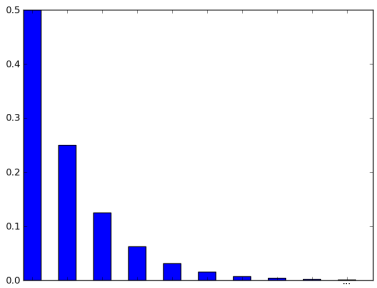


Figure 11: Case 1 posterior

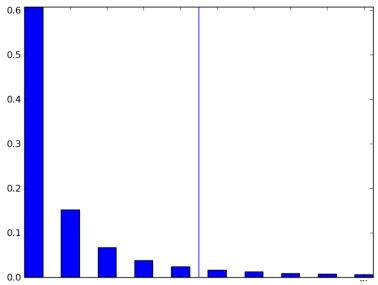


Figure 12: Case 2 prior

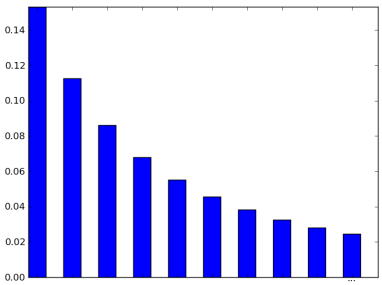


Figure 13: Case 2 posterior

retains this same shape, and the adversary's expected number of guesses remains unchanged.

- Case 2: Suppose the prior distribution was such that $p_{(j)} = \frac{6}{j^2\pi^2}$. If the $t$ most probable passwords are all eliminated, the adversary's expected number of guesses increases.
- Case 3: Suppose the prior distribution was such that $p_{(1)} = p(2) = ... = p(t) = \frac{1}{t+1}$ and $p_{(j)} = c^{j-t-1}(1-c)$ for some constant $0 < c < 1$ for $i > t$. If the $t$ most probable passwords are all eliminated, the adversary's expected number of guesses decreases.

Thus, under the naïve model, the effect of the password meter on the adversary's cracking ability is completely dependent on the prior distribution.

On the other hand, when a password is weak, the user may be more likely to simply modify the password until it is strong (e.g., changing "password" to "password1"). It is possible that this modification would result in a significantly different password distribution. We are unsure how to model this situation, as it seems to be grounded in psychology and the choices people make.

In the case of adaptive password meters, the composition of existing passwords is constantly changing, which affects the passwords that are considered weak. Thus the distribution for future passwords changes with each new password.

We believe that understanding the effect of password meters on password distributions is an important topic for future work.

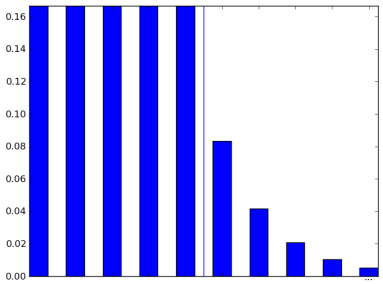


Figure 14: Case 3 prior

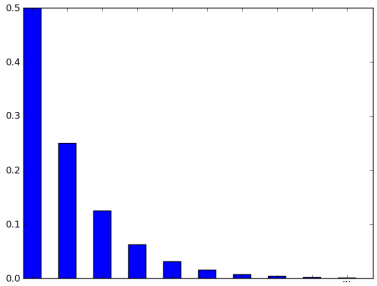


Figure 15: Case 3 posterior

# 6 Conclusions

Adaptive password meters can outperform the accuracy of standard password meters, but using them requires thinking carefully about their security. Although our upper bounds on an adversary's advantage against our scheme are too high for real-world application, our empirical results give a more optimistic view of its security. The empirical results suggest that there is room for improvement in our analysis.

It seems likely that there are better attacks against our scheme than the one discussed in Section 4.1, as well as other schemes for adaptive password meters that might be more secure.

We also feel it is important to further consider the long-term effects of password meters on password distributions and security.

## REFERENCES


[1] William E. Burr, Donna F. Dodson, Timothy W. Polk, Elaine M. Newton, and Ray A. Perlner. Electronic authentication guideline: NIST special publication 800-63. (2006).

[2] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. Adaptive password-strength meters from Markov models. In *NDSS*, 2012.

[3] Chris Davies and Ravi Ganesan. BApasswd: A New Proactive Password Checker. 1993.

[4] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[5] Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N. Rothblum, and Sergey Yekhanin. Pan-private streaming algorithms. In *ICS*, pages 66–80, 2010.

[6] Serge Egelman, Andreas Sotirakopoulos, Ildar Muslukhov, Konstantin Beznosov, and Cormac Herley. Does my password go up to eleven?: the impact of password meters on password selection. In *CHI*, pages 2379–2388, 2013.

[7] Chlotia Posey Garrison. Encouraging good passwords. In *InfoSecCD*, pages 109–112, 2006.

[8] Cormac Herley and Paul C. van Oorschot. A research agenda acknowledging the persistence of passwords. *IEEE Security & Privacy*, 10(1):28–36, 2012.

[9] Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio Lopez. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *IEEE Symposium on Security and Privacy*, pages 523–537, 2012.

[10] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman. Of passwords and people: measuring the effect of password-composition policies. In *CHI*, pages 2595–2604, 2011.

[11] Cynthia Kuo, Sasha Romanosky, and Lorrie Faith Cranor. Human selection of mnemonic phrase-based passwords. In *SOUPS*, pages 67–78, 2006.

[12] Michael D. Leonhard and V. N. Venkatakrishnan. A comparative study of three random password generators. In *IEEE Conference on Electro/Information Technology*, pages 227–232, May 2007.

[13] Wanli Ma, John Campbell, Dat Tran, and Dale Kleeman. Password entropy and password quality. In *NSS*, pages 583–587, 2010.

[14] Arvind Narayanan and Vitaly Shmatikov. Fast dictionary attacks on passwords using time-space tradeoff. In *ACM Conference on Computer and Communications Security*, pages 364–372, 2005.

[15] Stuart Schechter, Cormac Herley, and Michael Mitzenmacher. Popularity is everything: a new approach to protecting passwords from statistical-guessing attacks. In *USENIX HotSec*, 2010.

[16] Richard Shay, Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Blase Ur, Timothy Vidas, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Correct horse battery staple: exploring the usability of system-assigned passphrases. In *SOUPS*, page 7, 2012.

[17] Richard Shay, Saranga Komanduri, Patrick Gage Kelley, Pedro Giovanni Leon, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Encountering stronger password requirements: user attitudes and behaviors. In *SOUPS*, 2010.

[18] Blase Ur, Patrick Gage Kelley, Saranga Komanduri, and Joel Lee. How does your password measure up? The effect of strength meters on password creation. In *USENIX Security*, page 5, 2012.

[19] Matt Weir, Sudhir Aggarwal, Michael P. Collins, and Henry Stern. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *ACM Conference on Computer and Communications Security*, pages 162–175, 2010.

[20] Matt Weir, Sudhir Aggarwal, Breno de Medeiros, and Bill Glodek. Password cracking using probabilistic context-free grammars. In *IEEE Symposium on Security and Privacy*, pages 391–405, 2009.

[21] Jeff Jianxin Yan, Alan F. Blackwell, Ross J. Anderson, and Alasdair Grant. Password memorability and security: Empirical results. *IEEE Security & Privacy*, 2(5):25–31, 2004.