Towards Quantum Large-Scale Password Guessing on Real-World Distributions

Markus Dürmuth[1], Maximilian Golla[2], Philipp Markert[1], Alexander May[1], and Lars Schlieper[1]

[1]Ruhr University Bochum, [2]Max Planck Institute for Security and Privacy
{markus.duermuth,philipp.markert,alex.may,lars.schlieper}@rub.de,
maximilian.golla@csp.mpg.de

Abstract. Password-based authentication is a central tool for end-user security. As part of this, password hashing is used to ensure the security of passwords at rest. If quantum computers become available at sufficient size, they are able to significantly speed up the computation of preimages of hash functions. Using Grover's algorithm, at most, a square-root speedup can be achieved, and thus it is expected that quantum password guessing also admits a square-root speedup. However, password inputs are not uniformly distributed but highly biased. Moreover, typical password attacks do not only compromise a random user's password but address a large fraction of all users' passwords within a database of millions of users.

In this work, we study those quantum large-scale password guessing attacks for the first time. In comparison to classical attacks, we still gain a square-root speedup in the quantum setting when attacking a constant fraction of all passwords, even considering strongly biased password distributions as they appear in real-world password breaches. We verify the accuracy of our theoretical predictions using the LinkedIn leak and derive specific recommendations for password hashing and password security for a quantum computer era.

Keywords: Passwords · Quantum Computing · Hash Function · Zipf.

1 Introduction

Despite its significant weaknesses, password-based authentication continues to be widely used. An average user may have hundreds of password-protected online accounts [38,51], needs to enter a password for decrypting and booting the computer, or accessing the Wi-Fi and VPN. Reasons for the continued use of passwords include their intuitive and simple usage and an "ecosystem" of coping strategies (such as password reuse [16,21,37] and choosing passwords with predictable patterns [45,46,47]) to work around limitations of passwords.

Password hashing is a central building block for ensuring the security of passwords at rest: for passwords stored in databases or hard disk encryption, it is recommended to use a salted, iterated, and memory-hard cryptographic hash

function [4]. Hashed passwords can be attacked by an offline guessing attack: an attacker generates password candidates that a human user likely chooses, in decreasing order of likelihood. Then, the attacker hashes the password candidates and compares them against the stored password hash, revealing if the tested password candidate matches the real password [19,34]. Considering the heavily skewed distribution of human password choice, consumer-grade hardware such as gaming GPUs can already pose a threat for password security [15,23], except for very slow password hashes such as Argon2i [3].

With the availability of quantum computers, many cryptographic primitives are at high risk [26,29,32,33,39,40]. Specifically, popular public-key cryptography that is in use today and based on the factorization or discrete logarithm problem, such as the RSA cryptosystem, can be broken by Shor's algorithm [43]. In symmetric cryptography, the situation is different. For many symmetric ciphers, it is assumed that the only impact in a quantum setting is the Grover square-root speedup, which can be countered by doubling the key length. Similarly, Grover's algorithm gives a square-root speedup for finding preimages of cryptographic hash functions. A straight-forward application of Grover's algorithm to *uniformly distributed* N passwords displays a similar speedup: quantum computers can find preimages for hashed passwords faster (in $\sqrt{N}$ steps) than traditional computers [12]. For more realistic *human-chosen* passwords that are not uniformly distributed but highly skewed, it is unknown if a square-root speedup can also be realized.

Our Contribution. In this work, we investigate the impact of quantum computers on password guessing. To the best of our knowledge, our work is the first to consider realistic (non-uniform) password distributions, as well as quantum attackers guessing passwords for more than a single password hash. Our contribution is three-fold:

1. We investigate how an attacker equipped with a quantum computer (of sufficient size) can use Grover's algorithm to guess non-uniformly distributed *human-chosen* passwords. We realize square-root speedups in two different attack scenarios: targeting a single (fixed) user and large-scale attacks targeting a whole password database at once. Both scenarios are commonly found in password research and password security practice.
2. As a central tool, we use a Zipf distribution to model human password choices. We provide analytic bounds for the required number of evaluations of the password hashing function that holds both in the classical and the quantum world, which may be of independent interest.
3. We use the well-known LinkedIn password leak to check the accuracy of our Zipf model, verifying the applicability of our results. We then discuss the implications and consequences for real-world practices. Even though quantum computers of the required size for our attack might not yet be available today, we discuss the possible consequences of our results on the required increase in password strength. Moreover, we address possible solutions based on alternative password hash functions.

Overall, we believe that our work allows for a better understanding of the long-term impact of quantum computers on the security of password hashing and provides a first step in taking appropriate steps to mitigate problems.

Related Work. A first step towards modeling user-chosen password distributions with Zipf's law has been done by Malone and Maher [35]. They conclude that Zipf's law may not allow for an exact yet good approximation of the frequencies in which users choose passwords. Bonneau [6] and Wang et al. [49] came to a similar conclusion. The latter also refined accuracy by independently modeling less and more frequent passwords. A subsequent work by Wang et al. [48] revised the approach even further, also utilizing it to measure password dataset security.

Corrigan-Gibs et al. [12] discussed how the security of hashed passwords changes in the presence of quantum computers. For this purpose, they assume a 10-character password that is randomly chosen from the space of all 95 printable ASCII characters. By applying Grover's algorithm [25], the security of such a password reduces from $95^{10} \approx 2^{66}$ to only $\sqrt{95^{10}} \approx \sqrt{2^{66}} = 2^{33}$. Hence, Corrigan-Gibs et al. conclude that quantum computers would put hashed passwords at risk, which are currently perceived as secure.

Moreover, the quantum scenario also allows for identity authentication protocols with improved privacy and security properties [13]. However, these protocols require all communicating parties to have quantum computer access, which is not the scenario we are considering.

2 Password Guessing

In the following section, we provide a brief introduction to password guessing and the real-world password datasets we are using. Moreover, we describe how human password choice can be approximated by Zipf's law and introduce our overall attack setting.

2.1 Threat Model

Passwords are typically stored in salted hashed form, i. e., instead of storing the password pwd in plain text, one stores $(s, h(pwd|s)$ for a hash function h and a random salt s. Due to the strong bias and the resulting low entropy of human-chosen passwords, effective attacks against such password hashes are *guessing attacks*. Here an attacker enumerates password candidates by their likelihood, and for each candidate tests if it is the correct password. In an *online guessing attack*, the attacker tests potential password candidates directly with the service provider. These attacks can be reasonably mitigated on the server-side, e. g., by rate-limiting. A more pronounced threat is *offline guessing attacks*, where the attacker is in possession of the database with the password hashes and is thus only limited by the available computational resources, as the correctness of the passwords can be tested locally. This offline scenario is the threat model we consider in the remainder of this work.

In order to minimize the number of hashing operations for the guessing attack, an attacker will try to guess the most frequent passwords first. For our work, we consider the strongest threat model possible in form of a *perfect knowledge attacker* that uses the actual password distribution and guesses passwords in order of decreasing probability.
We distinguish between the following two attack scenarios:

Scenario A: Fixed User Attack. This is the simplest attack scenario, where the attacker targets a *certain fixed password hash* by testing a list of common passwords in decreasing probability. A typical example for Scenario A is when an attacker is interested in a single user in a password file (e. g., law enforcement). Another important use case is (software-based) *hard disk encryption*, where the content of the disk is encrypted using a symmetric key. This key needs to be stored on the hard disk as well (unless supported by specialized hardware), protected by a password. To this end, the password is the input to a key derivation function (KDF), which typically uses hash functions (e. g., PBKDF2) or similar cryptographic constructions to derive a cryptographic key from the password. This key is not directly used to encrypt the hard disk but to encrypt a keystore, which facilitates key management, e. g., to allow for changing the password or for multiple users. Such a keystore construction enables the attacker to verify a password candidate by feeding the candidate password through the KDF, decrypting the keystore, and testing if it has the correct form.

Scenario B: Large-Scale Attack. An often even more relevant attack scenario is when an attacker tries to recover *a certain fraction of all of the password hashes* from a large number of accounts. A typical example for this Scenario B are breached databases, like the one from LinkedIn (as described in Section 2.2), of which 98 % of the passwords were recovered as they were hashed using SHA-1 and no salt [23]. In order to optimize the attack, the attacker will guess passwords based on decreasing likelihood and test each password for all hashes before continuing with the next guess. The recovered passwords can then be used in a variety of ways: they can be further weaponized in a *credential stuffing* attack [36,37], in a targeted password guessing attack [50], they can be monetized on black markets [17], or used for further illegal activities such as sending spam emails. Moreover, they can be of interest for password security research [4,16,20,35,47] or for enthusiasts that crack passwords out of fun or as a sort of competition [14,23].

2.2 Password Datasets

When analyzing human-chosen passwords, one must consider that password choice is contextual and influenced by many factors [2,18,38]. While it is difficult to adjust for all factors, we will analyze the impact of these influencing factors and describe our used comparison metrics in Section 2.3.

An overview of our used datasets that are described in the following is given in Table 1. We selected the datasets to allow for easy verification and to generate reproducible results based on publicly available data.

Table 1: Evaluated Password Datasets

Name	Service	Year	Policy	# Accounts
LinkedIn	Social Network	2012	6+	160.8 million
RockYou	Social Games	2009	5+	32.6 million
000Webhost	Web Hosting	2016	6+ [a-Z][0-9]	15.3 million

To reason about the strength of a password distribution considering a *perfect knowledge* attacker, we use the partial guessing entropy (α-guesswork) G_α for $\alpha = 0.25$ as described by Bonneau [6]. Our datasets are:

- **LinkedIn**: The social networking website LinkedIn was hacked in June 2012. It consists of a database dump of approx. 163 million unsalted SHA-1 hashes. We use a 98.68 % recovered plaintext version of the leak, as we expect the bias introduced by ignoring 1.32 % of (presumably strong) passwords to be low. We include LinkedIn because we consider those passwords to be a reasonable candidate for *medium*-strong passwords ($\tilde{G}_{0.25} = 19$ bits).
- **RockYou**: This is a well-established leak used extensively in previous work. The 32 million plaintext passwords leaked from the RockYou web service in December 2009 via an SQL injection attack. Its passwords are considered relatively *weak* ($\tilde{G}_{0.25} = 16$ bits).
- **000Webhost**: This leak occurred in October 2015 and contains 15 million plaintext passwords from a free web space provider. We include this leak because of its enforcement of a stricter password composition policy, which results in a different password distribution containing relatively *strong* passwords ($\tilde{G}_{0.25} = 21$ bits).

2.3 Approximating Human Password Choice by a Zipf Distribution

Real-world datasets allow researchers to study human password choice, and it has been realized that real-world distributions $\mathcal{D}_{\mathrm{Pw}}$ typically follow a Zipf distribution. The Zipf distribution was originally formulated in the context of quantitative linguistics based on the frequency of words in English text [53] but has since been shown to be a good model for human password choice as well. It is well documented that real-world password distributions (roughly) follow a Zipf distribution (cf. Wang et al. [48,49], Malone and Maher [35], and Bonneau [6]).

Let $P = \{pwd_1, \ldots, pwd_N\}$ be an ordered set of N passwords, and let $s \geq 0$. We define the generalized harmonic number as

$$H_s(N) := \sum_{i=1}^{N} \frac{1}{i^s} \ .$$

Let X be a $\mathrm{Zipf}_{N,s}$-distributed random variable over P with *steepness parameter* s (the larger s, the steeper). Then

$$p_i := \mathbb{P}[X = pwd_i] = \frac{1}{H_s(N)} \cdot \frac{1}{i^s} \ ,$$

where $H_s(N)$ normalizes the distribution. Throughout the paper, we use that partial sums of ZIPF probabilities can be easily expressed as

$$\sum_{i=1}^{k} p_i = \frac{1}{H_s(N)} \sum_{i=1}^{k} \frac{1}{i^s} = \frac{H_s(k)}{H_s(N)}. \tag{1}$$

Notice that for $s = 0$ we have $H_0(N) = N$ and $p_i = \mathbb{P}[X = pwd_i] = \frac{1}{N}$. Thus, $\text{ZIPF}_{N,0}$ is the uniform distribution on all passwords. For $0 \leq s < 1$, the following lemma shows that $H_s(N) \leq N^{1-s}$, and therefore $p_1 \geq N^{s-1}$.

For example, for steepness $s = \frac{3}{4}$ the most likely password pwd_1 has probability $p_1 \geq N^{-\frac{1}{4}} \gg N^{-1}$. This already implies that on expectation we can identify a user with password pwd_1 with at most $N^{\frac{1}{4}}$ tries.

Lemma 1. *For $0 \leq s < 1$ we have*

$$\frac{N^{1-s}}{1-s} - \frac{1}{1-s} < H_s(N) \leq \frac{N^{1-s}}{1-s}.$$

In particular, $H_s(N) = \Theta(N^{1-s})$.

Proof.

$$\begin{aligned} \frac{N^{1-s}}{1-s} - \frac{1}{1-s} &= \int_1^N i^{-s}\, di < \sum_{x=1}^{N} \frac{1}{i^s} = H_s(N) \\ &\leq 1 + \int_1^N i^{-s}\, di = 1 + \frac{N^{1-s}}{1-s} - \frac{1}{1-s} \leq \frac{N^{1-s}}{1-s}\,. \end{aligned}$$

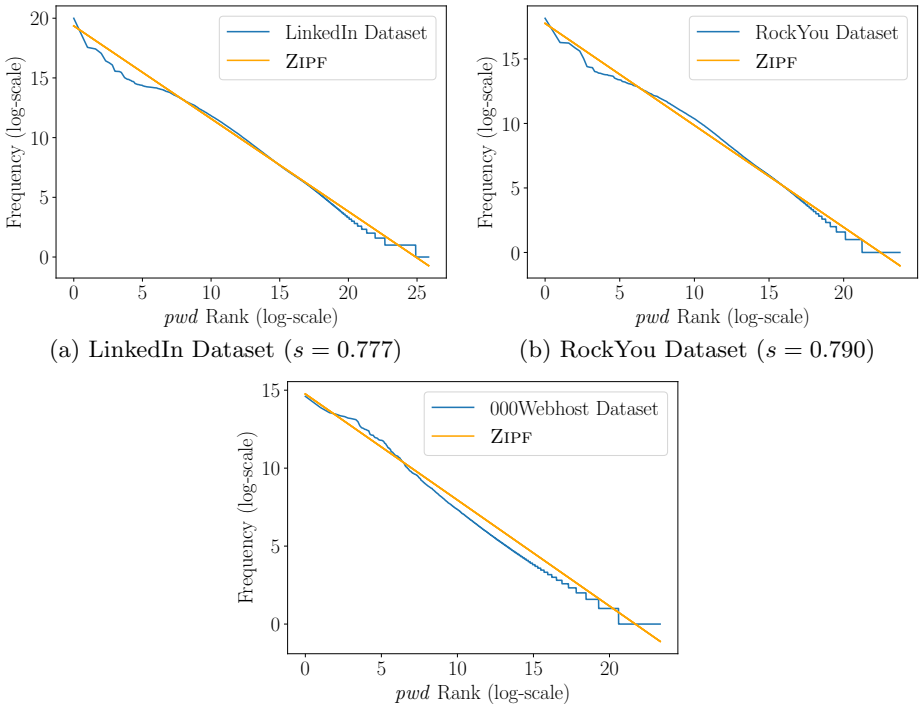
□

In order to determine typical steepness values s in practical settings, let us provide an explicit approximation using the ZIPF distribution. As a first example, we take the LinkedIn database [23], with roughly $160 \cdot 10^6 \sim 2^{27}$ users and $N \sim 60 \cdot 10^6 \sim 2^{26}$ different passwords $P = \{pwd_1, \ldots, pwd_N\}$. We choose s so that $\text{ZIPF}_{N,s}$ is the best approximation of the password-distribution $\mathcal{D}_{\text{Pw}}$ of the database. For this, we use the coefficient of determination R^2 from Definition 1 between $\text{ZIPF}_{N,s}$ and the password distribution of the database, and select s so that R^2 is maximized. This is the case for $s \sim 0.777$ with a coefficient of determination of $R^2 = 0.781$. A log/log-scaled plot of a $\text{ZIPF}_{N,0.777}$ distribution and $\mathcal{D}_{\text{Pw}}$ can be seen in Figure 1a. The results of analog calculations for the RockYou and 000Webhost leak are shown in Figure 1b and 1c. In the following, we will use the LinkedIn leak as our main dataset, since it is by far the largest publicly available dataset that includes *medium*-strong passwords ($\tilde{G}_{0.25} = 19$ bits).

Definition 1 (Coefficient of Determination (R^2)). *We define the coefficient of determination between two datasets $D = \{y_1, \ldots, y_n\}$ and $\hat{D} = \{\hat{y_1}, \ldots, \hat{y_n}\}$ as*

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2}\,,$$

where $\overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$ is the mean of the dataset D.



(a) LinkedIn Dataset ($s = 0.777$) (b) RockYou Dataset ($s = 0.790$)

(c) 000Webhost Dataset ($s = 0.680$)

Fig. 1: Approximation of password leaks by the Zipf distribution.

In the following, we show that the complexity of password guessing attacks in the fixed user case can be described by a random variable X distributed according to some Zipf distribution. Therefore, we are interested in the expectation of X. The following Lemma 2 shows that the expectation is linear in the number N of passwords.

Lemma 2. *Let X be a $\text{Zipf}_{N,s}$ distributed random variable with $0 \leq s < 1$. Then*

$$\mathbb{E}[X] = \frac{1-s}{2-s} \cdot N\left(1 \pm o\left(1\right)\right) \ .$$

Proof. By definition of expectation, we have

$$\mathbb{E}[X] = \sum_{i=1}^{N} i \cdot \mathbb{P}[X = pwd_i] = \sum_{i=1}^{N} i \cdot \frac{1}{H_s(N)} \cdot \frac{1}{i^s} = \frac{1}{H_s(N)} \sum_{i=1}^{N} \frac{1}{i^{s-1}} = \frac{H_{s-1}(N)}{H_s(N)} \ .$$

Using Lemma 1, we show the upper and lower bound for $\mathbb{E}[X]$, starting with the upper bound.

$$\begin{aligned}\mathbb{E}[X] = \frac{H_{s-1}(N)}{H_s(N)} &\leq \frac{\frac{N^{2-s}}{2-s}}{\frac{N^{1-s}}{1-s} - \frac{1}{1-s}} = \frac{1-s}{2-s} \cdot N \cdot \left(\frac{N^{1-s}}{N^{1-s}-1}\right) \\ &= \frac{1-s}{2-s} \cdot N \cdot \left(1 + \frac{1}{N^{1-s}-1}\right) = \frac{1-s}{2-s} \cdot N \cdot \left(1 + o\left(1\right)\right) \ .\end{aligned}$$

Analogously, we derive the lower bound

$$\mathbb{E}[X] = \frac{H_{s-1}(N)}{H_s(N)} \geq \frac{\frac{N^{2-s}}{2-s} - \frac{1}{2-s}}{\frac{N^{1-s}}{1-s}} = \frac{1-s}{2-s} \cdot N \cdot \left(\frac{N^{1-s}-1}{N^{1-s}}\right)$$
$$= \frac{1-s}{2-s} \cdot N \cdot \left(1 - \frac{1}{N^{1-s}}\right) = \frac{1-s}{2-s} \cdot N \cdot (1 - o(1)) \ .$$

□

2.4 Password Guessing Scenario

Let $U = \{u_1, \ldots, u_w\}$ be a set of w users and $P = \{pwd_1, \ldots, pwd_N\}$ a set of N passwords. We denote by $g : U \mapsto \{1, \ldots, N\}$ a function that maps each $u \in U$ to its password index, i. e., u has password $pwd_{g(u)}$. Further, we define for each user u a random salt s_u.

We denote by L a leaked database of $|U|$ triples $(u, s_u, h(pwd_{g(u)}|s_u))$, where h is a cryptographic hash function. In other words, the leaked database L reveals for each user u its salt s_u and a salted hash of its password.

As in the previous Section 2.3, the set of password follows a $\textsc{Zipf}_{N,s}$ distribution. Hence, a uniformly random user $u \in U$ has $pwd_i \in P$ with probability

$$p_i := \mathbb{P}_{u \in U}[g(u) = i] = \frac{1}{H_s(N)} \cdot \frac{1}{i^s} \ .$$

Let us define a *password verification function* $V : L \times P \mapsto \{0, 1\}$ as

$$V_{u,s_u,h(pwd_{g(u)}|s_u)}(pwd) := \begin{cases} 1 & \text{if } h(pwd_{g(u)}|s_u) = h(pwd|s_u), \\ 0 & \text{else} \end{cases} \ . \tag{2}$$

By the definition of V, a password guess pwd is correctly linked to user u with entry $(u, s_u, h(pwd_{f(u)}|s_u)) \in L$ if pwd has the correct salted hash value. We call every user for which our guesser finds a correctly linked password *compromised*.

We check the correctness of our password guesses via function evaluation of V. Notice that each evaluation requires a hash evaluation of h, which is our unit cost measure. Thus, we define the *average cost* of password guessing as

$$C := \frac{\#\text{Evaluations of } h}{\#\text{Compromised Users}} \ . \tag{3}$$

Notice that our cost measure is independent of the underlying hash function. In Section 6.2, we discuss the effects of taking the run time of hash function evaluations into account.

In the following, we show that for both scenarios described in Section 2.1 (i. e., attacking a fixed user or large-scale attacks), the average cost of a quantum attacker is only the square-root of the average cost of a classical attacker.

2.5 Quantum Password Guessing

While previous applications of quantum algorithms to the problem of password guessing considered uniform password distribution [12], we generalize to a Zipf distribution, which more accurately captures real-world password distributions. As in previous work [12], we consider error-free quantum computations [1,10,31].

When attacking a fixed user u (Scenario A), the key advantage of quantum computations is that we check the correctness of *all passwords* on u in a parallel superposition. In the large-scale Scenario B, first studied quantumly in our work, the key advantage of quantum computation is that we can check the correctness of a single password on *all users* in parallel.

Grover's Algorithm. The key to many square-root speedups in quantum computation is Grover's algorithm [25] and its generalizations [7,8].

Theorem 1 (Grover [7,8,25]). *Let Ω be a finite search space with solutions $T \subseteq \Omega$. Let $f : \Omega \mapsto \{0,1\}$ be an efficiently computable* target function *with $f(x) = 1$ iff $x \in T$.*

1. *In the single solution case $|T| \leq 1$, Grover's algorithm computes the unique solution $x \in T$ with $c \cdot \sqrt{|\Omega|}$ f-queries, where $c = \frac{\pi}{4} \cdot \left(1 + o\left(\frac{|T|}{|\Omega|}\right)\right) \sim 0.785$, or outputs FAIL if $|T| = 0$.*
2. *In the general case, Grover's algorithm computes a random solution $x \in T$ after $\mathcal{O}\left(\sqrt{\frac{|\Omega|}{|T|}}\right)$ f-queries, or outputs FAIL if $|T| = 0$.*

For illustrative purposes, let us first consider the case of uniformly distributed passwords.

Fixed User Attack. Let L be our database with entries $\ell_u = (u, s_u, h(pwd_{f(u)}|s_u))$. Consider some fixed ℓ_u, and take our password verification function $V : L \times P \to \{0,1\}$ from Equation (2). We define the Grover target function

$$f_{\ell_u} : P \to \{0,1\}, \quad pwd \mapsto V(\ell_u, pwd). \tag{4}$$

Since a unique password $pwd \in P$ verifies correctly for user u (unless we find collisions in h), we are in the single solution case $|T| = 1$ of Theorem 1. Thus, Grover's algorithm recovers the correct password with $c \cdot \sqrt{|P|} \sim 0.785 \cdot \sqrt{N}$ hash evaluations.

Large-Scale Attack. We check for all users in $U = \{u_1, \ldots u_w\}$ the correctness of a single fixed password pwd. To this end, define the Grover function

$$f_{pwd} : U \to \{0,1\}, \quad u \to V(\ell_u, pwd). \tag{5}$$

Since many users may use the same password pwd, we are in the general case of Theorem 1. Let $T \subseteq U$ be the number of users that share pwd. Then Grover's algorithm recovers a random user $u \in T$ within $\mathcal{O}(\sqrt{\frac{|U|}{|T|}})$ hash evaluations. In the following sections, we also take the effects of our Zipf distribution into account.

3 Scenario A: Fixed User Attack

Classical. First, we study how the ZIPF distribution affects an optimal classical attacker targeting a fixed user u with leaked data $\ell_u = (u, s_u, h(pwd_{f(u)}|s_u)) \in L$.

A classical attacker's optimal strategy is to try passwords $pwd_1, pwd_2, \ldots$ in order of decreasing probability. Let X be a random variable for the number of hash evaluation. Then the attacker succeeds with a single hash evaluation with probability $\mathbb{P}[X = 1] = p_1 = \mathbb{P}[u \text{ has password } pwd_1]$. In general, X is $\text{ZIPF}_{N,s}$-distributed. By Lemma 2 and neglecting low order terms, the expected number of hash evaluations is

$$\mathbb{E}[X] = \sum_{i=1}^{N} i \cdot p_i = \frac{1-s}{2-s} \cdot N. \tag{6}$$

This implies that for the uniform distribution with $s = 0$, we need to test on expectation half of the passwords. For the typical value of $s = \frac{3}{4}$ from Figures 1a and 1b, on expectation it suffices to try only $\frac{1}{5}N$ passwords.

Quantum. In the quantum setting, we use Grover's theorem (Theorem 1) with target function $f_{\ell_u} : P \to \{0,1\}, f_{\ell_u}(pwd) := V(\ell, pwd)$ from Equation (4). Moreover, we split our search space P at the mean $\mu := \mathbb{E}[X]$ of our $\text{ZIPF}_{N,s}$ distribution from Equation (6). Our quantum attack first checks whether our desired password is in $P_1 = \{pwd_1, \ldots, pwd_\mu\}$. By Theorem 1, this first check can be performed with $c\sqrt{\mu}$ hash evaluations and succeeds with probability

$$p_1 + \ldots + p_\mu = \frac{1}{H_s(N)} \sum_{i=1}^{\mu} \frac{1}{i^s} = \frac{H_s(\mu)}{H_s(N)}.$$

Using Lemma 2, we have to perform the second check on the remaining search space $P \setminus P_1$ with probability at most $1 - \frac{H_s(\mu)}{H_s(N)} \leq \sqrt{1-s}$. In total the number of hash evaluations is upper bounded by

$$c\left(\sqrt{\mu} + \sqrt{1-s} \cdot \sqrt{N-\mu}\right) = 2c\sqrt{\mathbb{E}[X]}.$$

Thus, up to a factor of at most $2c \leq 1.6$ our quantum algorithm achieves the square-root cost of the optimal classical cost from Equation (6).

Remark 1. A result of similar quality can be achieved by using the *Amplitude Amplification* technique of Brassard et al. [8]. However, our Grover-based approach benefits from its simplicity, since in Amplitude Amplification, we have to create a superposition over all passwords weighted by their $\text{ZIPF}_{N,s}$-distribution. This creates some technical difficulties and unnecessary overhead.

4 Scenario B: Large-Scale Attack

Let us now look at the scenario where an attacker wants to compromise just a single user with a weak password. In some attack scenarios, this may already provide an attacker access to an infrastructure, e. g., in a company.

4.1 Scenario B.1: Attacking a Single (and All) Weakest User(s)

Classical. To identify a single weakest user, the optimal classical approach is to try the most likely password pwd_1 with success probability p_1 on random users. This takes expected running time $\frac{1}{p_1}$. If our passwords followed a uniform distribution, then this attack still takes expected time $\frac{1}{p_1} = N$. However, in the more general case of a $\textsc{Zipf}_{N,s}$ distribution by Lemma 1 we have

$$\frac{1}{p_1} = H_s(N) \leq \frac{N^{1-s}}{1-s}.$$

This implies, for our typical value $s = \frac{3}{4}$ from Figures 1a and 1b, that an attacker finds a user with a weakest password in time with at most $4N^{\frac{1}{4}}$ hash evaluations.

Quantum. In the quantum setting, we use Grover's algorithm over a superposition of *all users* to identify a user with password pwd_1.

Let $U = \{u_1, \ldots, u_w\}$ be the user set, and L be a our leaked database with entry ℓ_u for user u. We use the Grover function $f_{pwd_1} : U \to \{0,1\}$, $u \to V(\ell_u, pwd_1)$, as defined in Equation (5).

Let T be the set of users with weakest password pwd_1. Then on expectation $|T| = p_1|U|$. An application of Theorem 1 shows that we find a random user from T in time

$$\mathcal{O}\left(\sqrt{\frac{|U|}{|T|}}\right) = \mathcal{O}\left(\sqrt{\frac{1}{p_1}}\right) = \mathcal{O}(\sqrt{H_s(N)}) = \mathcal{O}(N^{\frac{1-s}{2}}).$$

For $s = \frac{3}{4}$, this implies that we quantumly compromise a user with the weakest password within only $\mathcal{O}(N^{\frac{1}{8}})$ hash function evaluations.

Using our cost function from Equation (3), we obtain average classical cost $\mathcal{O}(N^{1-s})$ respectively quantum cost $\mathcal{O}(N^{\frac{1-s}{2}})$ for compromising a single user with weakest password. In the following, we show that with the same average cost per user, we can also compromise all users with password pwd_1.

All Weakest Users. Classically, we simply test for all $|U|$ users the validity of password pwd_1, resulting in expected $|T| = p_1|U|$ compromised users. This implies average cost $C = \frac{|U|}{p_1|U|} = H_s(N) = \mathcal{O}(N^{1-s})$.

Quantumly, we use the aforementioned Grover algorithm until we find $|T| = p_1|U|$ different users with password pwd_1. Using coupon collector, this takes $|T| \cdot \ln|T|$ runs of the above algorithm. Omitting the low order $\ln|T|$-term, we obtain average cost

$$\frac{|T| \cdot \mathcal{O}\left(\sqrt{\frac{|U|}{|T|}}\right)}{|T|} = \mathcal{O}\left(\sqrt{\frac{1}{p_1}}\right) = \mathcal{O}(N^{\frac{1-s}{2}}),$$

giving us again the desired square-root speedup.

4.2 Scenario B.2: Attacking a Constant Fraction of all Users

Finally, we want to generalize the techniques from the previous Section 4.1 to large-scale adversaries that try to recover a constant fraction c of all users. As an illustrating example, we use $c = 10\,\%$ respectively $c = 50\,\%$ of the users. In a nutshell, in both the classical and quantum setting an attacker recovers those users that use the k weakest passwords $pwd_1, \ldots, pwd_k$ with probabilities $p_1, \ldots, p_k$. We set k such that we obtain the desired c-fraction of all users. Using Equation (1) and Lemma 1 we obtain

$$\sum_{i=1}^{k} p_i = \frac{H_s(k)}{H_s(N)} \approx \left(\frac{k}{N}\right)^{1-s} \stackrel{!}{=} c,$$

where $\approx$ suppresses an $(1 + o(1))$-factor. This means we can solve the above relation in k by setting

$$k = c^{\frac{1}{1-s}} N. \tag{7}$$

Classical. Consider an attacker that in a first pass tries on all $|U|$ users password pwd_1, thereby recovering $p_1|U|$ users. In a second pass, the attacker tries on all remaining $|U| - p_1|U|$ users pwd_2, recovering $p_2|U|$ users, etc. The attacker stops on identifying at least $c|U|$ many user passwords.

The amount of hash function evaluations per pass is clearly upper bounded by $|U|$, and lower bounded by $(1 - c)|U| = \Omega(|U|)$. Thus, the attacker recovers $c|U|$ passwords in total time $k \cdot \Theta(|U|) = \Theta(c^{\frac{1}{1-s}} N|U|)$. This gives us an average cost per password of

$$C = \frac{\Theta(c^{\frac{1}{1-s}} N|U|)}{c|U|} = \Theta\left(c^{\frac{s}{1-s}} N\right). \tag{8}$$

Let us briefly ignore the small constant hidden in the Θ-notion. For the uniform distribution with $s = 0$, the average cost per password is N, as one would expect. For the typical value $s = \frac{3}{4}$, we obtain the average cost $c^3 N$. This means that $\textsc{Zipf}_{N,\frac{3}{4}}$ gives us a speedup of factor c^{-3} per compromised password. Thus, for $c = 0.5$ we obtain a speedup of 8, and for $c = 0.1$ we even obtain a speedup factor of 1000 over the uniform distribution.

Quantum. From Section 4.1 we know that we can quantumly recover all $p_i|U|$ users with password pwd_i in time $\mathcal{O}(p_i|U| \cdot \sqrt{\frac{|U|}{p_i|U|}}) = \mathcal{O}(\sqrt{p_i}|U|)$. Hence, for all k passwords $pwd_1, \ldots, pwd_k$ we need a total time of

$$\begin{aligned}\mathcal{O}\left(\sum_{i=1}^{k} \sqrt{p_i}|U|\right) &= \mathcal{O}\left(\frac{1}{H_s(N)} \sum_{i=1}^{k} i^{-\frac{s}{2}}|U|\right) = \mathcal{O}\left(N^{\frac{s-1}{2}} \cdot k^{1-\frac{s}{2}} \cdot |U|\right) \\ &= \mathcal{O}\left(N^{\frac{s-1}{2}} \cdot (c^{\frac{1}{1-s}} N)^{1-\frac{s}{2}} \cdot |U|\right) = \mathcal{O}\left(c^{1+\frac{s}{2(1-s)}} N^{\frac{1}{2}} \cdot |U|\right).\end{aligned}$$

Since we recover $c|U|$ passwords, this implies an average cost per password of

$$\mathcal{O}\left(\sqrt{c^{\frac{s}{1-s}}N}\right),$$

i. e., the square-root of the classical cost from Equation (8).

5 Real-World Impact

Let us now check the accuracy of our theoretical predictions for the Zipf distribution from Sections 3 and 4, when applied to a real-world password leak distribution $\mathcal{D}_{\mathrm{Pw}}$. As described in Section 2.2, we take the LinkedIn database for this purpose because it is a good example for a medium-strong real-world distribution.

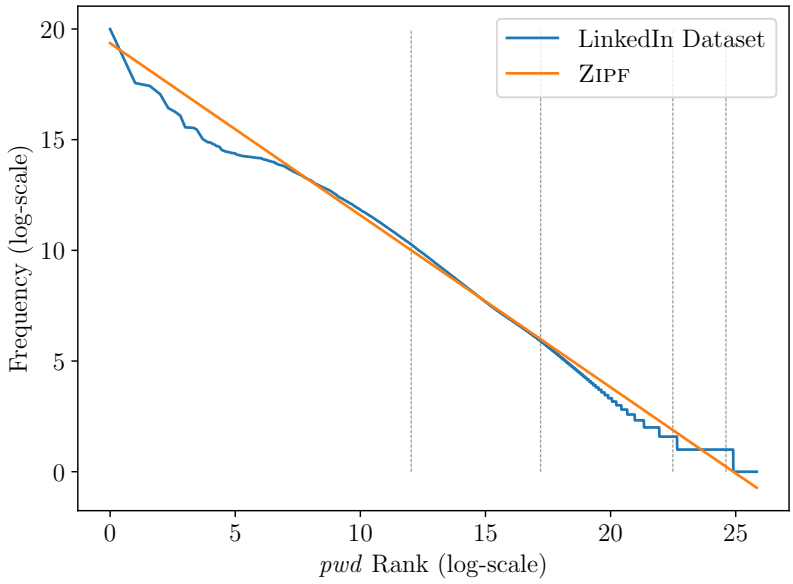


Fig. 2: The best approximation of the LinkedIn password leak by a Zipf distribution ($s = 0.777$). The dotted lines correspond to a fraction of 10 %, 25 %, 50 %, and 75 % of all users.

Notice that we only take the distribution from LinkedIn, see also Figure 2, but otherwise stay in our attack model with *salted hashes* (which is not the case for the LinkedIn dataset).

Scenario A: Fixed User Attack. Table 2 provides an overview of our results of the real-world Scenario A. We see that $\textsc{Zipf}_{N,0.777}$ quite accurately approximates the number of required hash evaluations for the real-world distribution $\mathcal{D}_{\mathrm{Pw}}$ – within a factor of 1.3 classically and $1.15 \approx \sqrt{1.3}$ quantumly.

Table 2: Scenario A: Fixed User Attack. Required hash evaluations for finding a fixed user's password (using the approaches of Section 3 and Grover with $c = 1$).

Distribution	**Required Hash Evaluations**	
	Classical	**Quantum**
Uniform $\sim$ $\textsc{Zipf}_{N,0}$	30 000 000	7 750
$\textsc{Zipf}_{N,0.777}$	11 100 000	5 560
LinkedIn $\mathcal{D}_{\mathrm{Pw}}$	14 600 000	6 430

Compared to a quantum attacker for the *uniform distribution* ($\textsc{Zipf}_{N,0}$), our new quantum attacker with knowledge of the *human-chosen password distribution* ($\mathcal{D}_{\mathrm{Pw}}$) only requires 83 % of the hash evaluations.

It is also worth stressing the absolute numbers for $\mathcal{D}_{\mathrm{Pw}}$. While the classical attacker needs almost 15 million hash evaluations, the quantum attacker succeeds with roughly 6400 hash evaluations.

Table 3: Attack Scenario B: Large-Scale Attack. Required number of hash evaluations *per user* for compromising a constant fraction of all users (using the approaches of Section 4 and Grover with $c = 1$).

Setting	**Distribution**	**Required Hash Evaluations per User**				
		10 %	**25 %**	**50 %**	**75 %**	**100 %**
Classical	Uniform $\sim$ $\textsc{Zipf}_{N,0}$	57 100 000	52 600 000	45 100 000	37 500 000	30 000 000
	$\textsc{Zipf}_{N,0.777}$	33 300	473 000	3 430 000	8 800 000	11 100 000
	LinkedIn $\mathcal{D}_{\mathrm{Pw}}$	38 700	482 000	6 820 000	14 300 000	14 600 000
Quantum	Uniform $\sim$ $\textsc{Zipf}_{N,0}$	7 750	7 750	7 750	7 750	7 750
	$\textsc{Zipf}_{N,0.777}$	158	613	1 880	3 710	6 030
	LinkedIn $\mathcal{D}_{\mathrm{Pw}}$	181	622	2 520	4 640	6 380

Scenario B: Large-Scale Attack. Table 3 provides an overview of our results for large-scale attackers, when compromising a total user fraction of 10 %, 25 %, 50 %, 75 %, and 100 %.

The values predicted by the $\textsc{Zipf}_{N,0.777}$ distribution differ only by a factor of at most 2 from the values for the LinkedIn distribution $\mathcal{D}_{\mathrm{Pw}}$, again validating the accuracy of Zipf.

Notice that in comparison to Scenario A, the reduction of the absolute numbers through the quantum square-root speedup is even more significant in this large-scale scenario. Within the weakest 10 % of the users, we require only about 180 hash evaluations per user on average. When attacking 50 % of the users, we still require just 2500 evaluations per user. Only if we address the full database, the average grows to 6400, matching the analysis from Scenario A.

Let us put the 10 % weakest user scenario into perspective. Assume that, both classically and quantumly, a hash evaluation takes about 1 second [3]. Then quantumly we would require only 180 seconds, i.e. 3 minutes, whereas a classical attacker would need, on average, about 10 hours per user.

6 Discussion

In light of our new results for the quantum setting, we now discuss recommendations for increasing password security in a post-quantum world.

6.1 Password Strength

A square-root speedup seems moderate from a cryptographic perspective and merely means doubling the cryptographic keys' length to achieve the same security level. However, from a user's perspective, this is much more dramatic. To bring this into perspective, let us consider an 8 character password *randomly* chosen from the alphabet of all 95 printable ASCII characters (lowercase, uppercase, digits, symbols) as the minimal level of security. If we want to achieve the same security in the quantum setting, users would need to remember 16 instead of 8 characters due to the square-root speedup-up from Grover's algorithm. While assigning random passwords to users is not recommendable from a usability perspective, studies showed that users are potentially able to remember random 8 character passwords [28,52,54]. Increasing this to 16 random characters crosses a threshold where we cannot expect users to be able to memorize such a password. This is the situation for randomly chosen passwords; let us now focus on human password choice.

As described earlier, users struggle and often fail to create secure passwords, resulting in a highly skewed distribution. This worsens the situation because we demonstrated in Section 5 that an attacker with knowledge of the password distribution is much more effective in guessing the passwords. Hence, if we want to achieve the security level as described above, users would need to increase the length of their password beyond 16 characters. While increasing the password length is recommended by current best practice policies, among others, NIST [24], they do not require more than 16 characters. This limit ensures that the created passwords are not only secure but also memorable [41,42,44]. Hence, we argue that for both random and user-chosen passwords, we are going to face the problem that we as humans simply can no longer memorize the passwords for all our accounts.

If we insist on solving the problem by increasing the password length, this seems only possible with password managers. Using them is already recommended nowadays, but their advantage of generating and storing a long, random string for each and every account becomes even more apparent in the quantum setting. Nevertheless, considering their low adoption rates and the goal in mind not to burden users, it is important to look at alternative solutions.

6.2 Password Hash Functions

So far, our cost measure used the number of hash function evaluations but ignored their individual run times. We did this to provide results independently from actual implementations. Still, for recommendations that tend to increase security, we remark on the importance of choosing suitable and future-proof password hashing functions. Currently widely deployed hash functions such as bcrypt, PBKDF2, or iterated versions of SHA-256 and SHA-512 [9,22], are based on the idea that the computing power of an adversary is limited. This also holds for attackers in the quantum setting.

Currently, the biggest limitations of quantum computers are the number of qubits that can be implemented as well as the time span they can keep their entangled states, i. e., their information. As of June 2021, the technology scales up to around 65 qubits [11], with predictions of multiple hundred in the foreseeable future. Thus, a short-term solution could be the use of an unusually long salt that does not add complexity but exploits the shortage of available qubits in a quantum computer. However, a far better approach is *memory-hard* password hashes [3,5], which require large amounts of memory to be efficiently computed. While memory-hard password hashes were originally intended to counter the massive computing power of ASICs, FPGAs, and GPUs, we believe that they are also effective in countering quantum computer-based attacks. Still, further research is necessary to focus on the quantum-hardness of memory-hard password hash functions.

6.3 Encrypted Passwords and Secret Salts

As an additional layer of protection NIST [24] recommends that service operators should "perform an additional iteration of a KDF using a salt value that is secret and known only to the verifier." This *secret salt*, sometimes also called *pepper*, needs to be stored separately from the password hashes and should ideally reside in a hardware security module (HSM) or similar protected device. Likewise, passwords could also be encrypted using a quantum-resistant authenticated encryption scheme like NTRU [27]. In both cases, the attacker would need to obtain or attack (using Grover's algorithm) the secret salt/encryption key first. Thus, these protection mechanisms are only applicable to scenarios where a remote device or rate-limited hardware component like a trusted platform module could be utilized to store the required high entropy key material.

In cases where an HSM is not available, one could use a different notion of a secret salt in the form of a random value that is not stored but needs to be rediscovered every time it is needed [30]. However, since the server itself has to brute-force this value for every authentication attempt, it has to be chosen from a smaller set than a stored salt. Hence, this countermeasure is only effective against a large-scale attacker that targets multiple hashes because the slowdown becomes significant if the individual but negligible brute-force attempts add up. Finally, a randomly chosen salt only gives a probabilistic guarantee, whereas iterated hash functions, as described in Section 6.2, ensure a fixed slowdown.

7 Conclusion

Motivated by the recent advancements in the field of quantum computers, we analyzed the potential impact of quantum computing on securely stored human-chosen passwords. We showed how a quantum computer-equipped attacker can take advantage of the bias in real-world password distributions and still gains a square-root speedup in the quantum world. We validated our theoretical Zipf modeling with a real-world distribution from LinkedIn. Our quantum speedup on real-world data leads to an already small number of 6400 hash evaluations for attacking a fixed user (Scenario A) and to a frightening number of less than 200 hash evaluations per user among the 10 % users with weakest passwords (Scenario B). Our results underline the necessity of new password protection mechanisms in a quantum world.

Acknowledgments. This research was supported by the research training group "Human Centered Systems Security" sponsored by the state of North Rhine-Westphalia and funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA – 390781972.

References

1. Aharonov, D., Ben-Or, M.: Fault-Tolerant Quantum Computation with Constant Error Rate. SIAM Journal on Computing **38**(4), 1207–1282 (Jul 2008)
2. Bailey, D.V., Dürmuth, M., Paar, C.: Statistics on Password Re-Use and Adaptive Strength for Financial Accounts. In: Security and Cryptography for Networks. pp. 218–235. SCN '14, Springer, Amalfi, Italy (Sep 2014)
3. Biryukov, A., Dinu, D., Khovratovich, D., Josefsson, S.: Argon2 Memory-Hard Function for Password Hashing and Proof-of-Work Applications. RFC 9106, RFC Editor (Sep 2021), https://tools.ietf.org/html/rfc9106
4. Blocki, J., Harsha, B., Zhou, S.: On the Economics of Offline Password Cracking. In: IEEE Symposium on Security and Privacy. pp. 35–53. SP '18, IEEE, San Francisco, California, USA (May 2018)
5. Boneh, D., Corrigan-Gibbs, H., Schechter, S.E.: Balloon Hashing: A Memory-Hard Function Providing Provable Protection Against Sequential Attacks. In: Advances in Cryptology – ASIACRYPT 2016, Part I. pp. 220–248. ASIACRYPT '16, Springer, Hanoi, Vietnam (Dec 2016)
6. Bonneau, J.: The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords. In: IEEE Symposium on Security and Privacy. pp. 538–552. SP '12, IEEE, San Jose, California, USA (May 2012)
7. Boyer, M., Brassard, G., Høyer, P., Tapp, A.: Tight Bounds on Quantum Searching. In: Workshop on Physics and Computation. pp. 493–506. PhysComp '96, Elsevier, Boston, Massachusetts, USA (Nov 1996)
8. Brassard, G., Høyer, P., Mosca, M., Tapp, A.: Quantum Amplitude Amplification and Estimation. Contemporary Mathematics **305**, 53–74 (Jun 2002)
9. Brewster, T.: Why You Shouldn't Panic About Dropbox Leaking 68 Million Passwords (Aug 2016), https://www.forbes.com/sites/thomasbrewster/2016/08/31/dropbox-hacked-but-its-not-that-bad/, as of October 1, 2021

10. Campbell, E.T., Terhal, B.M., Vuillot, C.: Roads Towards Fault-Tolerant Universal Quantum Computation. Nature **549**(7671), 172–179 (Sep 2017)
11. Cho, A.: IBM Promises 1000-Qubit Quantum Computer by 2023 (Sep 2020), https://www.sciencemag.org/news/2020/09/ibm-promises-1000-qubit-quantum-computer-milestone-2023, as of October 1, 2021
12. Corrigan-Gibbs, H., Wu, D.J., Boneh, D.: Quantum Operating Systems. In: Workshop on Hot Topics in Operating Systems. pp. 76–81. HotOS '17, ACM, Vancouver, British Columbia, Canada (May 2017)
13. Crawford, H., Atkin, S.: Quantum Authentication: Current and Future Research Directions. In: Who Are You?! Adventures in Authentication Workshop. pp. 1–5. WAY '19, USENIX, Santa Clara, California, USA (Aug 2019)
14. Croley ("Chick3nman"), S.: Abusing Password Reuse at Scale: Bcrypt and Beyond (Aug 2018), https://www.youtube.com/watch?v=5su3_Py8iMQ, as of October 1, 2021
15. Croley ("Chick3nman"), S.: NVIDIA GeForce RTX 3090 Hashcat Benchmarks (Nov 2020), https://gist.github.com/Chick3nman/e4fcee00cb6d82874dace72106d73fef, as of October 1, 2021
16. Das, A., Bonneau, J., Caesar, M., Borisov, N., Wang, X.: The Tangled Web of Password Reuse. In: Symposium on Network and Distributed System Security. NDSS '14, ISOC, San Diego, California, USA (Feb 2014)
17. Digital Shadows, Ltd.: From Exposure to Takeover: The 15 Billion Stolen Credentials Allowing Account Takeover (Jul 2020), https://resources.digitalshadows.com/whitepapers-and-reports/from-exposure-to-takeover, as of October 1, 2021
18. Florêncio, D., Herley, C.: A Large-Scale Study of Web Password Habits. In: The World Wide Web Conference. pp. 657–666. WWW '07, ACM, Banff, Alberta, Canada (May 2007)
19. Florêncio, D., Herley, C., Van Oorschot, P.C.: An Administrator's Guide to Internet Password Research. In: Large Installation System Administration Conference. pp. 44–61. LISA '14, USENIX, Seattle, Washington, USA (Nov 2014)
20. Golla, M., Dürmuth, M.: On the Accuracy of Password Strength Meters. In: ACM Conference on Computer and Communications Security. pp. 1567–1582. CCS '18, ACM, Toronto, Ontario, Canada (Oct 2018)
21. Golla, M., Wei, M., Hainline, J., Filipe, L., Dürmuth, M., Redmiles, E., Ur, B.: "What was that site doing with my Facebook password?" Designing Password-Reuse Notifications. In: ACM Conference on Computer and Communications Security. pp. 1549–1566. CCS '18, ACM, Toronto, Ontario, Canada (Oct 2018)
22. Goodin, D.: Once Seen as Bulletproof, 11 Million+ Ashley Madison Passwords Already Cracked (Sep 2015), https://arstechnica.com/information-technology/2015/09/once-seen-as-bulletproof-11-million-ashley-madison-passwords-already-cracked/, as of October 1, 2021
23. Gosney ("epixoip"), J.M.: How LinkedIn's Password Sloppiness Hurts Us All (Jun 2016), https://arstechnica.com/information-technology/2016/06/how-linkedins-password-sloppiness-hurts-us-all/, as of October 1, 2021
24. Grassi, P.A., Fenton, J.L., Burr, W.E.: Digital Identity Guidelines – Authentication and Lifecycle Management: NIST Special Publication 800-63B (Jun 2017)
25. Grover, L.K.: A Fast Quantum Mechanical Algorithm for Database Search. In: ACM Symposium on Theory of Computing. pp. 212–219. STOC '96, ACM, Philadelphia, Pennsylvania, USA (May 1996)
26. Häner, T., Roetteler, M., Svore, K.M.: Factoring Using $2n+2$ Qubits With Toffoli Based Modular Multiplication. Quantum Information and Computation **17**(7–8), 673–684 (Apr 2017)

27. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A Ring-Based Public Key Cryptosystem. In: Algorithmic Number Theory Symposium. pp. 267–288. ANTS '98, Springer, Portland, Oregon, USA (Oct 1998)
28. Huh, J.H., Kim, H., Bobba, R.B., Bashir, M.N., Beznosov, K.: On the Memorability of System-Generated PINs: Can Chunking Help? In: Symposium on Usable Privacy and Security. pp. 197–209. SOUPS '15, USENIX, Ottawa, Canada (Jul 2015)
29. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Breaking Symmetric Cryptosystems Using Quantum Period Finding. In: Advances in Cryptology – CRYPTO 2016, Part I. pp. 207–237. CRYPTO '16, Springer, Santa Barbara, California, USA (Aug 2016)
30. Kedem, G., Ishihara, Y.: Brute Force Attack on UNIX Passwords With SIMD Computer. In: USENIX Security Symposium. pp. 93–98. SSYM '99, USENIX, Washington, District of Columbia, USA (Aug 1999)
31. Knill, E., Laflamme, R., Zurek, W.H.: Resilient Quantum Computation: Error Models and Thresholds. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **454**(1969), 365–384 (Jan 1998)
32. Kuwakado, H., Morii, M.: Security on the Quantum-Type Even-Mansour Cipher. In: International Symposium on Information Theory and Its Applications. pp. 312–316. ISITA '12, IEEE, Honolulu, Hawaii, USA (Oct 2012)
33. Leander, G., May, A.: Grover Meets Simon – Quantumly Attacking the FX-Construction. In: Advances in Cryptology – ASIACRYPT 2017, Part II. pp. 161–178. ASIACRYPT '17, Springer, Hong Kong, China (Dec 2017)
34. Liu, E., Nakanishi, A., Golla, M., Cash, D., Ur, B.: Reasoning Analytically About Password-Cracking Software. In: IEEE Symposium on Security and Privacy. pp. 380–397. SP '19, IEEE, San Francisco, California, USA (May 2019)
35. Malone, D., Maher, K.: Investigating the Distribution of Password Choices. In: The World Wide Web Conference. pp. 301–310. WWW '12, ACM, Lyon, France (Apr 2012)
36. Mirian, A., DeBlasio, J., Savage, S., Voelker, G.M., Thomas, K.: Hack for Hire: Exploring the Emerging Market for Account Hijacking. In: The World Wide Web Conference. pp. 1279–1289. WWW '19, ACM, San Francisco, California, USA (May 2019)
37. Pal, B., Daniel, T., Chatterjee, R., Ristenpart, T.: Beyond Credential Stuffing: Password Similarity Models Using Neural Networks. In: IEEE Symposium on Security and Privacy. pp. 866–883. SP '19, IEEE, San Francisco, California, USA (May 2019)
38. Pearman, S., Thomas, J., Naeini, P.E., Habib, H., Bauer, L., Christin, N., Cranor, L.F., Egelman, S., Forget, A.: Let's Go in for a Closer Look: Observing Passwords in Their Natural Habitat. In: ACM Conference on Computer and Communications Security. pp. 295–310. CCS '17, ACM, Dallas, Texas, USA (Oct 2017)
39. Roetteler, M., Naehrig, M., Svore, K.M., Lauter, K.E.: Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms. In: Advances in Cryptology – ASIACRYPT 2017, Part II. pp. 161–178. ASIACRYPT '17, Springer, Hong Kong, China (Dec 2017)
40. Roetteler, M., Svore, K.M.: Quantum Computing: Codebreaking and Beyond. IEEE Security & Privacy **16**(5), 22–36 (Oct 2018)
41. Shay, R., Bauer, L., Christin, N., Cranor, L.F., Forget, A., Komanduri, S., Mazurek, M.L., Melicher, W., Segreti, S.M., Ur, B.: A Spoonful of Sugar?: The Impact of Guidance and Feedback on Password-Creation Behavior. In: ACM Conference on Human Factors in Computing Systems. pp. 2903–2912. CHI '15, ACM, Seoul, Republic of Korea (Apr 2015)

42. Shay, R., Komanduri, S., Durity, A.L., Huh, P.S., Mazurek, M.L., Segreti, S.M., Ur, B., Bauer, L., Christin, N., Cranor, L.F.: Can Long Passwords Be Secure and Usable? In: ACM Conference on Human Factors in Computing Systems. pp. 2927–2936. CHI '14, ACM, Toronto, Ontario, Canada (Apr 2014)
43. Shor, P.W.: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In: IEEE Annual Symposium on Foundations of Computer Science. pp. 124–134. FOCS '94, IEEE, Santa Fe, New Mexico, USA (Nov 1994)
44. Tan, J., Bauer, L., Christin, N., Cranor, L.F.: Definitive Recommendations for Stronger, More Usable Passwords Combining Minimum-Strength, Minimum-Length, and Blacklist Requirements. In: ACM Conference on Computer and Communications Security. pp. 1407–1426. CCS '20, ACM, Virtual Event, USA (Nov 2020)
45. Ur, B., Bees, J., Segreti, S.M., Bauer, L., Christin, N., Cranor, L.F.: Do Users' Perceptions of Password Security Match Reality? In: ACM Conference on Human Factors in Computing Systems. pp. 3748–3760. CHI '16, ACM, Santa Clara, California, USA (May 2016)
46. Ur, B., Noma, F., Bees, J., Segreti, S.M., Shay, R., Bauer, L., Christin, N., Cranor, L.F.: "I Added '!' at the End to Make It Secure": Observing Password Creation in the Lab. In: Symposium on Usable Privacy and Security. pp. 123–140. SOUPS '15, USENIX, Ottawa, Ontario, Canada (Jul 2015)
47. Veras, R., Collins, C., Thorpe, J.: On the Semantic Patterns of Passwords and their Security Impact. In: Symposium on Network and Distributed System Security. NDSS '14, ISOC, San Diego, California, USA (Feb 2014)
48. Wang, D., Cheng, H., Wang, P., Huang, X., Jian, G.: Zipf's Law in Passwords. IEEE Transactions on Information Forensics and Security **12**(11), 2776–2791 (Jun 2017)
49. Wang, D., Wang, P.: On the Implications of Zipf's Law in Passwords. In: European Symposium on Research in Computer Security. pp. 111–131. ESORICS '16, Springer, Heraklion, Greece (Sep 2016)
50. Wang, D., Zhang, Z., Wang, P., Yan, J., Huang, X.: Targeted Online Password Guessing: An Underestimated Threat. In: ACM Conference on Computer and Communications Security. pp. 1242–1254. CCS '16, ACM, Vienna, Austria (Oct 2016)
51. Wash, R., Radar, E., Berman, R., Wellmer, Z.: Understanding Password Choices: How Frequently Entered Passwords are Re-used Across Websites. In: Symposium on Usable Privacy and Security. pp. 175–188. SOUPS '16, USENIX, Denver, Colorado, USA (Jul 2016)
52. Yan, J., Blackwell, A., Anderson, R., Grant, A.: Password Memorability and Security: Empirical Results. IEEE Security & Privacy **2**(5), 25–31 (Oct 2004)
53. Zipf, G.K.: Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology. Addison-Wesley Press, Cambridge, Massachusetts, USA (1949)
54. Zviran, M., Haga, W.J.: A Comparison of Password Techniques for Multilevel Authentication Mechanisms. The Computer Journal **36**(3), 227–237 (Jan 1993)