УДК 004.056.5

METHODOLOGY FOR ESTIMATING PASSWORD STRENGTH AGAINST GUESSING ATTACKS

Zhaparov M. K. PhD, **Akhmet K. M.** MSc student
Suleyman Demirel University

Түйін

Мақалада құпия сөзді шешу (онлайн мен оффлайн құпия сөздерін) шабуылына қарсы қажетті құпия сөздің сенімділігі зерттелген. Сонымен қатар құпия сөздің сенімділігіне негізделмеген оффлайн шабуылға қарсы басқа да қауіпсіз жүйелер қарастырылған. Құпия сөзге шабуылға қарсы қауіпсіздік қауымдастығынан қолданушыларға ұсынылған құпия сөздің саясатының пайдалары зерттелген.

Кілт сөздер: құпия сөз, жобалау шабуыл, онлайн шабуыл, оффлайн шабуыл

Резюме

В статье исследуется требуемая надежность пароля от атаки подборки пароля (онлайн и оффлайн подборки паролей). А также рассматриваются алтернативные системы защиты против оффлайн атак которые не основываются на надежности пароля. Рассматриваются пользы от политики паролей для пользователей которые даны от сообщества безопастности против атак на пароли.

Ключевые слова: пароль, атака подборки, онлайн атака, оффлайн атака

Abstract. *The paper investigates the required password strength from the password guessing attacks (online and offline collection of passwords). And also considered an alternate system of protection against offline attacks are not based on the strength of your password. Consider the use of password policies for users who are from the community against the safety of attacks on passwords.*

Introduction

Passwords exit from the earliest days of computing. Without passwords, the Internet could not grow up to current size. Today non-profit sites such as Wikipedia and Craigslist have tens of millions of users. Social-networking sites such as Facebook grew up to one million users *before taking any funding*.

Passwords showed us that they have incredible persistence. More than ten years ago, Bill Gates declared that "the password is dead". From this time, we were not only able to replace them, but they continue to become the dominant means of Internet authentication, account protection for hundreds of millions of users on some large sites. During this time usability has decreased. The reason is that enormous emphasis is put on coaxing users to choose strong passwords and the rules, policies that govern password choice have grown over time. Yet there is no consensus on what strength various situations demand. This raises the following questions. What strength is required to resist online attacks? When are offline attacks a threat? How much strength is required to protect against offline attacks? Are there ways to protect against offline attacks besides password strength? In this paper, I intend to answer these questions. [1]

Guessing attacks

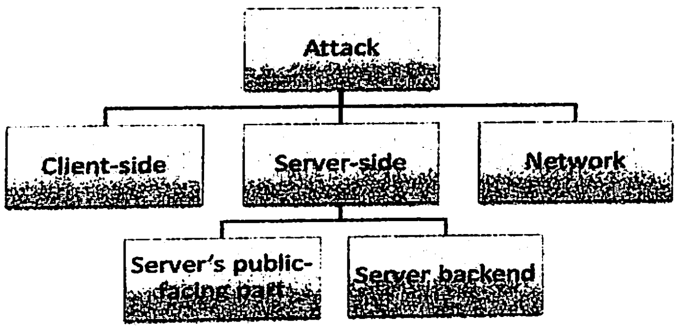

Figure 1. An attack can be addressed at the client side, the server or in the network.

Passwords are mainly attacked on the client, at the server, and in the network as shown in Figure 1. In general, attacks at the client or in the network do not involve guessing. If user has keylogger on his machine, the password is simply stolen. Therefore, guess-resistance is irrelevant. Attacks involving guessing are at the server. These types of attacks are perpetrated at the server's public-face and backend. Attacks on the server's public-face are generally called online attacks. Anyone with a browser can mount basic online guessing attacks on the publicly facing server. An attacker guesses username and password, and lets the servers to do the checking. Usually the process is automated using scripts and guessing dictionaries. Attacks on the backend are harder than the server's public face. It is recommended to avoid storing plaintext passwords at server backends. The server backends should store salted hashes of passwords. [3]

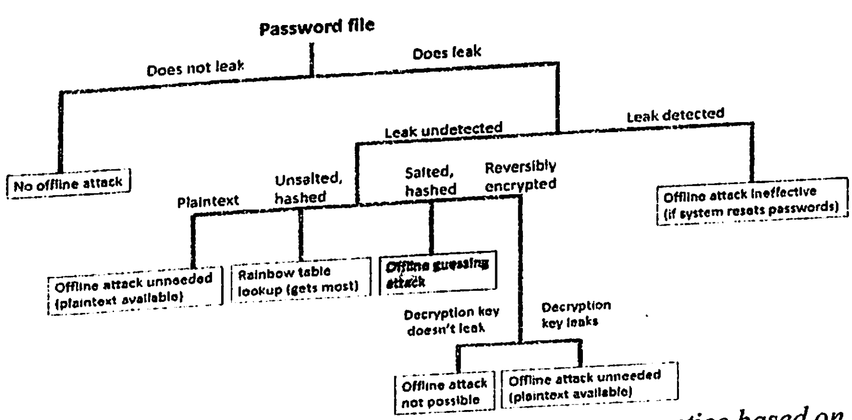

Figure 2. Decision tree for guessing-related threats in common practice based on password file details.

There are three conditions must be met for an offline attack to better than online attack:

1) An attacker must gain access to the server to get to the stored password file. This requires attacker to bypass all backend defences at the server. If he is able to do this, and export the file of salted-hashes, then he can test guesses at the rate his hardware supports.
2) Attacker must go undetected in gaining password file access. If file leak is detected on time, then administrator can reset all passwords in well-designed systems. This will greatly limit attacker's time to guess against the file.
3) The file must be properly both salted and hashed. Otherwise, an offline attack is either not the best attack, or is not possible as we explain next.

If the password file is leaked, and the file leak is undetected, then there are four main cases exist as shown in Figure 2:

1) *the file is plaintext.* In this case, offline attack is clearly unneeded because the attacker reads all passwords directly without guessing.
2) *the file is hashed but unsalted.* In this case, the passwords cannot be simply read as plaintext, but *rainbow tables* allow fast hash reversal for passwords. For example, LinkedIn stored hashed passwords at the server, but they were not salted. In six days, 90% of passwords were guessed.
3) *the file is both salted and hashed.* In this case, an offline attack is needed. The attacker must compute the salted hash for every guess and compare to the stored value. The fate of each password now depends on how many guesses it will withstand.
4) *the file has been reversibly encrypted.* This case, there are two possibilities:
   a. The attacker gets the decryption key
   b. He does not get the decryption key

In Case 4A, offline attack is unnecessary because decryption provides all passwords. In Case 4B, there is no effective offline attack. Even the password is simple as 123456, the attacker cannot verify it without the key from the encrypted file.

In summary, Figure 2 shows that offline guessing is a primary concern only in the narrow circumstance when all of the following apply: a leak occurs, goes undetected, and the passwords are suitably hashed and salted. In all other common cases, offline attack is either impossible (guessing at the public-facing server is better) or unneeded (the attacker gets passwords directly, with no guessing needed). [3]

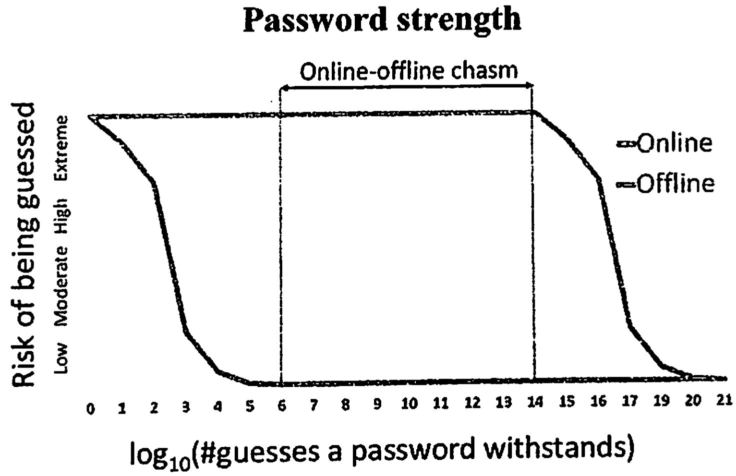

Figure 3. Conceptualized risk from online and offline guessing as a function of the number of guesses a password will withstand over a 4-month campaign. In the region from 10^6 to about 10^{14}, improved guessing-resistance has little effect on outcome (online or offline).

First of all, consider the online attacker. Assume that attacker sends a guess every 1 second for four months at a sustained rate. The attacker is able to send approximately 10^7 guesses in this period of time.

Online guessing (breadth-first). Guessing at this rate against all accounts would overwhelm servers. For example, it is unlikely that Facebook's servers could handle simultaneous authentication requests from all users. In practice, a very small fraction of users authenticate in any 1 second period. Therefore, 10^7 guesses per account seems entirely infeasible, but 10^4 guesses per account is more realistic.

Online guessing (depth-first). If there is no defensive mechanism on server-side, then an attacker can send 10^7 guesses against a single account. This is an extreme case. In a depth-first attack, 10^6 seems a more realistic upper bound on how many online guesses a password must withstand over 4 months. This view is collaborated by study of password policies in 2010. According to that study, 6-digit PIN is allowed for user authentication on Amazon.com, Facebook, and Fidelity Investments and on many other sites. Passwords that will survive 10^6 guesses can be protected from online attacks.

Offline guessing. Now consider the offline attacker. He is able to test guesses using his hardware at a rate far exceeding online attacks. According to a series of papers from CMU, offline attacker can test as far as 10^{14} guesses. We must assume that a lower bound of at least 10^{14} to be protected from offline guessing, and more as hardware and cracking methods improve. This is illustrated in Figure 3.

To summarize Figure 3, there is a large gap between online and offline guessing. *Either* an attacker sends guesses to a server's public-face (online) *or* guesses on his hardware (offline). There are no other possibilities in between. The number of guesses that a password must withstand to expect to survive each attack is extremely different. A threshold of at most 10^6 guesses is sufficient for high probability of surviving against online attacks, whereas at least 10^{14} guesses seems necessary for any confidence against offline attack. These two thresholds for probable safety are different by 8 orders of magnitude. The gap increases if the offline attack brings more distributed machines, and hardware improves, but it is decreased with hash iteration.

Now, consider the benefit in password improvement as a function of the number of guesses it can withstand (10^m). Safety from online guessing can be reached at about m = 6. When m = 6, this effect is gone, further improvement delivers little protection against offline attacks until m = 14. *In gap between online and offline guessing, increasing the number of guesses the password incrementally will survive delivers little or no security benefit.*

As an example, consider two passwords which withstand 10^6 and 10^{12} guesses respectively. Both passwords will survive online guessing, but none of them will survive offline attack. If additional guess-resistance comes at the cost of user effort as in the second case, then the effort appears entirely wasted. In this case, *an effort-conserving approach is to aim to withstand online attacks, but not put in the extra effort to withstand offline attacks.* [3]

Password storage

Salted hashed passwords are far better than most alternatives such as plaintext passwords, hashed-unsalted passwords, or reversibly encrypted passwords and a leaked decryption key. In alternative storage options, advanced offline guessing attacks are either unnecessary (simpler attacks are enough) or impossible. Using site-wide (global) salt protect from generic rainbow tables. Salts per account, iteration counts and storable with hashes, provide additional protection.

Salting also protects from "parallel attacks". For example, if two users have the same password, this will not be apparent and cannot be exploited to simplify attacks.

So storing salted hashes is a preferred way for password storage. If an attacker accesses the password file, and exports it undetected, then he still faces a computationally expensive offline attack as shown in Figure 2. [3]

Alternative system defences against offline attacks

There are alternative approaches to address offline attacks where users do not have to choose better passwords. If the file doesn't leak, or the leak is detected, things are very different as shown in Figure 1. So alternative ways are protecting the password file or allowing leak detection.

Crescenzco et al. found a method to prevent an offline attack. This method works even if an attacker gains unrestricted access to the server backend. It depends on the fact that an offline attacker must guess at a rate far exceeding the normal authentication requests. They introduced a new hashing algorithm that requires randomly indexing into a large collection of random bits (e.g., 1 TByte). Ensuring that the only physical connection to the server with the random bits is matched to the expected rate of authentication requests from the user population guarantees that the information needed to compute the hashes can never be stolen. While the scheme is not standard, it illustrates those ingenious approaches to prevent password file leaks are possible (thereby eliminating the possibility of offline attacks).

Leaked password files can be detected by spiking password files with false passwords called *honeywords*. Honeywords are indistinguishable from actual user passwords. When an offline attacker attempts to authenticate with a "successfully" guessed honeyword, an auxiliary server, called "honeychecker", notifies system administrator to start recovery plans. [3]

Password policies

Passwords and rules are one of the main points of interaction between users and the security community. Users frequently choose poor passwords. This caused sites to set policies that force minimum strength rules. Strength rules generally the constrain passwords with respect to:

1. Length
2. Composition (e.g. lowercase and uppercase, digits, special characters)
3. Dictionary membership (in any language).

Moreover, there are many rules once the password has been chosen by a user. Commonly users have been advised to:

4. Do not write their passwords down
5. Do not share their passwords with anyone
6. Change their passwords frequently
7. Don't re-use their passwords across sites.

Rules 1-2 are merely the most common policies usually given to users. Many sites require passwords to be at least 8 characters long, and contain at least three of four character types (lowercase and uppercase, digits, special characters). For example, PayPal requires a new password of length at least 8, with both upper and lower case letters, with at least one special character, and it is not a word that can be found in the dictionary. This metric based on entropy very poorly models guessing-resistance of password. Users respond to composition policies with minimally compliant choices such as Sn00py2 and Pa$$w0rd. For example, "Sn00py2" is stronger cryptographically than "749278". However, it might be much easier to guess "Sn00py2" for someone who knows the user. Passwords scoring better by the metric are not guaranteed to fare better under guessing attacks. A study by Weir et al. concludes that "the entropy value doesn't tell the defender any useful information about how secure their password creation policy is." [2][3][4]

Rule 6 was apparently designed to limit the time an attacker had to guess a password in the case of offline attacks. A study by Zhang et al. found that new passwords very closely related to old ones. They were allowed to submit guesses for new passwords with knowledge of old passwords. As a result, they could quickly guess 17% of passwords under 5 guesses in online attack, and 41% of passwords under 3 seconds in offline attack. Thus, new passwords are highly predictable from old ones. Rule 7 may be more important. [3][5]

Rule 4 is questioned by many security experts. It's obvious that writing passwords in plain text is dangerous. Even keeping written passwords in a safe place such as a wallet, only increases the risk.

Finally, none of Rules 1-5 help at all against phishing and keylogging. This advice is moot and the user effort is wasted against these threats. [2]

Conclusion

In this section, we are going to summarize some of the key findings of the paper. Password guessing attacks are either online or offline. The guessing-resistance differs enormously from each other. Withstanding 10^6 guesses is probably sufficient against online attacks, while withstanding 10^{14} or more guesses may be needed against offline attacks. There is a large gap between two thresholds. There is small security benefit in exceeding the online threshold while failing to reach the offline one. Passwords that fail to completely cross this gap waste user effort.

Strict password composition policies are unnecessary burden for users when sites store passwords in plaintext or reversibly encrypted. If the password file has not been properly salted and hashed, then user effort to withstand beyond 10^6 guesses is better spent elsewhere.

In further study, I intend to research about phishing. I am going to write about user education to avoid phishing and protection from it.

References

1. C. Herley and P. van Oorschot. A research agenda acknowledging the persistence of passwords. IEEE Security & Privacy, 2012.
2. C. Herley. So Long, And No Thanks for the Externalities: Rational Rejection of Security Advice by Users. Proc. NSPW, 2009.
3. D. Florencio, C. Herley, and P. van Oorschot. An Administrator's Guide to Internet Password Research. Proc. USENIX LISA, 2014
4. D. Florencio, C. Herley, and B. Coskun. Do Strong Web Passwords Accomplish Anything? Proc. Usenix Hot Topics in Security, 2007.
5. Y. Zhang, F. Monrose, and M. K. Reiter. The security of modern password expiration: An algorithmic framework and empirical analysis. Proc. ACM CCS, 2010.