CYBERHOUND

PASSWORD SECURITY
A RESEARCH PAPER

WHAT MAKES A SECURE PASSWORD

A deep dive into what makes a secure password and why.

Christopher Gersch
Dr James Birkett PhD.

INTRODUCTION

Computer passwords have been ubiquitous since the earliest time-sharing systems of the 1960s. Often maligned, passwords are almost considered a necessary evil. The struggle to think of a new password whenever the dreaded "Your password must be changed" notification appears is enough to send people into a stunned stupor—staring at the monitor thinking of ways to slightly alter the existing password such that it fits the seemingly arbitrary password strength requirements.

Passwords serve a necessary purpose of course. The inconvenience of having to remember multiple passwords far outweighs the cost of our data getting compromised, our money siphoned from our bank accounts, and nefarious entities reading our private conversations.

The problem with passwords is that passwords that are easy for humans to remember are typically also easy for a machine to crack, and the most widespread methods of making passwords stronger, such as requiring the inclusion of numbers or punctuation characters, or a mix of upper and lower case letters, typically make passwords much harder for users to remember.

THE BASICS

Before delving into what is wrong with current passwords, and how to make yours more secure, it's necessary to cover off the basics. These will, (or should be), plainly obvious, and pretty ho-hum, but more often than not, it's the basics that can untie the whole knot in dramatic ways.

- Don't give out your password. This means more than telling your partner your laptop password. It also means not emailing your password, (or ask someone to email theirs), nor put it on a post-it-note to hang off your computer monitor. Your passwords should be personal. If you have to give it out, change it to something else beforehand, (that's not being used elsewhere). If possible, set a timeframe before changing the password back.
- Don't save your password anywhere in cleartext. This means in a notebook, Excel spreadsheet, or a text file. It also means not adding the password as a description to an Lightweight Directory Access Protocol account.
- Be aware when logging onto computers that you do not control. Avoid logging onto services such as Gmail, or Facebook. If you have to, open the browser in private, or incognito mode. This will not protect you from keyloggers that may have been installed onto the device, but it will help clean things up properly after logging out of the web site, (please remember to logout!), and quit the browser.
- If you join an open Wi-Fi network, it's very strongly recommended that you protect yourself via a VPN service (such examples being Express-VPN, X-VPN, or PIA). Some of these services offer free and annual subscriptions. They all provide a secure open-Wi-Fi connection, otherwise it is trivial for someone to literally view the packets in the air, and see any usernames and passwords sent in cleartext.

HASHING

Passwords should never be stored in clear text. If a website or a service can retrieve your old and forgotten password, then that should be very concerning. At this point, if you can think of any such websites, go now - delete your account, or at the very least, change your password to one you don't use anywhere else.

For sites that do not store your password in clear text, they will have hashed your password. People sometimes confuse hashing with encryption. The main difference is that when you encrypt a message it should be possible for a recipient who knows the right private key to decrypt the message; in other words, encryption is a two-way process. In contrast, hashing is a one-way process to turn your password into a seemingly random string of data. The key term here is "one-way".

Even if you have the hash of a password, there is no secret key that will enable you to quickly determine what the original password was. This works well for passwords because the login system on a website or computer system does not actually need to decrypt the hashed password; instead it hashes the password provided by the user and compares the resulting hash to the one stored in its records. If the user entered the wrong password, the hashes will be different, so the system can refuse access without ever needing to know the correct password in clear text.

The sporting giant Under Armour announced in April 2018 that the user database for their MyFitnessPal App, containing usernames, email addresses, and hashed passwords had been compromised in March 2018. Contrasting this hack with the infamous LinkedIn.com hack in 2012 demonstrates the difference between good and bad password security. The MyFitnessPal app used two security defences that will make recovering the passwords much more difficult.

Another Day, Another Hack: 117 Million LinkedIn Emails And Passwords

Four years later, the 2012 LinkedIn breach just got way worse.

The first is the choice of hashing algorithm. There are a number of algorithms used to hash passwords. Many systems use general-purpose hashing algorithms such as MD5, SHA-1, or SHA-3, but MD5 and SHA1 are considered extremely weak. Even the more modern SHA-3 is not ideal for password hashing, so it is considered best-practice to use an algorithm designed specifically for hashing passwords, such as bcrypt or Argon2. This is because general-purpose hashing algorithms are designed to be fast, but for passwords it's actually better if computing the hash is much slower (on the order of milliseconds rather than microseconds). Users will not notice a few extra milliseconds delay when logging in, but the delay becomes significant for hackers trying millions of possible passwords in a brute-force cracking attempt.

The second and perhaps more important security defence used by the MyFitnessPal app (but not by LinkedIn) is the concept of password salts.

When the user sets their password, the system generates a random value known as the salt, which should be different for each user on the system.

The salt is combined with the user's password before hashing and the system stores both the salt (in clear text) and hash value in its records. When the user logs in a second time, the system looks up the salt for that user and combines it with the password the user has entered, computes the hash of this combination and checks it matches the hash on record.

Using password salts does not make it any more difficult to crack the password for a single user by brute force, but without password salts, it is possible to compute a lookup table for hashes of common passwords in advance. When password hashes are leaked, if a hash value appears in the lookup table, the cracker can recover the corresponding password immediately from the table. In practice, hackers use a more sophisticated version of this principle known as rainbow tables. With password salts, the hacker would need a different table for each possible value of the salt. Since these salts should be different for each user, it makes the pre-computed tables worthless as they would need a different lookup-table for each user. As such, it would be faster to simply brute-force each user's password separately.

In the case of the 2012, and 2016 LinkedIn hacks, a vast majority of the passwords (>90%) were cracked by security professionals within a week. This hurts the Internet community on a number of fronts, however the main reason why this is such a critical issue is not immediately obvious to the layperson. Security professionals and hackers alike now can take the millions of passwords that were cracked and then used to make dictionary attacks a lot more sophisticated; the sample size of known passwords for everyday people had just significantly grown almost overnight.

> *...recent studies showing that 90% of all passwords are vulnerable to attack in seconds.*
>
> **http://www.computerworld.com, 1 May 2017**

PASSWORD ENTROPY

It should be quite obvious that the more random and complex your password is the more difficult it is for it to be cracked. This concept is known as password entropy and is measured in bits. Password entropy is a guide on how much effort is required to crack a password via brute-force. A password with 20 bits of entropy will be as strong as a string of 20 bits chosen randomly (via a coin toss for example). In other words, a password with 20 bits of entropy will require 2^{20} (1,048,576) attempts to exhaust all possibilities during a brute-force attack. This means adding one bit of entropy to a password doubles the number of guesses required, which makes an attacker's task twice as difficult.

20 bits of entropy sounds like a lot, but a specialised system using multiple GPUs which are optimised for parallel processing could compute around a million bcrypt hashes per second, Entropy values of around 45 bits should be the absolute minimum requirement for passwords used to protect sensitive data, such as your email, online banking or social networking sites, but more is better. This will give you months (instead of minutes, or hours), to change your password in the case of a compromise.

In its simplest form, if every character of a password was chosen entirely at random, the entropy of the password would be:

$$Entropy = log2(S^L)$$

Where:

S is the size of the symbol pool, and

L is the password length

Consider the common password Passw0rd! It has a combination of upper and lowercase, special characters, and numbers, so we could analyse this in a naive way by assuming that each character was chosen at random from the 94 printable characters found on a standard US keyboard (a-z, A-Z, 0-9 and 32 other symbols). With a length of nine characters, the entropy of this password would be:

$$log2(94^9) = 58\ bits$$

But each character of this password was not chosen at random, so this formula should not be used. The 2012 LinkedIn attack allows us to find a better estimate of the entropy in this password, since it was among the most common passwords used. We know that 18,208 of the 117 million compromised accounts used this password, so a better estimate of the entropy is:

$$log2(117000000 \div 18208) = 13\ bits$$

Even if LinkedIn had used a good password hashing algorithm and password salts, a dictionary-based attack, which will look at common words and simple substitutions (5 for s, @ for a, etc), could crack this password in a matter of milliseconds. So how can we do better?

People are better at remembering words and phrases than random strings of characters. If the system you're using allows long passwords, one possibility is to simply use a password consisting of multiple words, often called a passphrase. Consider the passphrase ***Nelson buys ugly fruit.*** Studies have found that the entropy in English text varies from around six bits per word when using only simple sentences up to around nine or ten bits per word if unusual words are included, but an average of around eight bits per word (excluding common words like "the", "and" and "of") is a reasonable rough estimate. This phrase has four words, so the entropy of this passphrase is probably somewhere around 30 bits. This is a lot better than Passw0rd!, but in the event of a compromise similar to the MyFitnessPal event, it could still be guessed by password cracking software in a matter of hours on fast hardware.

If we want more security than this, using longer phrases is an option, but there's a caveat: it may be tempting to use song lyrics, quotes from movies or the bible as these are easier to remember, but using sources such as these drastically reduces the amount of entropy in the passphrase because hackers can build dictionaries of song lyrics, movie quotes or bible verses into their password cracking algorithms. When choosing a good passphrase it is best to avoid using phrases that can be found in published works (including this one: don't use "Nelson buys ugly fruit"!)

If we want even stronger passwords, using words selected entirely at random, rather than coherent English phrases is an option. There are computer programs that will do this for you, but some may use poorly designed random number generators that produce easily guessable results, while others are malicious and could report the passwords back to their designer. The truly paranoid may wish to consider entirely offline systems for generating random passwords, such as the diceware wordlist, which is designed to be used with good old-fashioned six-sided dice.

The following table gives an indication of the type of passwords that are used, their corresponding entropy and an estimate on how long a modern, multi-GPU system may take to crack the password.

"...When most credentials-based attacks no longer bother with brute-force methods, relying on password complexity doesn't really help."

https://www.infoworld.com, 5 May 2017

TYPE OF PASSWORD	EXAMPLE	ENTROPY (APPROX)	TIME TO CRACK
Common English word with simple symbol substitutions	Duckl1ng	< 15 bits	Seconds
Song lyric	Friday night and the lights are low	< 20 bits	Minutes
Short English phrase	Neslon buys ugly fruit	~ 30 bits	Hours
Three words chosen using diceware	fogy lead devil	39 bits	Months
Long English phrase	Hotel where we stayed in Barcelona was noisy	~ 50 bits	Years
Four words chosen using diceware	avow sequin drama iffy	52 bits	Years
Six words chosen using diceware	prove allen gown sense observe mustang	77 bits	Probably not in our lifetime

RE-USE?

Purists would suggest to never re-use passwords, but most people have dozens of online accounts so it is not practical for most people to remember that many strong passwords or passphrases. One strategy for coping is to consider how severe the consequences would be if the password were compromised. For most people, banking sites or sites that store credit card details (such as online shopping) are highly sensitive for obvious reasons; you don't want your money stolen. It is important to use a strong password that is different for each of these sites. Almost all banks will offer some form of two factor authentication, such as security tokens - use it! Deleting stored credit card information from online shopping sites and entering the credit card number each time when making a purchase can help limit the consequences of accounts being hacked.

Email accounts often store a lot of sensitive information, and can also be used to reset the passwords for other services, so these must be treated as sensitive. Social media sites and messaging services may also contain personal information. For other sites, such as forums or online gaming services which don't contain sensitive information, users may consider the difficulty of remembering unique passwords for each site not worth the benefit. If you do choose to reuse a password, make it a good one; aim for at least 45 bits of entropy.

The problem with reusing passwords is you are relying on the other end to do the right thing by using strong password hashing algorithms and salts, not storing the password in clear text, patching servers, employing robust security principles, and so on. We can trust some site more than others. However, even widely trusted sites like LinkedIn and Yahoo can be, and indeed have been, susceptible to data breaches.

We tend to use the same email address to register accounts with (Dropbox, Adobe, etc). Along with this, we also tend to use the same password. Sites like https://haveibeenpwned.com/ search most data breach dumps to see if your email address has been exposed. Whilst some sites do the right thing by using a strong hashing algorithm such a bcrypt with a salt, others still do, or have employed Secure Hash Algorithm 1 (SHA1) with no salts. If your email address has been exposed, and unless your password has a very high entropy, it's safe to assume your password has been cracked.

It takes one service—one weak link in the chain—to bring everything crashing down. Even using long passphrases with high entropy values cannot guarantee the password will not be exposed in a future crack. Therefore, limiting your exposure by using individual passwords per website/service is the only way to guarantee your online, (and offline) safety. The problem with this is how can one remember the potentially dozens of different passwords for each of the accounts we all have?

> *..the more often you ask someone to change their password, The weaker the passwords they typically choose.*
>
> **Prof Alan Woodward** - University of Surrey, 9 August 2017

PASSWORD MANAGERS

Latest figures put the average number of online accounts a person has at between 10–25, and trying to remember over two dozen complex, and/or long passwords with high entropy values is most likely asking too much.

Password managers are software that are designed to store your individual passwords encrypted with one master passphrase as the key. They have the ability to generate very complex passwords, and safely store them on your device, locally, or in the cloud. This enables a user to easily create high-entropy, individual passwords, strongly immune to brute-force and dictionary attacks, while the user only has to remember the one master password for the password manager. Password managers typically have extensions for integrating with the popular web browsers, and may be available on mobile devices such as phones or tablets as well.

Like any software, password managers may have bugs or security vulnerabilities, so is worth reading reviews and search to find out about their past history of security vulnerabilities and how they handled them. Another issue with password managers is that if the master password is forgotten, it's next to impossible to retrieve it, meaning all the individual passwords contained within the manager are also lost. However, memory can be a funny thing – recall can happen at any time, and muscle memory of typing the password out on the keyboard should also not be underestimated.

CONCLUSION

Common password criteria employed by most organisations and websites, typically comprises of 6-8 characters, mix of upper and lowercase with numbers, and can arguably be considered woefully lacking.

The natural instinct to add a "1" or "!" at the end of a password or employing "l33tspeak" in the attempt to make it more complex will not protect it against modern password cracking techniques and should be avoided at all costs. If your passwords looks like P@ssw0rd1 or Adm1n!str@t0r, go change them straight away.

Such password criteria ultimately do nothing but annoy users and instill bad password practices. Passphrases—passwords that contain a series of uncommon, apparently random words—are typically much easier to remember for the user, and are inherently more robust due to high password entropy values.

The need to use separate passwords for each site that holds your sensitive data is still paramount. This protects the user in case the website is compromised, or employs poor password security, such as storing passwords in clear text, using weak hashing algorithms, or not using salts.

If the site offers two factor authentication, use it. This extra step makes targeting your accounts significantly more difficult than another account that doesn't employ two factor authentication. In other words, it makes you a hard target, and should give you time to reset your password.

Investigate password managers and use them to create or store your pass-phrases. Whilst they can be considered a potential single point of failure, the advantages of having a secure, easy to use system to manage all your passwords should not be overlooked.