How do you make a strong passphrase?

Tom Eccles

October 15, 2015

1 Introduction

I am writing this in response to repeated claims by some large news organisations that a good way of having a strong passphrase is to use as many different character types as possible (the example they gave was to use "pa$$word"[1] instead of "password"). This in it's self I agree with, however they did not even mention the importance of passphrase length and in some disastrous cases impose limits upon the length. In this document I hope to provide a simple mathematical argument for the importance of passphrase length.

2 Threat Model

I will be discussing the case where an attacker is attempting to gain access to an account by trying every possible password until one works (a brute force attack). I am also assuming that the attacker is not targeting the user directly because in this case I believe that the majority of users have no chance at all due to the effectiveness of social engineering attacks such as spear phishing.

3 The Mathematics

3.1 Assumptions and Probability

I am going to assume that the probability of an attacker guessing one character in a passphrase is independent of them guessing another character. This may not be always true because the attacker may have some intelligent software which (if the passphrase is composed of words) may be able to guess the remaining characters of a word after it already has some of

[1] It is worth noting that using "leet speak" such as replacing "s" with "$" or replacing "l" with "1" or "a" with "@" provides less security than one might expect because many modern brute forcing tools such as John The Ripper (http://www.openwall.com/john/) can automatically try "leet speak" variations to a given dictionary.

them (like the child's game 'Hangman'). This assumption has been made to make the mathematics solvable with my current knowledge. I think that this assumption is reasonable because we are discussing an attacker who is trying every possible passphrase.

As anyone who has studied probability knows, the probability of two independent events both occurring equals the product of the probabilities of the two events: $P(A \cap B) = P(A)P(B)$. And such follows for more independent events: they are all multiplied together.

Therefore, if an attacker tries every password with a length of n and a character space of s, the number of passwords which the attacker has tried is s^n.

However, in our threat model we specify that the attacker tries every password and so the attacker must try all of the lengths up to n first. Therefore the number of attempts made by the attacker is $s + s^2 + s^3 + ... + s^n$. This is similar to the geometric[2] series $1 + s + s^2 + ... + s^n$, which (assuming that $s \neq 1$) has a sum equal to

$$\frac{s^n - 1}{s - 1}$$

This is geometric series is 1 too high for us because a geometric series starts at $s^0 = 1$ and so the sum of all the attempts made by the attacker is

$$\frac{s^n - 1}{s - 1} - 1$$

I will refer to this as the complexity of the passphrase, c. This inherits the same constraint that $s \neq 1$. This does not matter because the character space will always be greater than 1.

3.2 Rates of Change

In this section I will try to demonstrate algebraically that the increase in passphrase complexity due to length is more rapid than the increase due to the character space.

The rate of change of complexity with respect to the length of the passphrase while keeping the character space constant (the partial differential of complexity with respect to length) is

$$\frac{\partial c}{\partial n} = \frac{s^n \ln(s)}{s - 1}$$

The rate of change of complexity with respect to the character space (the partial differential of complexity with respect to character space) is

$$\frac{\partial c}{\partial s} = \frac{(n-1)s^n - ns^{n-1} + 1}{(s-1)^2}$$

[2]see https://en.wikipedia.org/wiki/Geometric_series

To make these two easier to compare I will give them both the same denominator:

$$\frac{\partial c}{\partial n} = \frac{s^n \ln(s)}{s-1} \cdot \frac{s-1}{s-1} = \frac{s^{n+1}\ln(s) - s^n \ln(s)}{(s-1)^2}$$

One can see that the numerator of $\partial c/\partial n$ is of the order s^{n+1}. This is greater than the order of the rate of change with respect to character space size which is of order s^n. Therefore the increase of complexity with respect to the number of characters in the passphrase is faster than the increase with respect to the character space.

4 Conclusion

In conclusion I have shown that it is important to have both a long passphrase and several types of characters. Although it is worth noting that just by having lower, upper case and space characters, you already have a character space of 49. So my advice would be to concentrate on increasing the length of your passphrases. A good way of doing this is to use several words which do not make grammatical sense together. Passphrases based on words are also more memorable, this makes users less likely to write their passwords on post-it notes next to their computers[3]. For a good explanation see https://xkcd.com/936/.

5 Dictionary Attacks On The XKCD scheme

A different possible attack is a "dictionary attack". An attacker who wanted to specifically target the scheme recommended in the xkcd article could brute force combinations of words based on a dictionary of words. As long as the words do not make grammatical sense together, this could be treated with the same mathematics as the previous case except s will now represent the space of different words tested and n will be the number of words in the passphrase. This is likely to be very large (at least 1000) and so it is equivalent to probably a short passphrase with an impossibly large character space. With a character space that is so large I do not think this is a viable attack.

[3]Although my advice is not to try to memorise individual passwords. Instead use a passphrase manager such as keypass http://keepass.info/ to store passphrases. This allows every password to be unique and very long. Unique passphrases are important because it limits the scope of compromise when one passphrase becomes known (for example due to a service you use becoming compromised). By not requiring passphrases to be at all memorable they also do not need to be constructed of words. This means that even a more intelligent attacker who uses a dictionary attack will not be at any advantage over the weak attack we have considered.

6 Future Work

As noted, within words the probabilities of letters are not independent and so a discussion of an attacker who takes advantage of this would require more complex calculations.

7 Appendices

7.1 "Passphrase?"

I have chosen to use the word "passphrase" over the word "password" because it encourages longer login identifiers. There is no reason to use a single word and this should be discouraged as it is easier to brute force.

7.2 Distribution and Licensing