Project TDDD17

Niklas Granberg (nikgr371) och Anna Pestrea (annpe689)

April 2019

1 Introduction

Creating passwords today can be quite the hassle. There are many ways to create your passwords to make sure they are secure. These days many products and services containing your personal and private information are protected by a password, making it a very important subject. But not all these services and products actually keep their users passwords secure. Many big companies have been found to store passwords in plain text without any sufficient security measures in place. This lack of security leads to the passwords being acquired by an intruder. If the passwords are stored in a unsecure way, then they can easily be recovered by the intruder. In worst case the intruder might also share the passwords on the internet, opening up for multiple attacks directed towards the affected users.

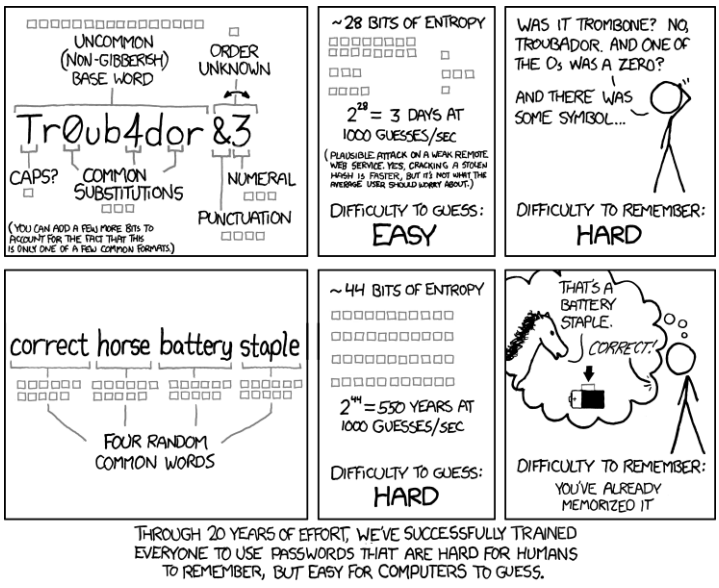

Figure 1: An example of how to design a password

A countermeasure for this is to follow the recommendations set by companies and experts when choosing an password. But how well are these followed and what kind of passwords do people use? Research show that users generally use very simple passwords that are easy to remember, even though there are new breaches happening every single year. We want to understand this subject better and therefore we state the following research questions:

- Which passwords that are most common and how secure are these passwords?
- What is the entropy (security measured in bits) of the common passwords?

2 Theory

2.1 Most common passwords

Different groups present the most common passwords each year. One of the more common is SplashDatas' list. For this report the 2018 top 25 list[1] will be used for analysis and discussion. The list is presented in Figure 2.

1. 123456	8. sunshine	15. abc123	22. aa123456
2. password	9. qwerty	16. football	
3. 123456789	10. iloveyou	17. 123123	23. donald
4. 12345678	11. princess	18. monkey	
5. 12345	12. admin	19. 654321	24. password1
6. 111111	13. welcome	20. !@#$%^&*	
7. 1234567	14. 666666	21. charlie	25. qwerty123

Figure 2: SplashDatas' 2018 list of top 25 most common passwords

2.2 Entropy

Entropy is a measurement in bits of how hard it is to correctly guess a password. How hard a password is to guess starts with two characteristics: How long the password is and how many possible characters that can be used for the password. For passwords of length L and a character set of N characters, there are L^N possible combinations of passwords. This assumes that the probability of each character is uniform, meaning it is truly random. To then calculate the entropy, the natural logarithm is used since it is measured in bits. The entropy can therefore be calculated with the following formula:

[1] http://time.com/5478693/worst-passwords-2018/

$$E = log_2(N^L) = /\text{according to logarithmic laws}/ = L * log_2(N)$$

2.3 Standard deviation

In order to get more interesting results the standard deviation was also calculated from the average value and the entropy's. The standard deviation was calculated with the following formula, where x is the current entropy, m the middle value, n the number of entropy's in our file and the σ is the standard deviation.

$$\sigma = \sqrt{\frac{\sum(x-m)^2}{n}}$$

2.4 Required key lengths

There are several groups that present what they have calculated as the minimum amount of required bits in a key for it to be properly secure. One such group is ECRYPT-CSA. Their recommendations for a symmetric key in 2018 was 80 bit for legacy systems, 128 bits for near future protection and 256 for long term [4]. This applies to hardware generated, truly random keys where the probabilities of each character is uniform. Passwords are generally not up to the same standards but for the entropy in this paper we will assume that it is.

3 Method

3.1 Downloading common passwords

The company known as RockYou Inc. had 2009 a security breach which allowed hackers to get their hands on over 32 million users password [1]. The leaked passwords was then distributed over the internet and can today be downloaded. In 2012 Linkedin was breached and around 60 million passwords were leaked. These passwords were hashed but not salted, meaning they are still relatively easy to find cleartext for. Hashes.org has been working on reversing these hashes and as of 7/3-19, 97.99 % of the passwords have been reversed. This data will also be used. Since not all of the passwords have been reversed, this dataset is not as reliable as the RockYou set, but is still useful.

To make sure that the files are not unsafe, filled with viruses for example, they will first be downloaded onto a virtual machine installed via Virtualbox[2]. This is only to make sure that anything bad cannot happen to the testing machine and will not affect the tests in any way. The software program MalwareBytes[3] will then be run on the virtual machine to scan the downloaded files.

Websites have different requirements and restrictions on passwords. Therefore the requirements have to be gathered to calculate entropy. The only demand

[2]https://www.virtualbox.org/
[3]https://www.malwarebytes.com/

that Rockyou made of the users passwords was that it was going to be at least 5 characters long. No other requirements existed. Since the datasets are old, the website waybackmachine[4] was used for LinkedIn. The company had (at the year 2012) the requirement that the password must have a minimum length of 6 characters or more.

3.2 Ethical considerations

Since this data set is actual data from companies and the data might have been retrieved in an unethical manner, some ethical regulations are to be applied. The data that will be presented in this paper will not have any kind of hints that would allow a reader to identify a user from this data set. In other words, only the passwords themselves will be shown.

3.3 Analysis

A python program (PasswordAnalyzer) is developed to analyze the different data sets. This program is developed by the authors. A another program (Counter) was also developed in order to compute the minimum, average and maximum entropy of the two data sets, since it would be ineffective to go through the data sets by hand.

To calculate the entropy of a password, the amount of possible characters needs to be known. RockYou did not appear to have had any special limitations on what characters were allowed in passwords, since many UTF-8 characters were present in the data, such as Ñ or ¿. We assume that the possible amount of characters to use for the entropy is the max of UTF-8, which is 1,114,111 [3]. But in the dataset only 206 characters appeared. Therefore both the UTF-8 max value will be used to calculate entropies and the 206 character count will be done too. The 206 character count not really the entropy though, since the possible available characters is the total UTF-8 count, 206 characters is way too low. But in a practical sense, many of the UTF-8 characters will never be used by the regular person. It is therefore assumed that there is not a uniform probability over the RockYou dataset entropy. Then two entropies are calculated where in one the probabilities are uniform and another where the probabilites are skewed towards the 206 characters found in the dataset.

LinkedIn had the normal 62 characters (a-z,A-Z,0-9) and also the following 32 special characters: $\$[]\_@+-.*!=/\%\#?\&,;)(\text{"}\hat{}<:'>|\backslash`\{\}$. These are all the special ASCII characters. Adding these characters to the total count adds up to 94 possible characters.

When calculating the standard deviation some code was added to the PasswordAnalyzer program.

[4] https://archive.org/web/

4 Result

This section will present our results. The python program PasswordAnalyzer is available at the github account Marsvinguy[5].

Minimum	Average	Standard deviation	Maximum
7.16	69.22	20.83	1711.21

Table 1: Table presenting the minimum, average and maximum entropies for the LinkedIn data set.

UTF-8	Minimum	Average	Standard deviation	Maximum
Not included	7.67	74.77	22.32	2198.37
Included	20.09	195.20	58.26	5745.01

Table 2: Table presenting the minimum, average and maximum entropy for the RockYou data set. The first row does not include the UTF-8 set in the calculation while the second row does

5 Discussion and conclusion

Looking at the data displayed in tables 1 and 2 it shows quite a difference in number. The minimum values for the tables are relatively close to each other at 7.16 for LinkedIn and 7.67 for RockYou. This is even though RockYou has a lot more possible characters than LinkedIn. This is most probably because of RockYou requiring only 5 character passwords whereas LinkedIn requires more characters.

The average values do show that LinkedIn set has almost the same average entropy as RockYou. This is probably because of LinkedIn passwords being longer than RockYou passwords. Since RockYou has many more possible characters the LinkedIn passwords need to be longer to be able to match the RockYou values. This is supported by the formula for calculating entropy. Since the character amount is factored in using the natural logarithm, it grows much slower than the linearly increasing length of the passwords. So without actually measuring the average length of the passwords we can see that LinkedIn passwords are on average longer.

However when assuming a uniform distribution over the full set of UTF-8 character when calculating the entropy of the RockYou data set, the results becomes very different. Since the entropy is larger the passwords are harder to guess. These results are very logical since more characters leads to many more possible password combinations. It was however very interesting to see that the difference in entropy (when assuming a uniform distribution over the full set of

[5]https://github.com/Marsvinguy/PasswordAnalyzer

UTF-8 or only over the 206 character set) was around 2.7 for both the minimum (2.8), average (2.6) and maximum (2.6) value.

From the standard deviation we get interesting results. It is logical that the deviation is higher for the RockYou data set, since the requirements for it was quite low and that gives more room for different passwords. This should mean that we have a bigger standard deviation since the passwords vary more. While the deviation did not change so much between the files, it changed a lot when adding the UTF-8 set. This is because of that we only add a factor that will affect all of our results.

But are these passwords secure? Looking at the average and standard deviation, most passwords are in the 50-90 bits of entropy range. According to ECRYPT-CSA, 80 bits is required in legacy systems for a symmetric key to be secure. This means many passwords fall below this threshold. Another problem is that these measures are for computer generated, truly random keys. Passwords are not up to the same standards since they are made by humans, as seen in section 2.1. There are patterns in how humans create passwords which means that the probabilities of the character set is not uniform. This means the entropy calculated in this paper is higher than the the actual entropy. So for the top range of these passwords, they can be seen as secure. But most of the passwords fall below what would be considered secure.

References

[1] Jack Schofield, "32.6m passwords may have been compromised in RockYou hack", The Guardian, `https://www.theguardian.com/technology/blog/2009/dec/15/rockyou-hacked-passwords`, December 2009.

[2] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[3] F. Yergeau, "UTF-8, a transformation format of ISO 10646" RFC 3629, RFC Editor, 2003, `https://tools.ietf.org/html/rfc3629` retrieved 17/04-19

[4] N. Smart et al. "Algorithms, Key Size and Protocols Report (2018)" ECRYPT-CSA. March 2018. Accessed May 2019 at http://www.ecrypt.eu.org/csa/documents/D5.4-FinalAlgKeySizeProt.pdf