UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Janne Uutela

UNDERSTANDING COMMON PASSWORD DESIGN: A STUDY TOWARDS BUILDING A PENETRATION TESTING TOOL

Bachelor’s Thesis
Degree Programme in Computer Science and Engineering
October 2023

Uutela J. (2023) Understanding Common Password Design: A Study Towards Building a Penetration Testing Tool. University of Oulu, Degree Programme in Computer Science and Engineering, 37 p.

ABSTRACT

Almost everything that is meant to be kept private is currently being protected by passwords. While systems and devices can be designed with robust security measures, the efficacy of such systems can be compromised if the end-user chooses a weak password, especially one easily found in common wordlists. Given the prevailing security dynamics, especially with the ongoing Ukraine war and Finland's NATO membership considerations, the inadequate protection of WiFi devices may transcend individual privacy concerns. Supo, the Finnish Security and Intelligence Service, posits that routers with subpar security could pose considerable national security risks.

This thesis aims to investigate the strategies people use when creating new passwords. This is done by using prior knowledge about password creation habits and by conducting an analysis of leaked passwords. The study also examines existing tools for password list generation for penetration testing to see what the strengths and weaknesses of those tools are. This will be the groundwork for creating a lightweight tool for password list generation that can be used to do penetration testing with dictionary attacks and possibly detect if weak passwords are being used. The problem with the current tools is that they either create a very large wordlist or are too small to be practical. They also seem to lack the mangling capabilities of the wordlists.

The proposed solution is evaluated using the wardriving method, accompanied by the acquisition of pmkid hashes from WiFi access points. Subsequently, these hashes are matched against passwords generated by the designated tool, leveraging Hashcat to ascertain their decryptability. Through this process, the study also provides a snapshot of WiFi password robustness within the City of Oulu.

The findings revealed that approximately 6% of WiFi access points employed passwords deemed too weak. This discovery aligns with earlier research conducted in the city of Oulu, where a related investigation highlighted that nearly 14.78% of devices lack password protection, effectively operating as open access points [1].

Keywords: Cyber security, Linux CLI, penetration testing, wlan security

Uutela J. (2023) Yleisten suunnittelumenetelmien ymmärtäminen salasanojen luomiseen: Tutkimus penetraatiotestaustyökalun rakentamiseen. Oulun yliopisto, Tietotekniikan tutkinto-ohjelma, 37 s.

TIIVISTELMÄ

Lähes kaikki yksityisenä pidettävät asiat ovat tällähetkellä salasanojen suojaamia. Laitteet ja järjestelmät voidaan suunnitella tietoturvaominaisuuksiltaan kattavaksi, mutta näiden laitteiden ja järjestelmien turvallisuus voi vaarantua, jos loppukäyttäjä valitsee laitteen salasanaksi heikon salasanan. Etenkin jos valittu salasana sattuu vielä löytymään yleisistä salasanalistoista. Wifi laitteiden riittämätön suojaaminen voi aiheuttaa turvallisuusongelmia, kun tarkastellaan vallitsevaa turvallisuusdynamiikkaa, Ukrainan sotaan ja Suomen Nato jäsenyyteen liittyen. Suojelupoliisi arvioi että heikosti suojatut reitittimet voivat aiheuttaa merkittäviä kansallisia turvallisuusriskejä.
Tämän opinnäytetyön tavoitteena on tutkia ihmisten käyttämiä strategioita salasanojen luomiseen. Tämä tehdään käyttämällä aiempaa tietoa salasanojen luomistavoista, sekä tekemällä analyysi aiemmin nettiin vuotaneista salasanoista. Tutkimuksessa myös tarkastellaan olemassa olevia työkaluja salasanalistojen luomiseen ja selvitetään mitkä ovat näiden työkalujen vahvuudet ja heikkoudet. Edellämainitut toimenpiteet ovat pohjatyö jonka perusteella rakennetaan kevyt työkalu salasanalistojen luomiseen penetraatiotestausta varten. Jo tehtävää varten olemassaolevien työkalujen ongelmana on että ne luovat joko liian suuria tai pieniä sanalistoja ollakseen käytännöllisiä. Niistä puuttuu myös toiminnallisuus sanalistojen muokkaamiseen.
Työkalun tehokkuutta arvioidaan ja testataan wardriving menetelmällä Wifi-tukipisteistä hankituilla pmkid hasheilla. Myöhemmin hashejä verrataan työkalun luomiin sanalistoihin käyttäen apuna Hashcat nimistä työkalua ja tutkitaan löytyykö vastaavuuksia, ts. vastaako jokin työkalun luomista sanoista salasanaa jolla hash on luotu. Tätä kautta saadaan myös tilannekuva Wifi-salasanojen vahvuudesta Oulun kaupungissa.
Tulokset paljastivat että noin 6 % Wifi-tukipisteistä käytetään liian heikkoa salasanaa. Tämä löytö on linjassa aiemmin Oulussa tehdyn tutkimuksen kanssa, jossa kyseinen tutkimus osoitti että lähes 14.78 % laitteista puuttuu salasanasuojaus ja laitteet toimivat noissa tapauksissa avoimina tukiasemina. [1]

Avainsanat: Tietoturva, Linux CLI, penetraatiotestaus, wlan turvallisuus

TABLE OF CONTENTS

FOREWORD

I thank Panagiotis Kostakos for guidance and patience as finishing the thesis took a little longer than expected.

Oulu, October 25th, 2023

Janne Uutela

LIST OF ABBREVIATIONS AND SYMBOLS

CSDN	Chinese Software Developer Network. An online forum.
WPA	Wifi protected access.
WPA-TKIP	Wifi protected access Temporal Key Integrity Protocol.
PBKDF2	Password-Based key derivation function.
PMKID	Pairwise master key identifier.
EAPOL	Extensible authentication protocol over lan.
WLAN	Wireless Local Area Network
WEP	Wired Equivalent Privacy
MAC address	Media access control address
WLAN	Wireless local-area network
PTK	Pairwise transient key
GTK	Group temporal key
PMK	Pairwise master key

1. INTRODUCTION

Passwords are the principal method of protecting our online presence. Almost every service we use is being protected by a password and typically one person is using several different passwords daily for different services. Organizations may fortify their systems, but if the end-user or staff chooses weak passwords, it raises the question: just how secure is the system? A method to assess a system's security is to conduct a penetration test, essentially attempting to hack into it [2]. A way of doing a penetration test is to do a dictionary attack to guess passwords in the system. Existing password lists (i.e., wordlists) have been obtained when systems have been compromised, and the plain text passwords have been leaked out. One of the most famous wordlists is the rockyou.com password leak from 2009 [3].

The challenge with these leaked password lists is their applicability to services provided in less commonly spoken languages. Not all passwords are formulated in the English language, so how effective are these lists for such services? A Finnish user named Jussi might be content with his password, "Jussi70!", believing it is secure given its mix of capital letters, numbers, a special character, and its absence from the haveibeenpwned.com database. Cultural differences also play a role in how individuals select or craft their passwords. For instance, in the Philippines, it is common for people to incorporate month names into their passwords, while in Arabic nations, the use of mobile phone numbers as passwords is more prevalent [4].

The objective of this thesis is to examine the password formulation tendencies of individuals, factoring in linguistic and cultural variations, with a primary focus on the Finnish language. Subsequently, the goal is to develop a novel tool tailored for generating wordlists suitable for penetration testing (see tool repository in GiHub [5]). This tool is designed to be modular, facilitating the addition of support for other languages in the future, provided its effectiveness is established. The final phase involves testing this tool against actual password hashes acquired through pmkid harvesting and contrasting the outcomes with known leaked password databases such as rockyou or LinkedIn passwords.

Existing tools for testing weak passwords often come with limitations and lack the desired flexibility. While there are wordlist generators available, they tend to offer either overly specific or excessively broad wordlists. Additionally, word mangling tools are available, but their utility is contingent upon already possessing a wordlist. At present, one of the more effective approaches is to use leaked password lists, such as Rockyou. However, the Rockyou list predominantly does not cater to Finnish users, presenting its own set of challenges. Consequently, there are numerous weak passwords that are absent from the Rockyou list.

Using weak passwords can lead to severe consequences, especially if the compromised account holds critical information or has significant influence. Consider the Vastaamo case, in which more than 30,000 individuals lost their personal data to criminals [6]. This incident is not an isolated event; there have been recent cases, albeit not as extensive as Vastaamo. For instance, the official websites of the Oulu city were exploited to mine cryptocurrency [7], and the Finnish site liiketoimintasuunnitelma.com had plain text passwords exposed [8].

1.1. Cyber Attacks

As of September 2023, Russian cyber groups had already launched 3,000 cyberattacks against Ukraine. These groups are also increasing their capabilities and refining their tactics as time goes on. It would be naive to think that when the war in Ukraine is over, they would just stop doing this [9]. After the war in Ukraine is over, these groups might have their targets from somewhere else. Now that Finland has joined NATO and Russia has stated that it will take action, these actions could be cyber operations done against Finland. Even when Russian cyber groups are busy with operations in the Ukraine war, daily cyber threats are becoming the new norm in Finland. Companies in Finland are increasingly becoming the targets of cyber-attacks [10]. It is important to note that not all cyberattacks are orchestrated by Russia. Other state-sponsored actors, including China and North Korea, have also been involved in such activities. For instance, this year, North Korean operatives hacked into the computer systems of the Finnish defense industry [11]. By increasing the public knowledge about security, Finnish society can be made a bit more resilient against cyber threats. Weak passwords represent just one of the low-hanging fruits in the realm of cybersecurity that needs to be addressed. It is not only the average person or developers who commit security blunders; individuals in positions of authority, who should be more knowledgeable, also make such mistakes. For instance, a Finnish member of parliament once conducted official business over a publicly accessible Wi-Fi network [12].

2. RELATED WORK

In relation to this thesis, wireless local-area network (WLAN) security has been the subject of numerous prior studies. The emphasis of these studies has often been the security of the encryption protocol used. This is expected as encryption protocols are the main line of defence in WLAN communication. Unfortunately, the encryption protocol is not the only place where weaknesses can be found. The devices themselves could have flaws, using weak default passwords or even hardcoded. It can also be complicated for the average user to change the default password to a more secure password [13]. Ultimately, even the most secure device is only as strong as its human user. If the individual opts to use a weak password, such as "password," then the device's security is barely more robust than an open-access WLAN. This thesis primarily concentrates on general password security rather than specific WLAN security. Nevertheless, password hashes will be sourced from WLAN access points using a method known as wardriving. Subsequently, these gathered hashes will be tested against selected password lists to determine their vulnerability. Therefore, while the main focus is on password security, this research also delves into aspects of WLAN security.
One of the studies focused on encryption protocols was conducted using wardriving in a medium-sized city in Southwest Finland during the Covid-19 outbreak. People were forced to remote working, which led to an increase of new WLAN access points. The amount of new access points deployed was 50.2%. Surprisingly, the amount of devices using the new, most secure WPA3 protocol did not increase, almost at all. Most of the new devices were using the WPA2 protocol, which is still considered secure. However, the amounts of devices using WEP and WPA-TKIP, which have been vulnerable for a long time, did not change. The study also revealed devices left unconfigured and unencrypted, which are a serious threat to the network user's security [14] .
Another study about wireless security was done in the city of Oulu in Finland in the year 2019. This study was also carried out with wardriving. In this study, the amount of devices using WPA2 protocols discovered in the city centre of Oulu was 71.11%, devices using WPA-TKIP protocol was 10,60% and WEP protocol in 0.71% of the devices. Surprisingly, 14.78% of the discovered devices had open endpoints, and 2.8% used an unknown protocol [1].

2.1. The Ethics and Legality of Wardriving in Finland

Wardriving itself is simply a passive way to listen to the packets that the access points are constantly broadcasting around them. Wardriving as a term may sound a bit shady, but it is well-recognised and legitimate tool used by security professionals and hobbyists. The line where it becomes illegal is when someone uses wardriving to reveal a security weakness and use it to gain unauthorized access to the network. In this case, when cracking a password and using it to gain access to the network. Because wardriving is passive, the target networks have no way of knowing that they are being "scanned" or listened to. This raises an ethical question: is consent required from the network owner for such actions? Such a task would be nearly impossible, as one would need to know in advance who resides in the area and whose devices might

fall within the scanning range. If wardriving were to be declared illegal, legitimate "scanning" activities would likely cease. This would significantly impede studies aimed at improving WLAN security and uncovering vulnerabilities [13]. However, it is worth noting that malicious actors might remain undeterred by the legal status and could continue their activities as before.
Furthermore as discussed in [13], the position of Finnish law regarding wardriving is clear-cut. Under Finnish criminal law, using an unsecured open WLAN network is entirely lawful. This permits activities such as web browsing, video streaming, and online gaming. However, it becomes illegal if one accesses other devices, internal networks, or services on that network without the consent of the network owner.
In the context of scientific research where WLAN data is collected and stored, the EU General Data Protection Regulation (GDPR) provides similar clear guidelines. Even though wardriving sessions might capture MAC addresses and SSIDs, which are considered personal data that could potentially be paired with other databases to identify a device's owner, it is practically impossible to do so. This is because, while MAC addresses are unique to each device, they cannot be directly linked to an individual's name. Furthermore, there are no databases that associate any personal information with a specific MAC address [13].
Therefore, the collection and storage of WLAN data for scientific research serving the broader public interest is permissible, as long as the data's confidentiality and integrity are ensured throughout every phase. Given that the security of wireless broadband falls strictly under the purview of the connection's owner, educational institutions play a pivotal role in fostering awareness and knowledge. This includes emphasizing the vulnerabilities associated with weak network configurations. Nevertheless, the data must be safeguarded and retained in a way that shields it from unauthorized modifications, loss, or destruction as described in GDPR.
This thesis adheres to the code of ethics outlined in the study of wireless security in Oulu [1] and also in [13]. This code provides valuable guidelines for wardriving and should be adhered to when conducting research on wireless security using this method. The main parts are as follows:

- Do not connect to the access points.
- Obey traffic laws. This is primarily for those who do their wardriving with car.
- Obey private property and no trespassing signs when collecting.
- Do not use your data for personal gain. The hashes will be deleted, once the thesis is finished.
- Be like hiker, take only pictures and leave only footprints. Detect the data that the access point is broadcasting and then move on.

[1]

2.2. About Password Strength

Studies have been done about password strength and how the guidance systems will affect how strong passwords users are creating. There is evidence that a lack of guidance will lead users to create weaker passwords and stronger ones with guidance. The guidance is just a visual aid to see the password strength. The guidance algorithms do not even have to be perfect, one algorithm gives weak points for passwords e.g., 1234567, iloveyou, luke33, with luke33 scoring 37 points. Medium strength was achieved with Luke23, scoring 50 points. Strong password candidate, maggie9876543 scored 71 points, even when it is a combination of a first name and a numeric pattern. Granted, the pattern is not the most popular [15]

2.3. Cracking Passwords

For passwords composed solely of numbers to be secure, they must be considerably long. Brute force cracking is a method where every combination is tried. This suites well for numeric passwords where there are only a few choices per character. Dictionary attack utilizes wordlists, which then are compared against the password hashes, one of the most popular wordlists being the Rockyou wordlist. A hybrid attack is a combination of the last two, the word from the word list and a mask attached to the word which is tried against all combinations. Rainbow table attack is a method where a table of passwords and their hashes are constructed. It is fast as it only has to check if the hash exists already in the table. The downside of it is that it requires a lot of space and can’t possibly have all the hashes and plain passwords stored there [16]. A dictionary attack requires a lot less space as the hashes are created on the fly to compare against the password hash. It is also slower as the hashes are not ready and created. In this thesis, the emphasis is placed on the dictionary attack, supplemented by some brute force methods, especially targeting numerical passwords.

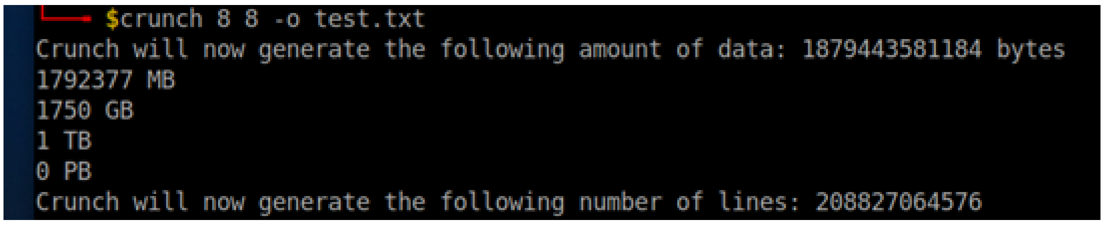
2.4. The Israeli Study

Wifi password strength has been studied earlier by the Israeli security researcher Ido Hoorvitch. In his research, which was done in 2021, he gathered 5000 PMKID hashes in Tel Aviv using a tool called hcxdumptool. The hashes were collected from wifi access points. After collecting the hashes, he managed to crack 70% of them in his lab using a powerful cracking rig. This was done with hashcat mask attack, which is just a term where hashcat has been given a set of instructions which characters to use and where while cracking the hash using brute force. In this case, the mask attack was used to test if there were phone numbers used in the passwords. In Israel, the phone numbers start with number 05 and the phone numbers are ten digits long, so he could give hashcat instructions that the first two characters of the password are 0 and 5. And for the rest of the eight characters, use only digits. Then all the possible combinations are tried, using these instructions. The result was that 2200 passwords were cracked by using phone numbers. After this, he did a dictionary attack with the Rockyou password list. With the dictionary attack, he was able to crack +900 additional passwords from the hashes [17].

3. EXISTING PASSWORD LIST GENERATORS

3.1. Crunch

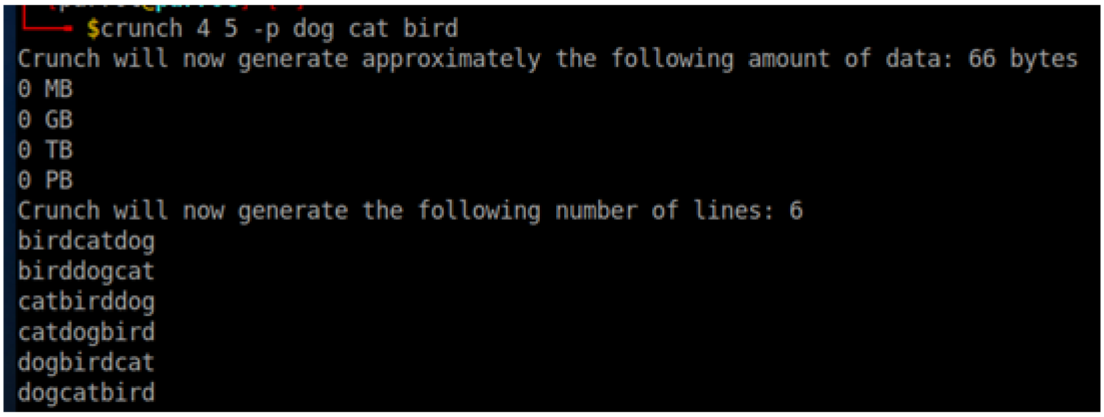
Crunch can be used to create a password list from desired characters. The basic usage is quite simple, crunch <min-len> <max-len> [<charset string>] [options]. You give the minimum and maximum length of the word, the characters you want to use for creating the list and some options if you so desire. If you do not specify the characters, crunch will use only the lowercase characters [18]



```
└──╼ $crunch 8 8 -o test.txt
Crunch will now generate the following amount of data: 1879443581184 bytes
1792377 MB
1750 GB
1 TB
0 PB
Crunch will now generate the following number of lines: 208827064576
```

Figure 1. Crunch creating a word list

As the Figure 1. Shows the list is enormous even with just 8 lowercase characters. The final list would be in this case over 1792 GB in size containing nearly 209 billion words.



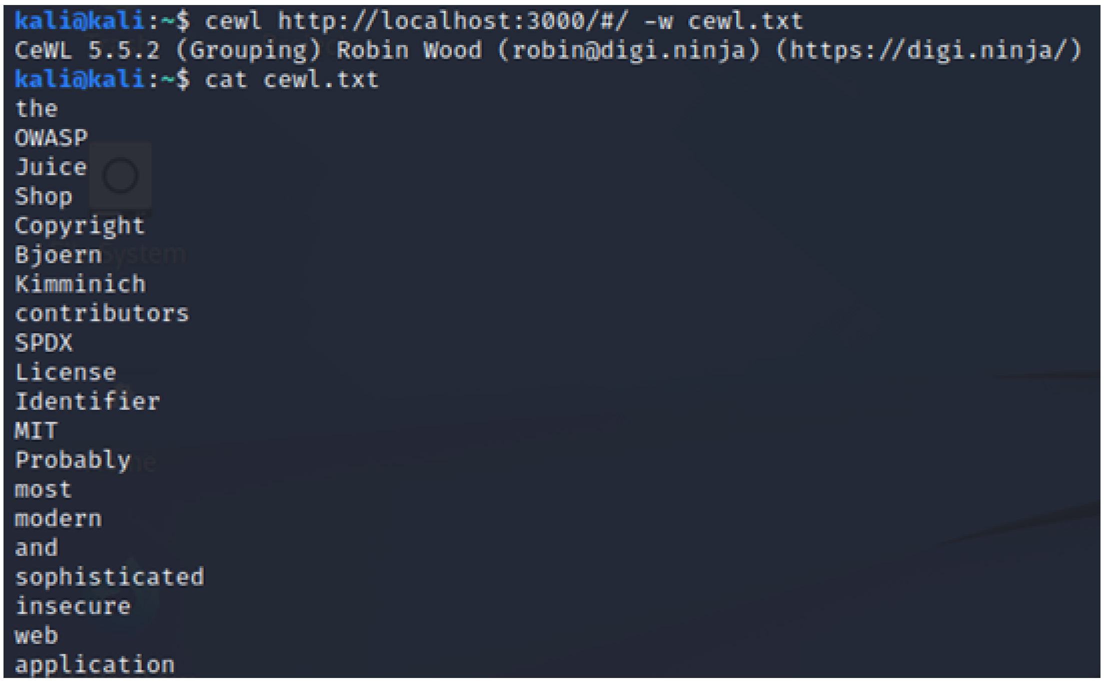
```
└──╼ $crunch 4 5 -p dog cat bird
Crunch will now generate approximately the following amount of data: 66 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 6
birdcatdog
birddogcat
catbirddog
catdogbird
dogbirdcat
dogcatbird
```

Figure 2. You can also use words, which might be suitable for penetration testing in some cases

The Crunch might be the tool to use when one password needs to be brute forced. Even then the password must be quite short for this to work even with an efficient computer. In the figure 1 Crunch was tested with 8 lowercase characters and the list was very large. If the password would also contain uppercase characters, numbers and special characters, the list would be considerably larger. The positives about Crunch are that it is very easy to use, although it has quite a few options. The negatives would be that the lists it creates are quite big and possibly not the most suitable for quick penetration testing. For brute forcing, this might be an option but again it creates a very large list. Crunch is also capable of creating wordlist that are not completely random, see figure 2

3.2. Cewl

Cewl is a Custom Word List generator that extracts words from a target website. These extracted words can subsequently be used as base words for password cracking [19].



```
kali@kali:~$ cewl http://localhost:3000/#/ -w cewl.txt
CeWL 5.5.2 (Grouping) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
kali@kali:~$ cat cewl.txt
the
OWASP
Juice
Shop
Copyright
Bjoern
Kimminich
contributors
SPDX
License
Identifier
MIT
Probably
most
modern
and
sophisticated
insecure
web
application
```

Figure 3. Scraped and printed words accuired by cewl

In Figure 3, words have been scraped from locally running website, the Juice shop, from the Open Web Application Security Project (OWASP). The tool is very easy to use, you basically just need to give the target address and the file where the results are written.
The service or website where the password is created has an effect on what words are selected in the password. End-users sometimes select words associated with the service in their passwords [3]. The tool is easy to use and can quickly grab some base words to your dictionary for penetration test. Cewl seems especially handy if you are performing a penetration test on a website or a business that has a website. You grab the words with the tool, add some additional words if needed, then mangle or modify the words and see if anyone is using those words.

3.3. Hashcat

Hashcat, rather than generating a word list, has a distinctive feature that allows users to manipulate and enhance their current wordlist in versatile manners. This not only diversifies the wordlist but also substantially increases its size. Additionally, this functionality can be employed directly during password-cracking attempts through a rule-based attack. To evaluate this feature, I utilized the `best64.rule`. I initiated a wordlist with three terms – "one," "two," and "three" – saved in a file named `test.txt`. The command to execute this was:

```
hashcat test.txt -r /usr/.../hashcat/rules/best64.rule --stdout > result.txt
```

This mangled the words in test.txt and saved in result.txt. The word one was modified to

- First letter capitalized. One
- The word capitalized. ONE
- Added digits to the end. E.g One0, one11 etc. Most of the digits at the end were between 0-99, not all the combinations though.
- Reversed word. Eno
- Rotated word. Eon
- Duplicated word. Oneone.
- Also, other modifications. e.g changing 1 letter, duplicating one or more letters.
- Unrelated words. 123, man, dog.
- To a lot of nonsense words, e.g os, oa, o, y, r, oo, etc.
- In the result list there were also the plain word "one" six times, which seemed like a waste of resources.
- Empty space, four lines. Again wasting resources.

The `best64.rule` seemed very good, even if it seemed to waste resources by bloating the wordlist with duplicates and nonsense words. Investigating further into hashcat rules, revealed that they had even a leetspeak rule. The word "one" was transformed into the expected 0ne, on3 and 0n3. It also had the word one repeated 14 times in the result file, which is a waste of resources. It was also missing other combinations that you might expect from leetspeak rule and the word one, e.g., on€. Hashcat wiki has detailed explanations of how they mangle words using the rules and the rule-based attack [20].

4. STATISTICS AND PASSWORD CREATION STRATEGIES

4.1. Length of the Passwords

The most common password length is 8 characters long. This is true for both wifi passwords and passwords to different services like LinkedIn. The share of exactly 8 characters long wifi passwords is 30,68% and for LinkedIn the share is 29,24%. For wifi passwords, 85,6% of passwords have a length between 6-12 characters. For LinkedIn, the percentage is 94,54 of passwords between 6-12 characters long [21]. There is not much point testing for passwords that go much beyond this length, as there are only about 5% of words that exceed this length, and they are harder to guess, making the required wordlist longer.

4.2. Structure of the Passwords

Using only digits is very popular with Chinese users. In India, Japan and UK people prefer using lowercase letters. Another popular structure is adding digits after the lowercase letters. Also using English words is very common in other than natively English-speaking countries too [22].

Wordlist	digit only	lowercase only	lc + d	lc + s	lc + d+s	combined
CSDN	45,06 %	11,68 %	35,60 %	0,42 %	2,04 %	94,80 %
Rockyou	15,93 %	41,68 %	33,17 %	1,64 %	1,44 %	93,86 %

Table 1. Password statistics (d = digit; s = symbol; lc = lowercase).

Table 1 displays the prevalence and structure of passwords in the respective services. In the CSDN (Chinese Software Developer Network), 94.8% of passwords are represented, while for Rockyou, it is 93.86%. Furthermore, when considering the possibility that one or more letters in these passwords could be in uppercase, the numbers escalate to 99.84% for Rockyou and 99.36% for CSDN [23]. Of course, we cannot blindly rely on these statistics and old password leaks. People are today more aware of online security, and I would imagine, that the passwords used today are more complex in general than they used to be.

4.2.1. What Words People Select in Their Passwords?

Some popular choices are first and last names, nicknames, loved ones and famous people. Also, animal and sport related words and places seem to be among the popular word choices for passwords. There are cultural differences in the choices. The most popular words in India were the first names, which was not very popular in UK. One of the popular choices in UK, an animal word, was one of the less popular choices in China. In general, people prefer personal words, like names, birthdays and dates in China and India. Generic words, like place names and animal words are preferred in UK [22]. Swear words and profanity can also be something that is used in passwords, also months, weekdays, seasons like summer and winter and professions like teacher or doctor [24].

4.2.2. Keyboard Patterns

Probably the most known keyboard pattern is the word qwerty. Another widely used keyboard pattern is 12345678. The reason why they are being used in passwords is that they can be easily remembered. The patterns seem to be random and might give the user a false sense of security about the password strength. The proportions of passwords created using keyboard patterns depend on the culture and site where they are created. On average, the proportion of keyboard patterns as passwords in China is around 14 %. The number for English passwords using keyboard patterns is around 7 % [25]. The majority of the passwords created using patterns consist completely of the pattern. When an additional word is added to the pattern, it usually is the first name, last name, date or a word associated with love [25].

4.2.3. Popular Patterns

Most popular patterns seem to be once again the digits. Leaked passwords from Gmail, which had been filtered to find the passwords consisting of patterns was put in the top 10 order, where the number 1 was the most frequently used.

popularity	digits included	only digits excluded
1.	123456	qwerty
2.	123456789	zaq12wsx
3.	12345	qwerty123
4.	qwerty	asdfghjkl
5.	12345678	qazwsx
6.	1234567	zxcvbnm
7.	111111	qwertyuiop
8.	123123	aaaaaa
9.	1234567890	asdfgh
10.	000000	asdasd

Table 2. Patterns found in passwords.

The top 10 of the patterns, see table 2, covered 35.42% of all the pattern-based passwords when digits were included in the results. When digits were excluded the top 10 covered 7.76 % of the passwords using patterns [25].

4.3. Special Characters

Including special characters in passwords is not particularly popular. From wifi passwords, only 5,64 % include a special character. LinkedIn users include special characters in 6,25% of the passwords. Some of the special characters are a lot more popular than the others [21].

	Wifi	%	CSDN	%	LinkedIn	%	Yahoo! Voices	%
1.	@	31,17	.	34,57	@	22,96	!	26,70
2.	.	16,82	@	25,43	!	17,61	@	21,29
3.	-	15,73	!	10,92	.	11,68	_	12,37
4.	space	11,45	*	9,19	*	8,43	$	12,12
5.	_	8,83	_	7,81	#	7,84	#	10,50
6.	#	5,52	#	6,62	_	7,78	*	9,79
7.	!	5,18	+	5,47	$	7,55	-	7,11
8.	$	4,30	/	3,36	-	6,07	;	4,69
9.	?	4,20	$	3,32	space	3,81	&	4,49
10.	*	3,78	?	2,69	&	2,77	.	3,55

Table 3. Special characters.

The table 3, special characters, showcases the most used special characters by service in popularity order. The top 10 most used characters in LinkedIn cover 96,5 % of the used special characters in LinkedIn passwords.

4.4. Mangling

Typically, people do not use plain words as passwords as they are too easy to crack. Instead, they modify the password somehow to make it more unguessable. One of the most common is adding digits at the end of the word like password1. Other mangling techniques are [26].:

1. Uppercasing the first character. E.g password -> Password
2. Uppercasing entire word. Password -> PASSWORD.
3. Uppercasing n number of characters - PassWOrd.
4. Duplicate character n times - paasword.

5. Append character at the beginning or end - passwordi
6. Delete character at index - passwod
7. Insert character at index - passw!ord

4.4.1. Leetspeak

Leetspeak is a word mangling technique, that replaces one or more of the characters of the word with a number or special character, resembling the original character's appearance (e.g., password turned into p@ssword, pa$$w0rd or p45s\/\/0rl) [2]. If using leetspeak in wordlists, the length of the wordlist will increase, and it will at least slow down the attacker but does not completely prevent the cracking.

5. STUDY OF THE LEAKED PASSWORDS

This thesis will do a quick investigation on a couple of leaked password lists, to find out what would be the most effective strategies to apply for the tool to cover as many passwords as possible for the shortest wordlist possible. Keeping the wordlist reasonably sized is important as if you are doing a quick penetration test, you probably do not have the time to wait days or even weeks for your wordlist to be checked against the passwords. Not everyone has access to a powerful cracking machine.

This thesis will also investigate if there are Finnish passwords among the leaked passwords and if we can find out something about the popularity of Finnish words in the passwords. We must keep in mind that only a fraction of the world's population speaks Finnish, and the results will be at best only directive as the lists we are using are not from Finnish sites. There exists a password leakage with Finnish plain text passwords from liiketoimintasuunnitelma.com but unfortunately, we were not able to get this list to be studied.

Initially, this thesis will explore the composition of passwords, examining the proportion that are purely numerical versus those that are exclusively alphabetical. We will also investigate the percentage of passwords that incorporate numerical segments, uppercase letters, or special characters. Additionally, emphasis will be placed on determining the positions of these numerical, uppercase, or special character components within the passwords. After this, the focus will turn to what kind of words are commonly used in passwords. The attention will be on the common poor choices for passwords, e.g. first and last name, but also on the words that are known to be used in passwords, e.g., place names, animal names and gender. As any word can be used in a password, in this investigation the focus is to try and use the most popular.

The words for analysis will be selected based on an estimate of their potential popularity. However, this approach is inherently directional. The chosen words might not ultimately be the most prevalent in passwords. Given the resource constraints of this thesis, it's impractical to scrutinize every conceivable word. Because any word can be used as a password, there will be no distinction made e.g., for place name category, where valid place name choices could be a city name, country name or a street name etc. Same goes for animal names category. Valid animal names could be dog, or Finnish Spitz, which is a dog breed. The estimate in this case is that the dog, as an animal name, would be more popular than Finnish spitz.

5.1. The Investigation

The LinkedIn list used in this investigation was +60 million passwords and it is publicly available. The rockyou list is also publicly available and contains more than 14 million passwords. In the investigation, we used regex patterns to find the words from the wordlist. The regex used is as follows:

```
1= `^\d{3,10}$`
2= `^\d{11,}$`
3=`^[A-Z][a-z\d\p{M}]*$`
4= `^[a-z]+$`
5= `^[a-z][a-z\d\p{M}]*$`
```

```
6= ^[a-z]+[A-Z]+[a-z]*[A-Z]*[\d\p{M}]*$
7= '\w{1,}\D\d{1,4}$'
8= '\w{1,}\D\d{1}$'
9= '\w{1,}\D\d{2}$'
10= '\w{1,}\D\d{3}$'
11= '\w{1,}\D\d{4}$'
12= '^[a-zA-Z]+\d{1,4}$'
13= '^[a-zA-Z]+\d{1,4}\W+$'
14='^[a-zA-Z]+\W+$'
15='^\d{1,4}\D[a-zA-Z]+$'
16='\W'
17='^\W'
18='\W$'
```

regex	Structure	LinkedIn %	Rockyou %
1	Only digits (3-10)	6.841106	14.94381
2	Only digits > 10	12.783801	1.415780
3	first letter uppercase, rest lowercase, digits or marks or nothing	7.133062	2.423407
4.	Only lowercase	17.927611	25.976207
5.	first letter lowercase, rest lowercase, digits or marks or nothing	54.416298	62.983729
6.	First letter lowercase, one or more uppercase letters in 1-2 spots in the word. Optional digits or marks at the end.	0.523113	0.250690
7.	Words that are not completely digits, ending at 1-4 digits	39.921558	36.902435
8.	Words ending exactly to 1 digit	7.081994	8.193767
9.	Words ending exactly to 2 digits	13.482045	14.311126
10.	exactly to 3 digits	6.031993	6.006445
11.	exactly to 4 digits	13.325526	8.391098
12.	Lowercase or uppercase (no digits in the middle) and ending in 1 to 4 digits	34.263979	32.885075
13.	Words ending to 1-4 digits and to one or more special characters	1.121601	0.475008
14.	Lowercase or uppercase + one or more special characters at the end.	0.383364	1.004260
15.	Starting with 1-4 digits, not completely numeric and containing upper or lowercase characters at the end	3.446201	2.911668
16.	Contains a special character somewhere in the word	6.395914	6.020952
17.	Special character at the beginning	0.707756	0.692800
18.	Special character at the end	2.516776	2.314410

Table 4. Password structures.

From table 4, we can see that the most fruitful structures would be only lowercase words and lowercase words ending from 1 to 4 digits. Also, the digits only could be considered a fruitful part as they are easily brute-forced especially if offline cracking with a weaker hash like MD5. Passwords using uppercase in the middle of the word seem to belong to the minority. The same goes for using special characters, but if they are used, the most frequently used placement is at the end. When using digits in passwords that are not completely digits, the digits are more likely to be inserted in the end of the password rather than in the beginning. One thing to notice when interpreting the results is that when searched with \w, the word can also contain digits in the middle. I noticed that my statistics seemed to be a bit different from the statistics presented in the table 1. Especially the amount of lowercase passwords in the Rockyou list. This difference could be because they had a rockyou list that contains 32 million passwords, and this thesis uses the default list included in Kali distros, containing 14 million passwords. Duplicate passwords are removed from the Kali Linux list to make it more efficient.

5.2. Substrings

5.2.1. First Names

Next, was to study what kind of words are used in passwords and if there are Finnish words among them. This was done by using regex again and the findings are returned when a word contains the word that was looked for. There are chances that the original word has nothing to do with the searched word, but it too is presented in the results. For example, when searching with the word “Jane”, the results could contain the word “Ninjanetwork”, which has nothing to do with the name “Jane”.

name	Rockyou amount	%	LinkedIn amount	%
John	21149	0.147437	62933	0.103978
Mary	12120	0.084493	40232	0.066471
Jussi	51	0.000356	430	0.000710
Jaana	47	0.000328	637	0.001052

Table 5. First names.

Names are often found in passwords, and as such, a selection of names was made to gauge their frequency in the password lists. The occurrences of each name can be seen in Table 5. These counts can serve as benchmarks when assessing the popularity of other words in passwords. The selected English names were John and Mary. Also, Jussi, which is a Finnish first name for males and Jaana, a Finnish female first name. Both should be common names in Finland. For Finnish names, the selection of the names had to be done more carefully, so that there would not be a lot of hits from other English names or words. For instance, a Finnish female name Anni, probably is used as a first name outside of Finland too. Also using the word Anni will get hits from Giovanni, cannibal, Britannia, etc.

When investigating the returned results, it was noticeable that the word "boy" was often attached to the name John. Exactly 274 times in the LinkedIn list and 104 times in the Rockyou list. Is adding gender details how common when creating passwords? It certainly is easy to remember and adds length to your password, but it does not add very much security to the password, as it is guessable.

5.2.2. Gender Additions

My exploration of first names led me to discover gender-related additions within words, prompting a subsequent investigation into their prevalence in password choices.

word(translation)	Rockyou amount	%	LinkedIn amount	%
boy	54224	0.378015	102680	0.169647
girl	53247	0.371204	73794	0.121922
woman	1497	0.010436	5890	0.009731
lady	15804	0.110175	31532	0.052097
sister	1801	0.012555	4689	0.007747
brother	1286	0.008965	3672	0.006067
mies (man)	1635	0.011398	4144	0.006847
poika (boy)	18	0.000125	347	0.000573
tyttö (girl)	0	0	1	0.000007
nainen (woman)	8	0.000056	75	0.000124
sisko (sister)	54	0.000376	324	0.000535

Table 6. Gender.

When inspecting table 6, we can observe that the word "boy" was more popular than the names "John" or "Mary". The Finnish word for man, "mies" was a lot more popular than the example names Jussi and Jaana. Finnish word for boy, poika and the word for girl, "tyttö" was rarely used in passwords. The word "man" was left out because it had a lot of additional hits like Manuela, Manchester etc., but it too had a lot of hits. The estimation of the use of gender in passwords is that it seems to be a marginal practice and usually the words were attached to another word. If targeting Finnish passwords, the word "mies" is somewhat popular, but when investigating the returned results there are a lot of English words that happen to contain that word, e.g dummies, roomies, yummies etc. The gender additions cannot be ignored entirely when testing for weak passwords. There are enough hits that suggest that this should be taken into consideration when designing the tool.

5.2.3. Place Names

Names of the countries or cities were not particularly popular, see table 7. The popularity of places seemed to be below the popularity of a single first name.

word(translation)	Rockyou amount	%	LinkedIn amount	%
England	665	0.004636	1516	0.002505
London	1181	0.008233	8219	0.013579
Italy	595	0.004148	2482	0.004101
Suomi	48	0.000335	540	0.000892
Finland	80	0.000558	532	0.000879
Helsinki	21	0.000202	303	0.000501

Table 7. Place names.

5.2.4. Animals

word(translation)	Rockyou amount	%	LinkedIn amount	%
Bear	15965	0.111298	47945	0.079215
Dog	33151	0.231108	143302	0.236763
Tiger	6160	0.042944	23220	0.038364
Bird	7480	0.052146	30528	0.050438
Kissa (Cat)	461	0.003214	1287	0.002126
Koira (Dog)	46	0.000321	639	0.001056

Table 8. Animal names.

Using animals is quite popular, see table 8, but it depends on the animal. Cat, which was left out because of the additional hits to e.g Catarina, Cathy etc. But it seemed to be near the popularity of the dog word in passwords. The same goes for cat, “kissa” in Finnish and dog, “koira” in Finnish.

5.2.5. Last Names

lastname	Rockyou amount	%	LinkedIn amount	%
Smith	4498	0.031357	17384	0.028722
Johnson	1291	0.009000	4097	0.006769
Williams	1334	0.009300	2856	0.004719
Korhonen	1	0.000007	9	0.000015
Virtanen	1	0.000007	17	0.000028
Nieminen	1	0.000007	4	0.000007

Table 9. Last names.

Popular Finnish last names like Korhonen or Virtanen get only a very few hits from the Rockyou and LinkedIn list, see table 9. It seems that Finnish people do not use their last names in their passwords. Popular English last names like Smith and Williams get a little more use, but they too are used very rarely in passwords. Leaving the last names out of the base wordlist appears to be the appropriate decision, given the abundance of last names, which tend to yield only minimal results.

5.2.6. Profanity

word	Rockyou amount	%	LinkedIn amount	%
fuck	17480	0.121859	24939	0.041204
shit	8259	0.057577	21311	0.035210
perkele	18	0.000125	160	0.000264
saatana	4	0.000028	35	0.000058

Table 10. Profanity.

Users definitely use profanity in their passwords, see table 10. According to the article [24], fuck was the second most popular swear word used in passwords. The most popular swear word was left out, “ass”, as it is short and ambiguous and will give a lot of false hits (e.g., password, grass, etc). Finnish swear words did not get particularly many hits, but enough so they can’t be completely ignored.

5.2.7. Professions

word (translation)	Rockyou amount	%	LinkedIn amount	%
Nurse	1467	0.010227	6189	0.010225
Teacher	492	0.003430	3496	0.005776
Police	701	0.004887	2921	0.004826
Opettaja (teacher)	0	0	4	0.000007
Poliisi (Police)	2	0.000014	15	0.000025

Table 11. Professions.

We conducted several searches using profession-related words that were initially assumed to be popular. The findings indicate that incorporating professions into passwords is not a common practice, as we obtained only a limited number of hits, as shown in Table 11.

6. TOOL CREATION

The idea for this study and creating this tool came when studying cyber security in one of the practice services, Hackthebox, and a necessity to create a wordlist arose. The use of a large wordlist would not be ideal e.g the Rockyou list to test the security of an admin password. As the point would not be spending all day cracking the password. The tools that could be found at that time were Crunch and Cewl. Cewl was almost what was needed, but it had a downside that the list it created could not be modified e.g with the leetspeak rules or something similar, another way than manually modifying the list, which was out of the question.

6.1. Why Golang Was Chosen

The choices for building the tool were Python and Golang. Both are fast to write and have libraries for easy command line tool creation. Golang's advantage was that it was advertised as fast and almost as easy to write as Python. Additionally, Golang was advertised to have most of the needed libraries as default, no need to get things from third parties. Because the tool is for quick testing, so even the wordlist creation should happen fast.
Later in the process of creating the tool the benefits of golang became clearer. It is memory intensive to process large wordlists and the golang pointers prevent unnecessary copying of the lists as they are passed to functions for processing. If there is no need for copying, then just use pointers. Golang also has a good garbage collection, that certainly does not hurt to have with this kind of application.

6.2. Base Words

Base words are set as text files and the users can freely edit them or add completely their own files to the tool as base files [5]. The base word files will be accessible with the language flag, with Finland "fi" being currently the default choice if the language is not defined by the user. You could define your own base file e.g some_base.txt and access that file with the language flag "some". Inside the categories, you can use whatever words you desire.

The categories in the base files are:

- male names
- female names
- miscellaneous words
- common adds. E.g 123 added in the end of the word
- marks. Some popular marks that are used in the words.
- less common adds. Tagged as category 1

- less common adds. Tagged as category 2.
- profanity
- patterns. E.g qwerty, 123456

Because names and particularly first names were common in passwords, 1000 most popular Finnish male first names and 1000 most popular Finnish female first names were chosen in the base words. The last names were left out as they were not very popular. The names can be found from [27] in popularity order. Also, the most popular last names can be found from there in popularity order. Female and male names were decided to keep separate, because of the gender additions in passwords. Male names ending with female gender adds seemed a bit far-fetched to have enough hits. This kept the final list a bit smaller. For miscellaneous words, random words were chosen that could be used in forming weak passwords (e.g., animal names, hobbies, sports, food, popular brands, etc). Even when this list was for Finnish passwords, a fair number of English words were included in also, as the English language is so commonly spoken in Finland. In addition of the base words, the tool can create dates in different formats (e.g., -d “1990 50 . dmy” will create dates in format DD.MM.YYYY starting from year 1990 + 50 years).

6.3. Commands and Strategies

There are four commands in the tool. The base – command is for adding the words you choose to your output list. The add – command adds the word at the end of your words in the output list. You can use the base word lists with these commands too if you wish, when building your wordlist from ground up or use whatever words you choose. The third command is the quick – command, where you can find a few ready-made strategies:

1. Dates from the past 50 years in different formats + patterns.
   - Dates and patterns seemed to be successful. By default, this strategy will give the dates from past 50 years in a few different formats, e.g DDMMYYYY, DD.MM.YYYY and DD/MM/YYYY.
   - With add and base commands, you can choose the time span and the format as you wish. E.g YYYY-MM-DD or DD??YYYY?MM or something like that.
2. Names with lower and uppercase, marks, and common addings.
   - Uppercase was the most common at the first letter of the word, if used at all. This will provide the upper and lowercasing of the word, word with marks and the word with the common adds. E.g: John, john, John!, john!, John1234, john1234 etc.
3. Names with lower and uppercase with birth year (past 50 years). E.g john08, john2008 + additional adding’s.

4. Names with lower and uppercase with birth years and marks. + additional adding's
5. Miscellaneous words from base language file combined with marks and common addings.
6. Miscellaneous words from base language file combined with birth year (50 years) + additional addings.
7. Miscellaneous words from base language file combined with years, gender adds and marks.
8. Profanity from base file with marks and common addings.
9. Profanity from base file with birth year (50 years) + additional addings.
10. Do all the previous at one go.

The fourth command is mangle. User given wordlist is mangled and saved in the output file. The current mangling options:

1. Uppercase and lowercase your list.
    - Simple first char uppercase or lowercase.
2. Light mangle
    - Uppercase and lowercase word. Sam -> Sam, sam or sam -> Sam, sam
    - duplicate. E.g. Sam -> SamSam. Also Sam2000 -> SamSam2000, as the most common place for digits is at the end.
    - reverse. Sam -> maS. Digits handled as in the duplicate method.
    - Uppercase entire word. Sam -> SAM.
3. Mangle your wordlist with leetspeak.
    - E.g name Sam transformed into $am, s@m, $@m.
    - The leetspeak mangle will be applied to your provided wordlist (-L flag) and written into the output file.
    - I did not provide the possibility for the user to add their own leetspeak mangle rules, as there was a chance that the user would choose rules that end up in endless cycle.
    - Use case when user might consider mangling their wordlist, could be when targeting and testing e.g admin password and the wordlist is not yet very large. Mangling the list will make the list grow significantly larger.

7. TESTING THE TOOL

The idea how to test this tool came from Ido Hoorvitch's study, see section related work or [17] where he gathered 5000 password hashes using hcxdumptool, monitor mode capable antenna and a laptop to collect the hashes. After collecting the hashes, he then cracked the hashes offline in his lab using a more powerful computer for the cracking process. Trying to crack the hashes with the laptop is not ideal as laptop would not have enough computational power for this. In his cracking rig, there was (8 x QUADRO RTX 8000 48GB GPUs) to speed up the process.

7.1. PMKID Attack

PTK = Pairwise transient key
GTK = Group temporal key
PMK = Pairwise master key
MAC-AP = Access points mac address
MAC-STA = Client's mac address (in this case yours)
AP = Access point, router etc.
Client = The device that wants to connect to AP

The concept of PMKID attack is not new. It used to require a 4-way handshake, which goes crudely like the following [28]:

- **AP:** broadcasts EAPOL frame that I am here, in case you want to connect. Here is also a random number for you so that you can create the PTK
- **Client:**Thank you, here is my random number so that you can create your PTK
- **AP:** Here is you GTK for connection.
- **Client:** Thanks, I got and saved the GTK and I am ready for connection

The fourth message (from client) is just a confirmation from the client that they got the GTK and it is saved. All this was needed before to crack a PMKID. The problem was that it required a client to enter the radius of the AP and start authenticating. If the client was already connected, there was no need for the 4-way handshake, so you would have to wait until a client wanted to connect again, or you would have to kick the client out of his network so that they needed to re-authenticate.
Hashcat developers accidentally discovered while researching WPA3 security, that the PMKID can be calculated even from the 1st message and there is no need for the 4-way handshake. PMKID hash is a hash created with HMAC-SHA1-128 algorithm and the routers use it for roaming capabilities. The algorithm wants PMK, which we don't initially know and a concatenation of the word "PMK Name" and MAC-AP and MAC-STA. The part "PMK Name" we know because it is constant, MAC-AP we know because it is broadcasted in the first EAPOL frame and MAC-STA is the mac address. The PMK is created with PBKDF2 which wants values: password, which we don't know, SSID, which is the name of the network we get in the first frame, and a number of iterations the PBKDF2 function should be run, which to my knowledge is always

the same 4096. So everything except the password is known after the first EAPOL frame. After the PMKID hash is grabbed, possibly in the first frame already, cracking it is only a matter of creating first the PMK with the PBKDF2 function and using the PMK to create a comparable PMKID and see if it matches the original PMKID. And if it does, then the password is cracked [17]

7.2. Gathering the Hashes

The plan was to collect 1000-1500 hashes, which should be sufficient for testing the tool. The more hashes are collected, the more reliable knowledge would be gained from the password habits of people in Finland and Oulu. However, since there was no access to a very powerful cracking machine, there was no point in collecting a lot of hashes, because running the wordlists against the hashes would just take too long. The setup to this was simple, an old laptop running parrot OS in VMware virtual machine connected with wifi adapter Alfa awus036ach, capable of monitoring mode. The laptop was nothing special, -i5 7200U processor with 8gb ram, nor does it need to be. The gathering was done by using a tool called hcxdumptool. The tool can be used to set it listening passively for wifi access points and the packets they are constantly broadcasting. For this you need to use options –disable_deauthentication –disable_client_attacks, otherwise it will be kicking users offline from their networks for re-authentication.
A script was created to conveniently start and stop the collection with a click of a button. The entire setup was compactly stored in a backpack, ensuring that from an external perspective, it was virtually indistinguishable between being a gym bag or a setup designed for hash collection. Remote desktop was also set up for laptop and phone, so that the gathering could be started and stopped at desired point by using phone without taking the laptop out of the backpack. Determining the timeframe required to accumulate 1000 hashes and the extent of the area over which this collection would need to occur proved to be challenging.
The convenient starting and stopping would be important so that the collection of the hashes could be done in parts, so that the same hashes would not be collected again every time a new gathering trip was started. In the end,only two trips was needed to collect the hashes. In day 1, hashes were collected from Oulu from starting point A and finished on point B. On the second day, collecting started at the point B and finished on point C. After this 1194 hashes were collected and combined into a one file. Gathering the hashes took around 2 working days, and the distance is a bit hard to estimate but it was many kilometers zigzagging the streets around and in the center of Oulu.

The data that `hcxdumptool` captures is in the following format: `signature*type*pmkid*ap-mac*client-mac*networkname***`, where:

- `signature` is, for example, `wpa`.
- `Type` can be either 01 or 02, with the majority being of type 01 and a smaller percentage, approximately 5-10%, being of type 02.
- No captures of type 03 were observed.

The hashes were initially stored on my laptop and later transferred to my desktop, which is accessible only to me. Upon completion of this thesis, all gathered hashes and cracked passwords were securely deleted.

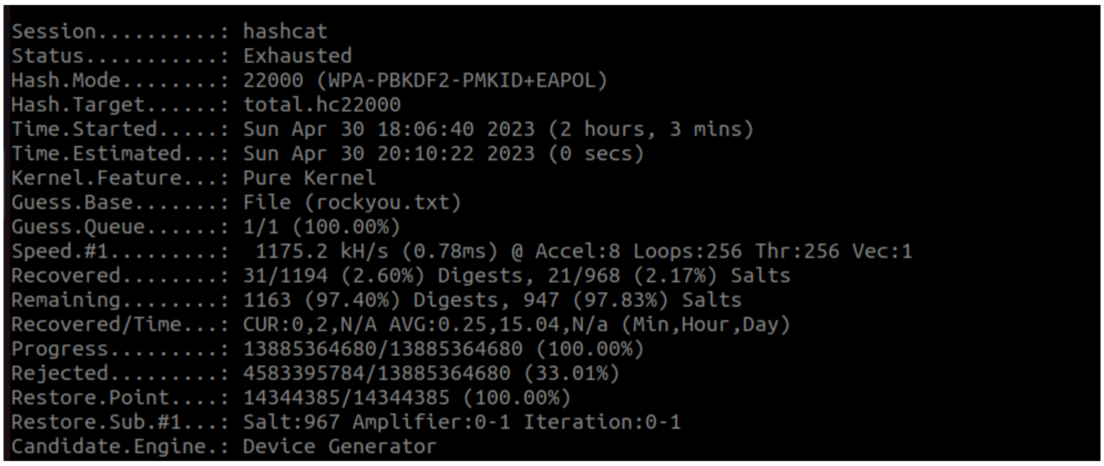
7.3. Cracking the Hashes

Cracking of the hashes was not done with the laptop used to gather them. That would have been too slow, and taken probably weeks of continuous runtime or even months. For cracking I used my desktop computer which had a decent graphics card geforce RTX 4070 ti 12G, which is the main engine in the cracking process. Cpu -i7 12700 makes only a little difference in the cracking speed. First, was tested how many of the hashes would be cracked just by brute-forcing the passwords using only digits. For wifi passwords, the hash mode 22000 was used, which is for WPA-PBKDF2-PMKID+EAPOL. Hashcat did not accept less than 8 digits, so that was the starting point. That is 100 million combinations against 1194 hashes and for the system at my disposal, this would take 22 hours of cracking. The result was 36/1194 passwords cracked. For the digits, hashcat's mask attack function was used so that you don't have to bring in the wordlist for the digits (e.g., 040?d?d?d?d?d?d?d). We know the first three digits 040 so we give hashcat it. The ?d is for digits 0-9 and hashcat will try all the combinations for them.

- Dates in form DDMMYYYY, three passwords
- Popular patterns like 12345678, three passwords
- Rest were seemingly random numbers, some had appearance of partial phone numbers or patterns.

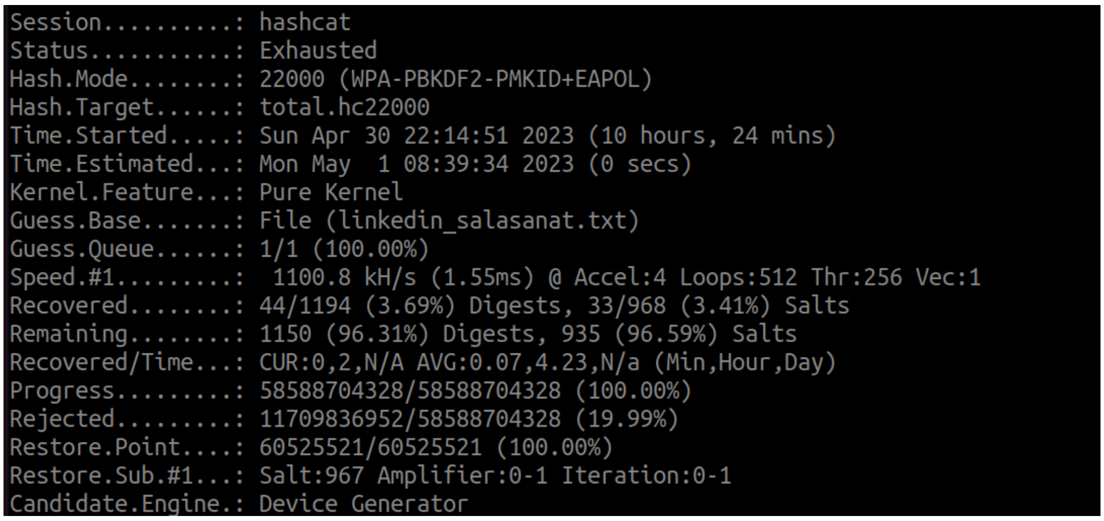
Unfortunately, the computing power was not sufficient for cracking 9 digits against WPA-PBKDF2-PMKID+EAPOL hashes. It would have taken around 9 days of continuous running, so that part had to be skipped. Next, was tested if the hashes contained phone numbers. Phone numbers are 10 digits long so brute forcing them was out of the question. Luckily Finnish mobile operator starting numbers are public knowledge. You can find them e.g from Traficom [29] . So that would be for example brute forcing 040 + 7 digits, which only took a little over 2 hours for phone numbers starting with 040. The rest of the phone number entries too were tested and the result was that only 2 phone numbers were used as passwords for these hashes. Using phone numbers as passwords does not sound to be very popular in Finland, at least in the city of Oulu. Of course, there could be phone numbers with international form +35840, which is the same as 040 or also with 040-.
Next step was to test how effective the Rockyou and LinkedIn password lists are against these hashes. Rockyou list had successful hits 31/1194, see figure 4. Patterns and dates were the most successful out of these 31 discovered passwords.

The LinkedIn list had successful hits 44/1194, see figure 5. Again, the patterns and dates seem to be the most successful. There are also a few names among the passwords.



```
Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 22000 (WPA-PBKDF2-PMKID+EAPOL)
Hash.Target......: total.hc22000
Time.Started.....: Sun Apr 30 18:06:40 2023 (2 hours, 3 mins)
Time.Estimated...: Sun Apr 30 20:10:22 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1175.2 kH/s (0.78ms) @ Accel:8 Loops:256 Thr:256 Vec:1
Recovered........: 31/1194 (2.60%) Digests, 21/968 (2.17%) Salts
Remaining........: 1163 (97.40%) Digests, 947 (97.83%) Salts
Recovered/Time...: CUR:0,2,N/A AVG:0.25,15.04,N/a (Min,Hour,Day)
Progress.........: 13885364680/13885364680 (100.00%)
Rejected.........: 4583395784/13885364680 (33.01%)
Restore.Point....: 14344385/14344385 (100.00%)
Restore.Sub.#1...: Salt:967 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
```

Figure 4. Cracking with rockyou wordlist



```
Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 22000 (WPA-PBKDF2-PMKID+EAPOL)
Hash.Target......: total.hc22000
Time.Started.....: Sun Apr 30 22:14:51 2023 (10 hours, 24 mins)
Time.Estimated...: Mon May  1 08:39:34 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (linkedin_salasanat.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1100.8 kH/s (1.55ms) @ Accel:4 Loops:512 Thr:256 Vec:1
Recovered........: 44/1194 (3.69%) Digests, 33/968 (3.41%) Salts
Remaining........: 1150 (96.31%) Digests, 935 (96.59%) Salts
Recovered/Time...: CUR:0,2,N/A AVG:0.07,4.23,N/a (Min,Hour,Day)
Progress.........: 58588704328/58588704328 (100.00%)
Rejected.........: 11709836952/58588704328 (19.99%)
Restore.Point....: 60525521/60525521 (100.00%)
Restore.Sub.#1...: Salt:967 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
```

Figure 5. Cracking with linkedIn wordlist

With the tool, a few more new hits were gained that were not already in Linkedin, rockyou or the all digits lists. Most successful out of these new hits seemed to be the name+digits combination. There were also, hits with animal names, place names and profanity. Most of the found passwords had been mangled somehow, not being completely lowercase. In total 73 distinct passwords were found. Some of the passwords seem to be in use on more than one device. That makes around 6% of devices, that are using too weak passwords.

8. CONCLUSION AND RECOMMENDATIONS

The aim of this thesis was to study common password creation habits using existing research and statistics over leaked password lists. The thesis also included a lightweight tool creation using the previous findings for creating passwords. It also investigated what are the existing tools for this job and if there is something that these tools are lacking or something that can be improved.
The prior tools that could be found were Crunch and Cewl. Both were very easy to use and particularly Cewl seemed to be useful tool for penetration testing. It just basically scraped a wordlist out of a website. Useful tool if doing a penetration test for a website or to an organization that has a website. With Crunch you can create wordlist with the characters that you choose. It will create all the combinations out of these characters. Unfortunately, the word lists this tool creates are too large to be useful. The Cewl had useful scraping feature, which should be leveraged with the tool that is going to be built. You should be able to use Cewl to scrape a site and then bring the Cewl-created list in for modifications and or mangling.
The tool tested in the thesis was created using Golang and is publicly available on GitHub [5]. The project was quite fun for learning a new programming language, as I had no prior experience of golang. The golang pointers were particularly helpful when processing very large wordlists. If the tool would have been created with python, which was the second choice, it would have needed more planning and work to get it running decently. For the tool itself, you could probably throw a year more work in, and it probably still could be improved. There are always ways to make the code cleaner, run faster and smoother or have more features.
The tool testing was done by gathering wifi hashes using hcxdumptool. The idea came from a study [17] performed by an Israeli security researcher. The Israeli researcher managed to crack 70% of wifi password hashes (5000) that he gathered passively using the hcxdumptool. The majority of cracked passwords are just digits or phone numbers, as this was, at least before the study very popular in Israel. Large number of passwords were cracked also using the rockyou password list.
In this thesis, it is worth noting that only a limited portion of such passwords was successfully cracked, accounting for approximately 6% of the collected hashes. Phone numbers were used in Oulu wifi passwords, but not nearly as frequently. Some hits from dates, names, words of animals, places, patterns, and miscellaneous words. Also, the rockyou password list had some hits but again not nearly as effective as it was in Israel. Linkedin password list and the password lists created with the tool, were also used against the hashes. Based on this small sample of gathered hashes, the passwords that Finnish people use are not terribly weak in general. The tool provided a few extra hits, which were not in the LinkedIn or rockyou list and of course most of the same passwords that were found from the leaked lists, as that is only a matter of adding the words in the base files.

8.1. Recommendations

To enhance network security, it is advisable to employ strong passwords. Avoid entirely numeric passwords since they become vulnerable to brute force attacks.

Incorporating uppercase characters only as the initial letter is insufficient, as attackers can easily exploit this. Users should also steer clear of common patterns like "12345678" or "qwerty" in passwords. It is preferable to include special characters and numbers, excluding birth year. Placing numbers at the end is a common practice, making it predictable for attackers. To enhance password strength, introduce unpredictability by using unconventional arrangements. For instance, passwords such as "Michael1995!" and "Michael1234567890" lack robustness, as they adhere to prevalent patterns in password creation.
It does not make a difference if a router is vulnerable to this attack or not. Attackers could just grab the pmkid the old-fashioned way and kick users out of the network and use the 4-way handshake. The only way to protect users is to use complex and long enough passwords so that it does not matter if they grab the hash. A complex password, so that it is not guessable but needs to be brute forced and long enough so that it cannot be cracked in reasonable time. So what makes a long enough password, lets assume that the complexity is good. A 10-character wifi password that contains lower, upper, special and numbers would take at the rate of 1200k hashes per second, 37 years and 2 months to crack. Taking into account that the attackers could (and would) have a bigger rig and each new generation of GPU is faster than the previous. 100 GPU rig would take 135 days of continuous running to crack this password, assuming each GPU has the power to crack 1200k pmkid hashes per second. That does not sound uncrackable anymore. By adding 2 more characters so that the length is 12 characters the time suddenly jumps from 37 years to 35484 years and 10 months. GPUs need some huge performance leaps in order to brute force 12-character passwords. If users also change the password regularly, e.g once a year, by the time they have cracked it, it would be useless for them and they would have to start all over again. However, even 12-character passwords might not be enough in future when the quantum computers come into the equation.

BIBLIOGRAPHY

[1] Yurong Zhao. Wireless network security status in oulu: war-driving. Master's thesis, Y. Zhao, 2019.

[2] Joakim Kävrestad, Fredrik Eriksson, and Marcus Nohlberg. Understanding passwords–a taxonomy of password creation strategies. *Information & Computer Security*, 27(3):453–467, 2019.

[3] Hazel Murray and David Malone. Choosing wordlists for password guessing: An adaptive multi-armed bandit approach. In *International Symposium on Foundations and Practice of Security*, pages 393–413. Springer, 2021.

[4] Mashael AlSabah, Gabriele Oligeri, and Ryan Riley. Your culture is in your password: An analysis of a demographically-diverse password dataset. *Computers & security*, 77:427–441, 2018.

[5] wordlistgenerator, . URL `https://github.com/jannepetter/wordlistgenerator`.

[6] Keskusrikospoliisi sai valmiiksi psykoterapiakeskus vastaamon jättimäisen tietomurron ja kiristysvyyhdin tutkinnan – epäilty kiistää yhä teot, . URL `https://yle.fi/a/74-20047643`.

[7] Hackers sneak code onto oulu city website to mine cryptocurrency, . URL `https://yle.fi/a/3-12426254`.

[8] Suomalaisten selväkielisiä salasanoja paljastunut, . URL `https://www.kyberturvallisuuskeskus.fi/fi/suomalaisten-selvakielisia-salasanoja-paljastunut`.

[9] Ukraine,us intelligence suggest russia cyber efforts evolving, growing, . URL `https://www.voanews.com/a/ukraine-us-intelligence-suggest-russia-cyber-efforts-evolving-growing-/7259396.html`.

[10] More russian cyber-attacks targeting finland, agencies say, . URL `https://yle.fi/a/74-20028302`.

[11] North korea hacked finnish defence industries, . URL `https://yle.fi/a/74-20050314`.

[12] Kansanedustaja käyttää eduskunnassa metron wifiä, . URL `https://www.verkkouutiset.fi/a/kansanedustaja-anna-kontula-kayttaa-toissa-metron-wifia-asiantuntija-ei-suosittele/#8d78455a`.

[13] Saku Lindroos, Antti Hakkala, and Seppo Virtanen. A systematic methodology for continuous wlan abundance and security analysis. *Computer Networks*, 197: 108359, 2021.

[14] Saku Lindroos, Antti Hakkala, and Seppo Virtanen. The covid-19 pandemic and remote working did not improve wlan security. *Procedia Computer Science*, 201: 158–165, 2022.

[15] Steven Furnell and Rawan Esmael. Evaluating the effect of guidance and feedback upon password compliance. *Computer Fraud & Security*, 2017(1):5–10, 2017.

[16] Tejaswi Kakarla, Aakif Mairaj, and Ahmad Y Javaid. A real-world password cracking demonstration using open source tools for instructional use. In *2018 IEEE International Conference on Electro/Information Technology (EIT)*, pages 0387–0391. IEEE, 2018.

[17] Cracking wifi at scale, . URL `https://www.cyberark.com/resources/threat-research-blog/cracking-wifi-at-scale-with-one-simple-trick`.

[18] Ubuntu-manpages-crunch, . URL `https://manpages.ubuntu.com/manpages/bionic/man1/crunch.1.html`.

[19] Cewl, . URL `https://digi.ninja/projects/cewl.php`.

[20] Hashcat, . URL `https://hashcat.net/wiki/doku.php?id=rule_based_attack`.

[21] Eleni Veroni, Christoforos Ntantogian, and Christos Xenakis. A large-scale analysis of wi-fi passwords. *Journal of Information Security and Applications*, 67:103190, 2022.

[22] Keika Mori, Takuya Watanabe, Yunao Zhou, Ayako Akiyama Hasegawa, Mitsuaki Akiyama, and Tatsuya Mori. Comparative analysis of three language spheres: Are linguistic and cultural differences reflected in password selection habits? *IEICE TRANSACTIONS on Information and Systems*, 103(7):1541–1555, 2020.

[23] Weili Han, Zhigong Li, Lang Yuan, and Wenyuan Xu. Regional patterns and vulnerability analysis of chinese web passwords. *IEEE Transactions on Information Forensics and Security*, 11(2):258–272, 2015.

[24] weakest-passwords-2022, . URL `https://cybernews.com/security/weakest-passwords-2022/`.

[25] Kunyu Yang, Xuexian Hu, Qihui Zhang, Jianghong Wei, and Wenfen Liu. Studies of keyboard patterns in passwords: Recognition, characteristics and strength evolution. In *Information and Communications Security: 23rd International Conference, ICICS 2021, Chongqing, China, November 19-21, 2021, Proceedings, Part I 23*, pages 153–168. Springer, 2021.

[26] Shunbin Li, Zhiyu Wang, Ruyun Zhang, Chunming Wu, and Hanguang Luo. Mangling rules generation with density-based clustering for password guessing. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[27] Väestötietojärjestelmän suomalaisten nimiaineistot, . URL `https://www.avoindata.fi/data/fi/dataset/none`.

[28] 4-way-handshake, . URL `https://www.wifi-professionals.com/2019/01/4-way-handshake`.

[29] Traficom, . URL `https://www.traficom.fi/fi/viestinta/laajakaista-ja-puhelin/matkaviestinverkkojen-suuntanumerot`.