Lab 1: Password Cracking

1. Methodology

The program used to crack the passwords was *Hashcat*, as it uses by default the graphics card instead of the processor. *John the Ripper* was also tested with the default options, and one password was cracked thanks to it, though it was later cracked with *Hashcat* as well.

The specific commands used to crack the different passwords will be included in the second section of the document. In this part I will specify the different tests, dictionaries, sets of rules created and used, as well as other tools employed for the resolution of the lab.

a. Types of attacks used.

- Straight (`-a0`): this attack mode was used to perform the dictionary-based attacks (including with rules).
- Combination (`-a1`): it was used as an alternative to the modes 6 and 7 (dictionary + mask and vice versa), to prepend/append a up to 4 alphanumeric symbols to a wordlist. This attack could have been done with the mentioned attack modes, but both demonstrated to be slower as the Estimated Time of Arrival for both was of 100 days to try the almost 450 million possibilities (at a rate of around 30 H/s). These attack modes were far slower since they could not make use of the Slow-candidates option. In comparison to them, with attack mode 1 the ETA to try all the combinations had a value of little more than 3 hours (at a rate of 23.000 H/s).
- Brute-Force (`-a3`): this mode was used to perform a brute force attack of a length of up to 4 characters in incremental mode with a mask selecting all alphanumeric symbols plus special characters.

b. Wordlists tried.

- Rockyou: famous wordlist with over 140 million passwords that was able to recover one of the passwords.
- Cain & Abel: another famous wordlist with over 306.000 passwords.
- JtR wordlist: dictionary with 3 thousand passwords; included in John the Ripper.
- NIAs & NIAs_noprefix: two dictionaries I wrote with a script I made in python to generate all the possible NIAs of the university, though they gave no good result.
- Wordlists generated with `cewl`: 4 different dictionaries were created with this tool. The wordlists generated were from the Computer Science main page, the Cybersecurity Engineering course main page, and the home page of the UC3M, all of them in Spanish and English.
- Suffixes.txt: self-made wordlist with some words related to the university, such as “uc3m” and variations, “alumnos”, etc.
- Users.txt: a wordlist with the usernames of the users in the *passwd* file.

c. Rules.

- `Best64.rule`: a ruleset included in hashcat which includes the 64 most effective rules when cracking passwords, such as number appends, rotations, reversals, overwrites, etc.
- `d3ad0ne.rule` & `T0XlC.rule`: other rulesets included in hashcat.

- Rulesets made with the tool `maskprocessor` (a tool to automate the creation of rules or words by putting in the charsets you want) to create rules to append and prepend different characters.

d. Other options used in hashcat.

- `-S`: usage of slower but advanced candidates. Increased the performance a lot in several cases, like the dictionary attacks.
- `-O`: enable optimized kernel mode to limit password length.
- `-w [1-4]`: selection of the workload profile. Ranging from 1 for the lightest impact in the system to 4 for the more exhaustive mode.

2. Results

I have been able to crack 7 out of the 10 passwords thanks to the different attacks I have tried.

User:password	`usuario_5_4:uc3mCoj8`
Time	Around 4 minutes.
Command	`hashcat -a1 -w4 -m1800 -O -S lab1/passwords.hash dictionaries/suffixes.txt maskprocessor-0.73/alphanum.txt -o lab1/cracked.txt`
User:password	`usuario_5_5: uc3mCos4.`
Time	Around 4 minutes
Command	`hashcat -a1 -w4 -m1800 -O -S lab1/passwords.hash dictionaries/suffixes.txt maskprocessor-0.73/alphanum.txt -o lab1/cracked.txt`
User:password	`usuario_5_6: uc3mEaj9`
Time	Around 4 minutes.
Command	`hashcat -a1 -w4 -m1800 -O -S lab1/passwords.hash dictionaries/suffixes.txt maskprocessor-0.73/alphanum.txt -o lab1/cracked.txt`
User:password	`usuario_5_7:usuario_5_7`
Time	6 seconds.
Command	`hashcat -a0 -w3 -m1800 lab1/passwords.hash -r rules/d3ad0ne.rule lab1/100405846@alumnos.uc3m.es_passwd -O -S -o lab1/cracked.txt`
User:password	`usuario_5_8:$HEX[3a](“”:””)`
Time	Less than 1 minute, around 40 seconds.
Command	`hashcat -a 3 -m 1800 -w 3 -O -S lab1/passwords.hash --increment --increment-min 1 --increment-max 4 "?a?a?a?a" -o lab1/cracked.txt`
User:password	`usuario_5_9:prove`
Time	Less than 10 seconds.
Command	`hashcat -a 0 -w 3 -m 1800 -O lab1/passwords.hash uc3m_computer_science_main_page.txt -o lab1/cracked.txt`
User:password	`usuario_5_10:status`
Time	Less than 3 minutes.
Command	`hashcat -a 0 -m 1800 -w 3 -O lab1/passwords.hash dictionaries/rockyou.txt`