

REPORT LAB 2

- Enrico Ferraiolo 0001020254 enrico.ferraiolo2@studio.unibo.it
- Simone Folli 0000974629 simone.folli2@studio.unibo.it
- Gianlorenzo Urbano 0001020458 gianlorenzo.urbano@studio.unibo.it

Exercise 1

Hash n.1: 6e6bc4e49dd477ebc98ef4046c067b5f

We executed `sudo rtgen md5 loweralpha 1 7 0 1000 100000 0`

We then sorted with `sudo rtsort /usr/share/rainbowcrack`

The plaintext found is `ciao`.

```
(kali@kali)-[~]
$ rcrack /usr/share/rainbowcrack -h 6e6bc4e49dd477ebc98ef4046c067b5f
2 rainbow tables found
memory available: 2544507289 bytes
memory for rainbow chain traverse: 16000 bytes per hash, 16000 bytes for 1 hashes
memory for rainbow table buffer: 2 x 14400016 bytes
disk: /usr/share/rainbowcrack/md5_loweralpha#1-7_0_1000x900000_0.rt: 14400000 bytes read
disk: /usr/share/rainbowcrack/md5_loweralpha#1-7_0_1000x100000_0.rt: 1600000 bytes read
disk: finished reading all files
plaintext of 6e6bc4e49dd477ebc98ef4046c067b5f is ciao

statistics
-----
plaintext found:          1 of 1
total time:              0.16 s
time of chain traverse:   0.15 s
time of alarm check:     0.00 s
time of disk read:       0.01 s
hash & reduce calculation of chain traverse: 499000
hash & reduce calculation of alarm check:    15262
number of alarm:         48
performance of chain traverse: 3.28 million/s
performance of alarm check:  3.82 million/s

result
-----
6e6bc4e49dd477ebc98ef4046c067b5f  ciao  hex:6369616f
```

Hash n.2: 427ade9c15ec643751860eba9899355b

We increased the chains from 100000 to 900000: `sudo rtgen md5 loweralpha 1 7 0 1000 900000 0`

We sorted with `sudo rtsort /usr/share/rainbowcrack`

We then executed `rcrack /usr/share/rainbowcrack -h 427ade9c15ec643751860eba9899355b`

The plaintext found is `gatto`.

```
(kali@kali)~$ rcrack /usr/share/rainbowcrack -h 427ade9c15ec643751860eba9899355b
2 rainbow tables found
memory available: 2551146086 bytes
memory for rainbow chain traverse: 16000 bytes per hash, 16000 bytes for 1 hashes
memory for rainbow table buffer: 2 x 14400016 bytes
disk: /usr/share/rainbowcrack/md5_loweralpha#1-7_0_1000x900000_0.rt: 14400000 bytes read
disk: /usr/share/rainbowcrack/md5_loweralpha#1-7_0_1000x100000_0.rt: 1600000 bytes read
disk: finished reading all files
plaintext of 427ade9c15ec643751860eba9899355b is gatto

statistics
-----
plaintext found:                1 of 1
total time:                    0.16 s
time of chain traverse:         0.15 s
time of alarm check:           0.01 s
time of disk read:             0.01 s
hash & reduce calculation of chain traverse: 499000
hash & reduce calculation of alarm check:    21212
number of alarm:                63
performance of chain traverse:   3.28 million/s
performance of alarm check:     3.03 million/s

result
-----
427ade9c15ec643751860eba9899355b  gatto  hex:676174746f
```

Exercise 2

Hash n.1:

6c00f2d6e1610bfc9b415daf80d45855f2c56443c2dc2f71e7ef27168d1f2857d6168f4d374ed8ec
a349f2debd18d4ccac339218ca70446adf999060395742b4 Salt: hjt88q

We executed the command `hashcat -a 0 -m 1710`

"6c00f2d6e1610bfc9b415daf80d45855f2c56443c2dc2f71e7ef27168d1f2857d6168f4d374ed8e
ca349f2debd18d4ccac339218ca70446adf999060395742b4:hjt88q"
/usr/share/wordlists/rockyou.txt

The plaintext found is `markinho`

```

OpenCL API (OpenCL 3.0 PoCL 5.0+debian Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEF,
DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]

-----
* Device #1: cpu-sandybridge-AMD Ryzen 7 4700U with Radeon Graphics, 1439/2943 MB (512 MB allo
catable), 2MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
Minimum salt length supported by kernel: 0
Maximum salt length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash
* Uses-64-Bit

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

6c00f2d6e1610bfc9b415daf80d45855f2c56443c2dc2f71e7ef27168d1f2857d6168f4d374ed8eca349f2debd18d4ccac339218ca70446adf999060395742b4:hjt88q:markinho

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 1710 (sha512($pass.$salt))
Hash.Target.....: 6c00f2d6e1610bfc9b415daf80d45855f2c56443c2dc2f71e7e ... hjt88q
Time.Started.....: Tue Apr 16 17:19:08 2024 (0 secs)
Time.Estimated...: Tue Apr 16 17:19:08 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)

```

Exercise 3

Hash n.1: **0e8ae09ae169926a26b031c18c01bafa**

HINT: It contains a phrase without spaces and some numbers at the end

We used **hashid** to uncover what type of hash we are dealing with

We then used **hashcat -a 0 -m 0 -r /usr/share/hashcat/rules/T0X1C.rule "0e8ae09ae169926a26b031c18c01bafa" /usr/share/wordlists/rockyou.txt**

The plaintext found is **ILOVEME8320**.

```

0e8ae09ae169926a26b031c18c01bafa:ILOVEME8320

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: 0e8ae09ae169926a26b031c18c01bafa
Time.Started.....: Fri Apr 19 17:14:00 2024 (17 mins, 7 secs)
Time.Estimated...: Fri Apr 19 17:31:07 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Mod.....: Rules (/usr/share/hashcat/rules/T0XlC.rule)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 8308.0 kH/s (3.51ms) @ Accel:128 Loops:128 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 8716314112/58596812725 (14.88%)
Rejected.....: 0/8716314112 (0.00%)
Restore.Point....: 2133504/14344385 (14.87%)
Restore.Sub.#1...: Salt:0 Amplifier:3584-3712 Iteration:0-128
Candidate.Engine.: Device Generator
Candidates.#1....: ILOVEOMAR40 → ILEEN11992^
Hardware.Mon.#1..: Util: 94%

Started: Fri Apr 19 17:13:59 2024
Stopped: Fri Apr 19 17:31:09 2024

```

Hash n.2: **c73fceaab80035a75ba3fd415ecb2735**

HINT: it contains, in order: a common word, some numbers and a special character

With the same steps as before, we found the type of the hash (MD5).

We then used `hashcat -a 0 -m 0 -r /usr/share/hashcat/rules/T0XlCv2.rule "c73fceaab80035a75ba3fd415ecb2735" /usr/share/wordlists/rockyou.txt`

The plaintext found is **soccer23!**.

```

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 286887700000

c73fceaab80035a75ba3fd415ecb2735:soccer23!

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: c73fceaab80035a75ba3fd415ecb2735
Time.Started.....: Fri Apr 19 17:35:14 2024 (1 sec)
Time.Estimated...: Fri Apr 19 17:35:15 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Mod.....: Rules (/usr/share/hashcat/rules/T0XlCv2.rule)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 5574.0 kH/s (11.13ms) @ Accel:256 Loops:128 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 1310720/286887700000 (0.00%)
Rejected.....: 0/1310720 (0.00%)
Restore.Point....: 0/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:2432-2560 Iteration:0-128
Candidate.Engine.: Device Generator
Candidates.#1....: 1234562046{ → letmein25
Hardware.Mon.#1..: Util: 54%

Started: Fri Apr 19 17:35:13 2024
Stopped: Fri Apr 19 17:35:16 2024

```

Hash n.3: **dc612dc12fb4540a88b88875c2bee3b4**

HINT: it contains, in order: a common word and one or two numbers. The common word has the case INVERTED.

To crack this hash we generate a new rule file in the rules directory. (`nano /usr/share/hashcat/rules/myrule1.rule`)

The new rules are the following:

```
C $0 $0
C $0 $1
C $0 $2
C $0 $3
C $0 $4
C $0 $5
C $0 $6
C $0 $7
C $0 $8
C $0 $9
C $1 $0
C $1 $1
C $1 $2
C $1 $3
C $1 $4
C $1 $5
C $1 $6
C $1 $7
C $1 $8
C $1 $9
C $2 $0
C $2 $1
C $2 $2
C $2 $3
C $2 $4
.
.
.
C $9 $7
C $9 $8
C $9 $9
```

We then used `hashcat -a 0 -m 0 -r /usr/share/hashcat/rules/myrule1.rule "dc612dc12fb4540a88b88875c2bee3b4" /usr/share/wordlists/rockyou.txt`

After some minutes the plaintext found is `dANIELELGUAP016`.

```
dc612dc12fb4540a88b88875c2bee3b4:dANIELELGUAPO16

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: dc612dc12fb4540a88b88875c2bee3b4
Time.Started.....: Tue Apr 23 10:58:46 2024 (23 secs)
Time.Estimated...: Tue Apr 23 10:59:09 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Mod.....: Rules (/usr/share/hashcat/rules/myrule1.rule)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 10352.2 kH/s (2.28ms) @ Accel:128 Loops:100 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 188108800/1434438500 (13.11%)
Rejected.....: 0/188108800 (0.00%)
Restore.Point....: 1880832/14344385 (13.11%)
Restore.Sub.#1...: Salt:0 Amplifier:0-100 Iteration:0-100
Candidate.Engine.: Device Generator
Candidates.#1....: dANIELELGUAPO00 → dANIEL20299
Hardware.Mon.#1..: Util: 96%

Started: Tue Apr 23 10:58:45 2024
Stopped: Tue Apr 23 10:59:11 2024
```

Bandit Game

Level 0

We simply connected to the ssh server with the given address, user and port.

Level 0 -> 1

We used the `ls` command to list the home directory files, then `cat readme` to read the contents of the readme file.

Level 1 -> 2

We again used the `ls` command to find the files in the home directory, then `cat ./-` to read the contents of the `-` file.

Note: we couldn't just use `cat -` because the `-` char is used to pass flags to linux commands.

Level 2 -> 3

Once again, the `ls` command. Then `cat ./spaces\ in\ this\ filename`.

Note: the whitespace character must be escaped with `.`

Level 3 -> 4

We changed directory with `cd inheres` command, then we listed ALL of the directory contents (included the hidden files) with `ls -la`. When then got the password from the `.hidden` file with `cat .hidden`.

Level 4 -> 5

We changed dir to `inhere` with `cd inhere`, then we listed all the directory contents with `ls`.

To print all the contents of the files, we used `cat ./-file0*`.

Note: `cat *` couldn't be used because all of the files begin with `-`.

Level 5 -> 6

We use a pretty convoluted command to find the password in this stage, to avoid having to "cd ls cd .. cat" everything...

We first use the `man du` to check if there are any options to see both all of the files in the directory tree and to list its size in bytes (`du -ba`).

We then pipe the output into an `awk` command to check if the first element printed (the bites in size) is equal to 1033. Then we printed the file contents with the `cat` command.

```
du -ab | awk '{if ($1 == 1033) print}'
```

Level 6 -> 7

We `cd ../..` to go in the server base dir, then `find -group bandit6 -user bandit7` to list all the files owned by user bandit7 and group bandit6, finding only one file.

Level 7 -> 8

We use the `cat data.txt` to list the contents of the file, then we pipe the output to an `awk` function that prints the line only if the first element is "millionth".

```
cat data.txt | awk '{ if ($1 == "millionth") print}'
```

Level 8 -> 9

We pipe the output of the `cat` command to `sort`, then to `uniq -c` (to also print the count). The password is the only line that appears once.

```
cat data.txt | sort | uniq -c
```

Level 9 -> 10

We use the `strings data.txt` to list all the lines that contain human readable chars, then we pipe the output to `grep "==="` to only print lines that have at least 2 equal signs.

```
strings data.txt | grep "==="
```

(the pass for the next level is `G7w8LIi6J3kTb8A7j9LgrywtEULyyp6s` but i dont know if we have to do it)