LAB ASSIGNMENT- 07

CYBERSECURITY

RITIK TIWARI

B21CS098

Aim: Understanding the use of password brute forcing tools.

In this lab, you have to find at least five password brute forcing tools and use them on your system. Make a report on comparison between them based on various factors such as type of password able to track/guess (numeric, alphanumeric, upper case, lower case, special character,length), platform (windows,linux, mac), time taken, etc. Include the screenshots of your experiments in the report.

Some examples of brute force tools are:

1. Aircrack-ng

2. John the Ripper

3. L0phtCrack

4. Hashcat

5. DaveGrohl

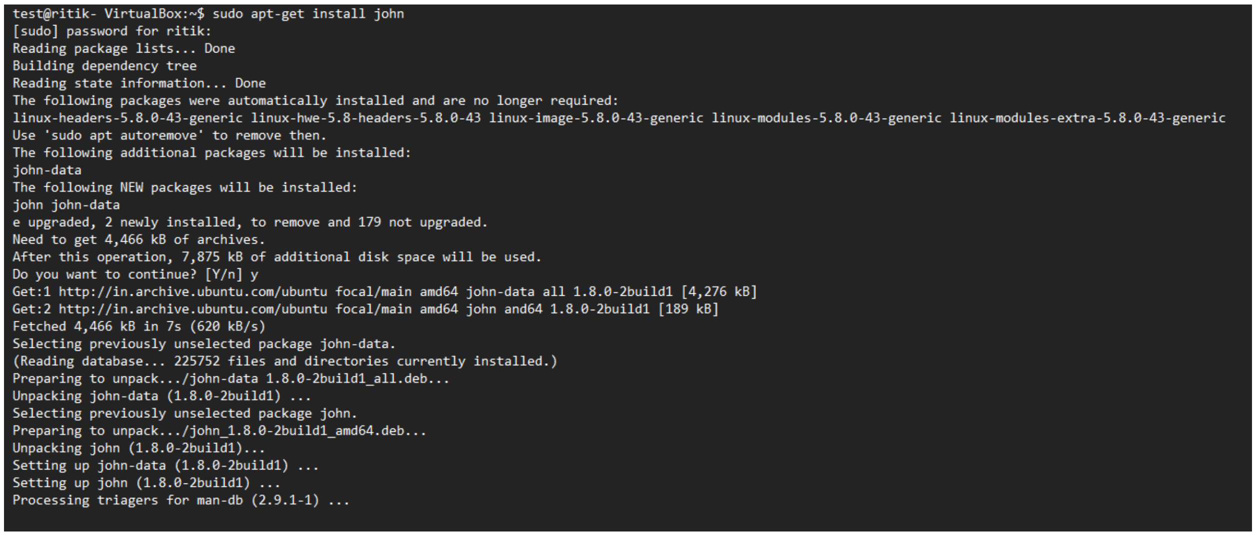
6. Ncrack

7. ...

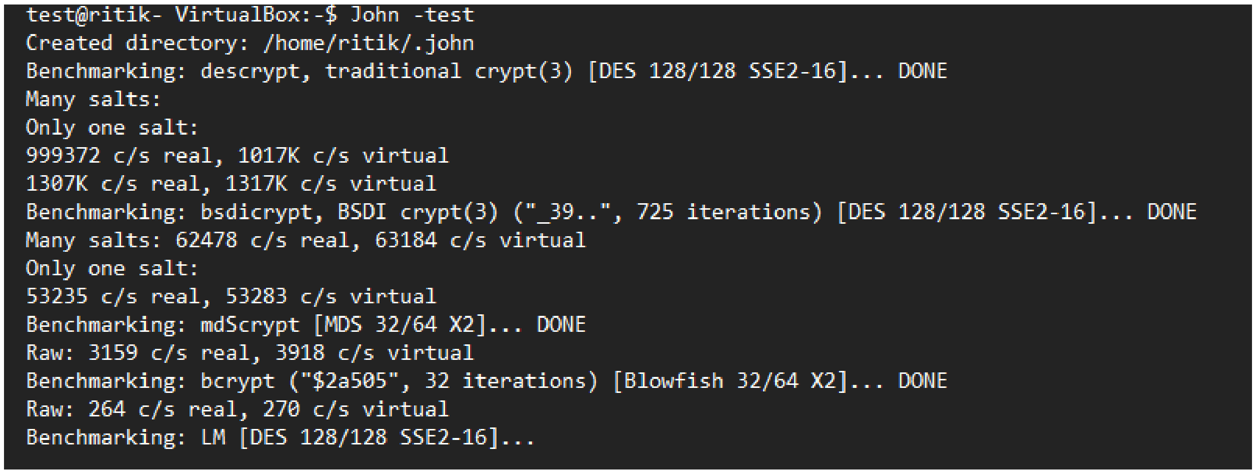
Some of the tools that are used and observed by me are:

a) **John the Ripper:**

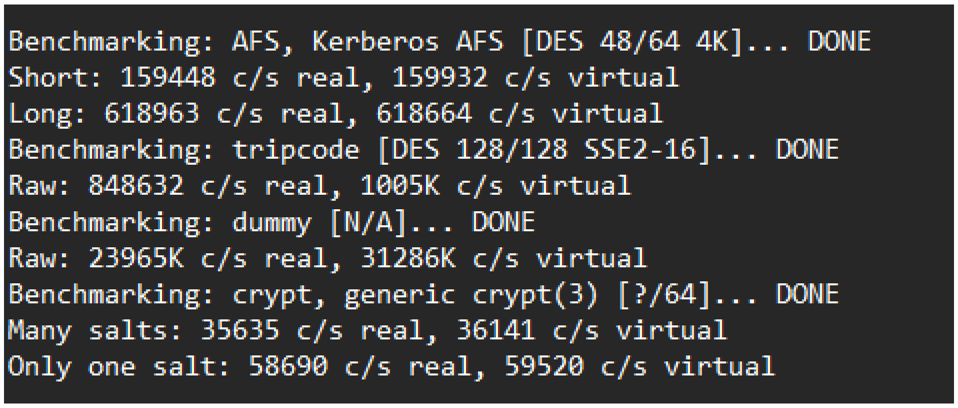
This programme can monitor passwords encrypted with DES, MD5, SHA-1, SHA-256, and other algorithms and perform dictionary, incremental, and brute force attacks on them. It supports Microsoft Windows, Linux, and Mac OS X.



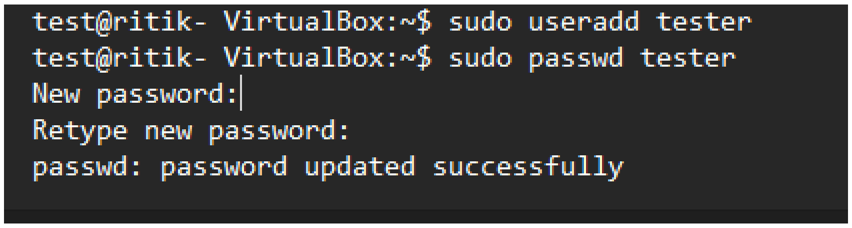
```
test@ritik- VirtualBox:~$ sudo apt-get install john
[sudo] password for ritik:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
linux-headers-5.8.0-43-generic linux-hwe-5.8-headers-5.8.0-43 linux-image-5.8.0-43-generic linux-modules-5.8.0-43-generic linux-modules-extra-5.8.0-43-generic
Use 'sudo apt autoremove' to remove then.
The following additional packages will be installed:
john-data
The following NEW packages will be installed:
john john-data
e upgraded, 2 newly installed, to remove and 179 not upgraded.
Need to get 4,466 kB of archives.
After this operation, 7,875 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal/main amd64 john-data all 1.8.0-2build1 [4,276 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/main amd64 john and64 1.8.0-2build1 [189 kB]
Fetched 4,466 kB in 7s (620 kB/s)
Selecting previously unselected package john-data.
(Reading database... 225752 files and directories currently installed.)
Preparing to unpack.../john-data 1.8.0-2build1_all.deb...
Unpacking john-data (1.8.0-2build1) ...
Selecting previously unselected package john.
Preparing to unpack.../john_1.8.0-2build1_amd64.deb...
Unpacking john (1.8.0-2build1)...
Setting up john-data (1.8.0-2build1) ...
Setting up john (1.8.0-2build1) ...
Processing triagers for man-db (2.9.1-1) ...
```



```
test@ritik- VirtualBox:-$ John -test
Created directory: /home/ritik/.john
Benchmarking: descrypt, traditional crypt(3) [DES 128/128 SSE2-16]... DONE
Many salts:
Only one salt:
999372 c/s real, 1017K c/s virtual
1307K c/s real, 1317K c/s virtual
Benchmarking: bsdicrypt, BSDI crypt(3) ("_39..", 725 iterations) [DES 128/128 SSE2-16]... DONE
Many salts: 62478 c/s real, 63184 c/s virtual
Only one salt:
53235 c/s real, 53283 c/s virtual
Benchmarking: mdScrypt [MDS 32/64 X2]... DONE
Raw: 3159 c/s real, 3918 c/s virtual
Benchmarking: bcrypt ("$2a505", 32 iterations) [Blowfish 32/64 X2]... DONE
Raw: 264 c/s real, 270 c/s virtual
Benchmarking: LM [DES 128/128 SSE2-16]...
```

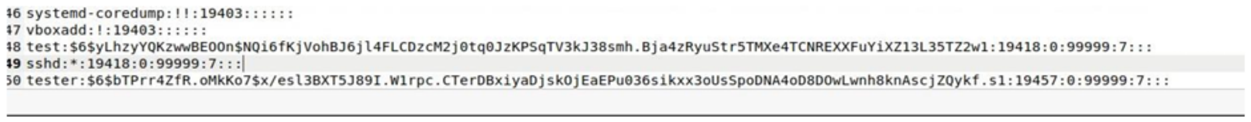


```
Benchmarking: AFS, Kerberos AFS [DES 48/64 4K]... DONE
Short: 159448 c/s real, 159932 c/s virtual
Long: 618963 c/s real, 618664 c/s virtual
Benchmarking: tripcode [DES 128/128 SSE2-16]... DONE
Raw: 848632 c/s real, 1005K c/s virtual
Benchmarking: dummy [N/A]... DONE
Raw: 23965K c/s real, 31286K c/s virtual
Benchmarking: crypt, generic crypt(3) [?/64]... DONE
Many salts: 35635 c/s real, 36141 c/s virtual
Only one salt: 58690 c/s real, 59520 c/s virtual
```



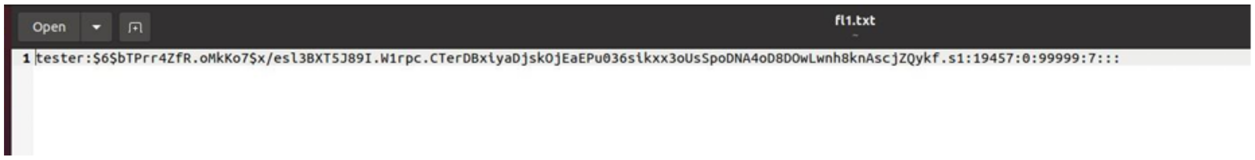
```
test@ritik- VirtualBox:~$ sudo useradd tester
test@ritik- VirtualBox:~$ sudo passwd tester
New password:
Retype new password:
passwd: password updated successfully
```

This is the strategy I used to discover the new user's password on the Linux server. I've set up a user named tester with the password (welcoma). A few details about the password in the path: /etc/shadows is provided to us.

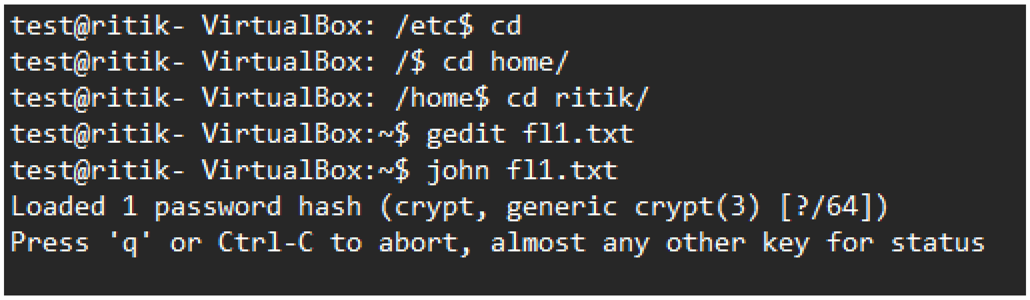


```
systemd-coredump:!!:19403::::::
vboxadd:!:19403::::::
test:$6$yLhzyYQKzwwBEOOn$NQi6fKjVohBJ6jl4FLCDzcM2j0tq0JzKPSqTV3kJ38smh.Bja4zRyuStr5TMXe4TCNREXXFuYiXZ13L35TZ2w1:19418:0:99999:7:::
sshd:*:19418:0:99999:7:::
tester:$6$bTPrr4ZfR.oMkKo7$x/esl3BXT5J89I.W1rpc.CTerDBxiyaDjskOjEaEPu036sikxx3oUsSpoDNA4oD8DOwLwnh8knAscjZQykf.s1:19457:0:99999:7:::
```

From this whole file we can extract out the information for the created user and the below figure shows that.

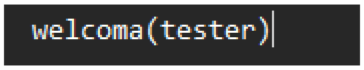


```
tester:$6$bTPrr4ZfR.oMkKo7$x/esl3BXT5J89I.W1rpc.CTerDBxiyaDjskOjEaEPu036sikxx3oUsSpoDNA4oD8DOwLwnh8knAscjZQykf.s1:19457:0:99999:7:::
```



```
test@ritik- VirtualBox: /etc$ cd
test@ritik- VirtualBox: /$ cd home/
test@ritik- VirtualBox: /home$ cd ritik/
test@ritik- VirtualBox:~$ gedit fl1.txt
test@ritik- VirtualBox:~$ john fl1.txt
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
```

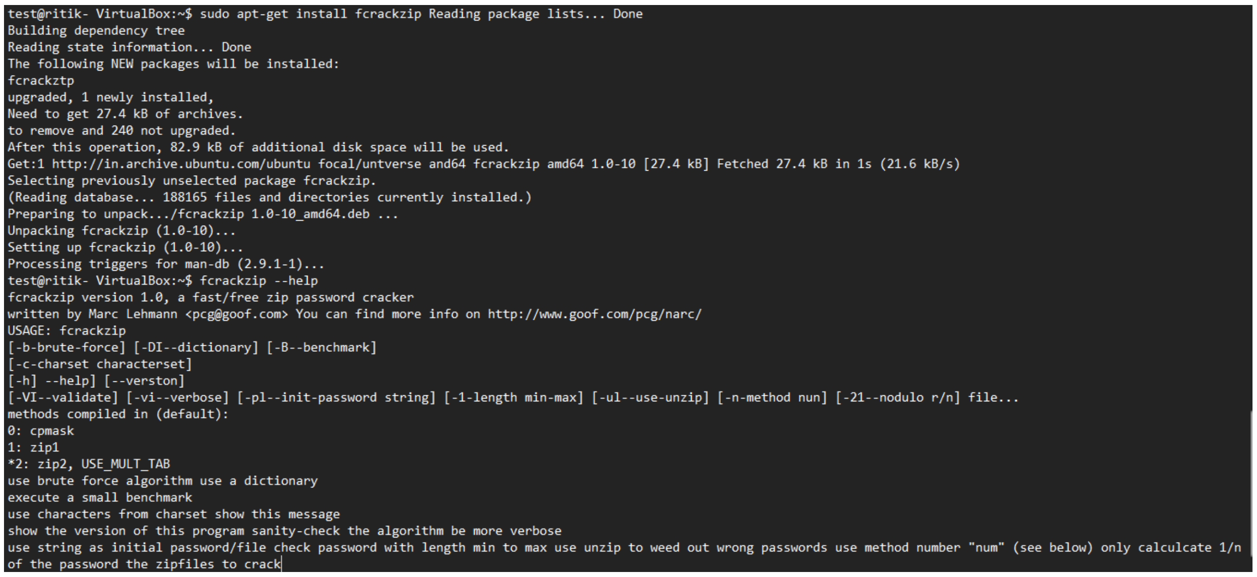
In the terminal we can invoke the john method which would give us the password by brutally guessing it.



```
welcoma(tester)
```

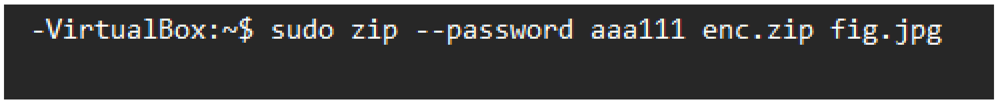
b) Fcrackzip

It is a brute force type of attack . It can guess and track the passwords of the type of ZIP archives. It is available for Windows, Linux, and macOS platforms.



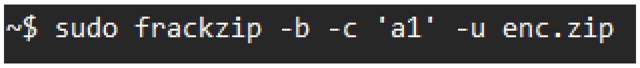
```
test@ritik- VirtualBox:~$ sudo apt-get install fcrackzip Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
fcrackztp
upgraded, 1 newly installed,
Need to get 27.4 kB of archives.
to remove and 240 not upgraded.
After this operation, 82.9 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu focal/untverse and64 fcrackzip amd64 1.0-10 [27.4 kB] Fetched 27.4 kB in 1s (21.6 kB/s)
Selecting previously unselected package fcrackzip.
(Reading database... 188165 files and directories currently installed.)
Preparing to unpack.../fcrackzip 1.0-10_amd64.deb ...
Unpacking fcrackzip (1.0-10)...
Setting up fcrackzip (1.0-10)...
Processing triggers for man-db (2.9.1-1)...
test@ritik- VirtualBox:~$ fcrackzip --help
fcrackzip version 1.0, a fast/free zip password cracker
written by Marc Lehmann <pcg@goof.com> You can find more info on http://www.goof.com/pcg/narc/
USAGE: fcrackzip
[-b-brute-force] [-DI--dictionary] [-B--benchmark]
[-c-charset characterset]
[-h] --help] [--verston]
[-VI--validate] [-vi--verbose] [-pl--init-password string] [-1-length min-max] [-ul--use-unzip] [-n-method nun] [-21--nodulo r/n] file...
methods compiled in (default):
0: cpmask
1: zip1
*2: zip2, USE_MULT_TAB
use brute force algorithm use a dictionary
execute a small benchmark
use characters from charset show this message
show the version of this program sanity-check the algorithm be more verbose
use string as initial password/file check password with length min to max use unzip to weed out wrong passwords use method number "num" (see below) only calculcate 1/n
of the password the zipfiles to crack
```

For the sake of this software's monitoring, I compressed the picture file fig.jpg into a zip archive and secured it with a password. Finally, I compressed the image fig.jpg using the zip command and the password aaa111.



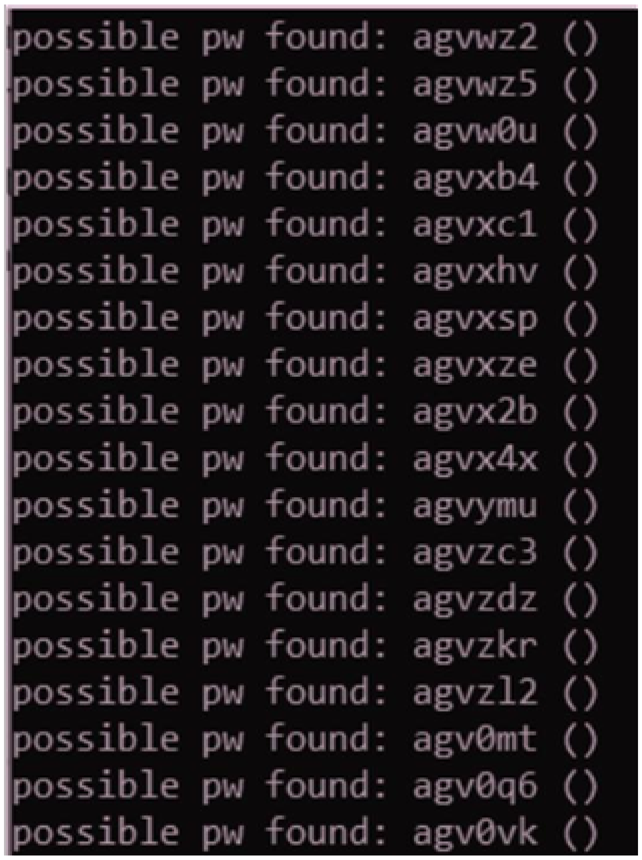
```
-VirtualBox:~$ sudo zip --password aaa111 enc.zip fig.jpg
```

The following command allows it to get the password (aaa111).



```
~$ sudo frackzip -b -c 'a1' -u enc.zip
```

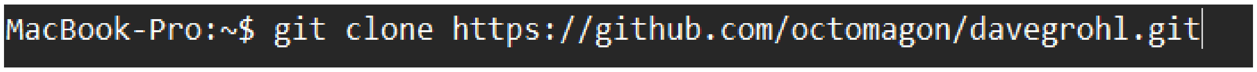
The below image shows that the method used for finding the passwords is brute force as it is looking for the characters combinations in a sequential manner.

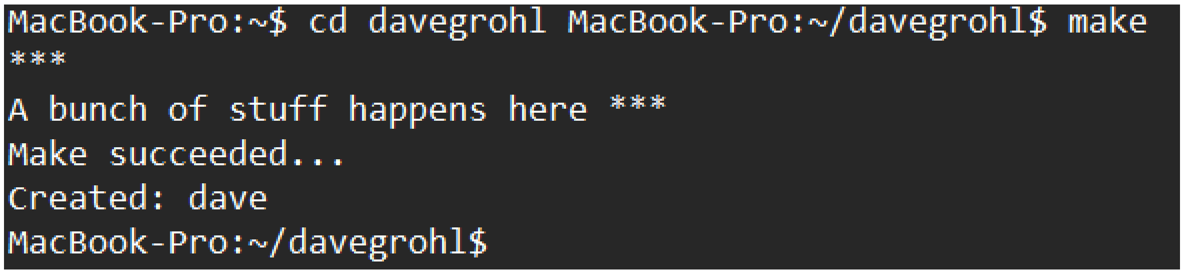

```
possible pw found: agvwz2 ()
possible pw found: agvwz5 ()
possible pw found: agvw0u ()
possible pw found: agvxb4 ()
possible pw found: agvxc1 ()
possible pw found: agvxhv ()
possible pw found: agvxsp ()
possible pw found: agvxze ()
possible pw found: agvx2b ()
possible pw found: agvx4x ()
possible pw found: agvymu ()
possible pw found: agvzc3 ()
possible pw found: agvzdz ()
possible pw found: agvzkr ()
possible pw found: agvzl2 ()
possible pw found: agv0mt ()
possible pw found: agv0q6 ()
possible pw found: agv0vk ()
```

Here, we can see that after some time, the results of trying every conceivable password combination were displayed. The output password is aaa111, which is the same as previously.

c) DaveGrohl

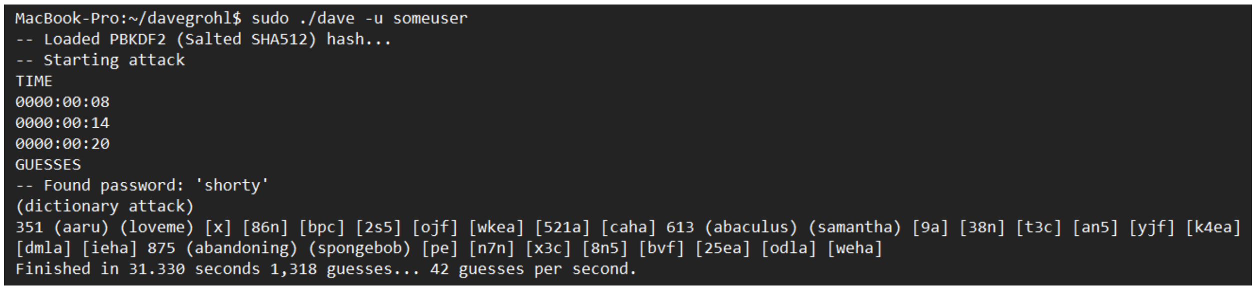
DaveGrohl is a brute-force password cracker for macOS.


```
MacBook-Pro:~$ git clone https://github.com/octomagon/davegrohl.git
```



```
MacBook-Pro:~$ cd davegrohl MacBook-Pro:~/davegrohl$ make
***
A bunch of stuff happens here ***
Make succeeded...
Created: dave
MacBook-Pro:~/davegrohl$
```

To execute Dave, press the spacebar to view the status of each task. Dave's threads that perform dictionary attacks will be enclosed in parentheses, while those that perform incremental attacks will be enclosed in square brackets. Dave will search for plain text files in a folder named 'wordlists' for dictionary attacks, and it will dedicate a separate thread for each file it finds.



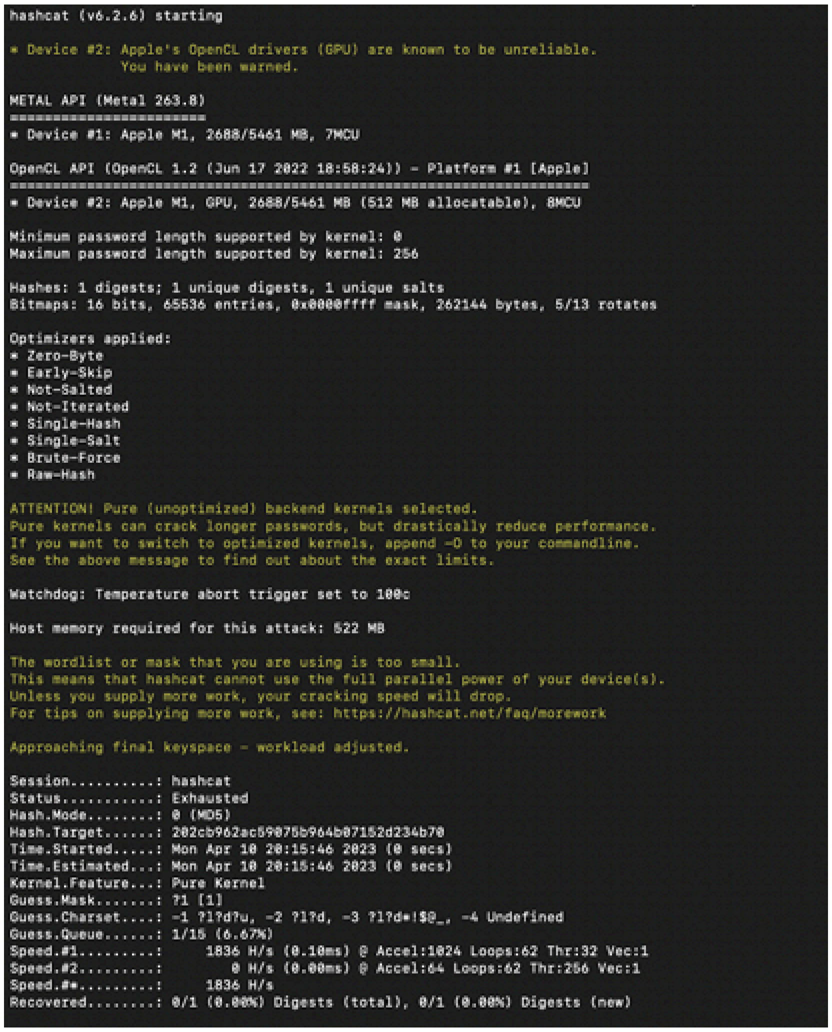
```
MacBook-Pro:~/davegrohl$ sudo ./dave -u someuser
-- Loaded PBKDF2 (Salted SHA512) hash...
-- Starting attack
TIME
0000:00:08
0000:00:14
0000:00:20
GUESSES
-- Found password: 'shorty'
(dictionary attack)
351 (aaru) (loveme) [x] [86n] [bpc] [2s5] [ojf] [wkea] [521a] [caha] 613 (abaculus) (samantha) [9a] [38n] [t3c] [an5] [yjf] [k4ea]
[dmla] [ieha] 875 (abandoning) (spongebob) [pe] [n7n] [x3c] [8n5] [bvf] [25ea] [odla] [weha]
Finished in 31.330 seconds 1,318 guesses... 42 guesses per second.
```

In this scenario, I intentionally selected a password that I knew Dave could easily guess. However, when utilizing PBKDF2, the operating system (OS X) slows down the password hashing process to enhance password security but makes it less feasible for brute force attacks.

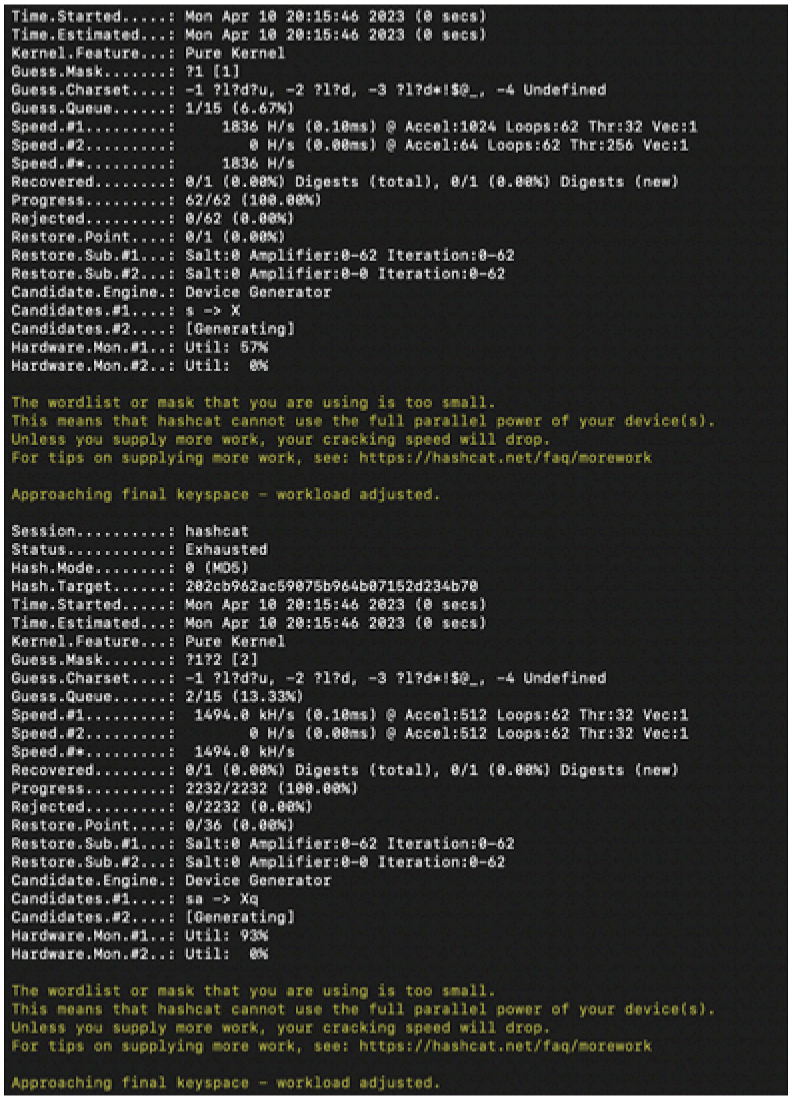
d) HashCat

Hashcat is a powerful password recovery tool designed to efficiently crack hashed passwords. It utilizes various attack modes, such as brute-force, dictionary, and mask attacks, to decipher passwords encrypted using popular hashing algorithms like MD5, SHA-1, and bcrypt. Security professionals, researchers, and ethical hackers often use Hashcat to assess the strength of passwords and enhance overall system security through vulnerability testing and password auditing.

It is now possible to utilize Hashcat on a via the terminal. To do this, a hash.txt file was created on the Desktop, containing the MD5 hash of a chosen password, which in this case was "123". The corresponding MD5 hash for "123" is "202cb962ac59075b964b07152d234b70". To indicate the attack mode and hash type in Hashcat, one can use the options-a and-m, respectively.

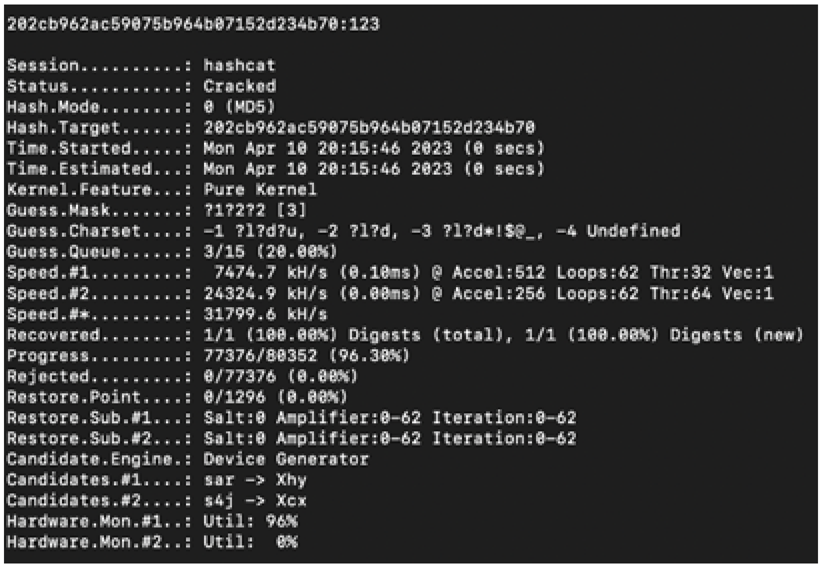


```
hashcat (v6.2.6) starting

* Device #2: Apple's OpenCL drivers (GPU) are known to be unreliable.
             You have been warned.

METAL API (Metal 263.8)
=======================
* Device #1: Apple M1, 2688/5461 MB, 7MCU

OpenCL API (OpenCL 1.2 (Jun 17 2022 18:58:24)) - Platform #1 [Apple]
====================================================================
* Device #2: Apple M1, GPU, 2688/5461 MB (512 MB allocatable), 8MCU

Minimum password length supported by kernel: 0
Maximum password 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

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Brute-Force
* Raw-Hash

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 100c

Host memory required for this attack: 522 MB

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 0 (MD5)
Hash.Target......: 202cb962ac59075b964b07152d234b70
Time.Started.....: Mon Apr 10 20:15:46 2023 (0 secs)
Time.Estimated...: Mon Apr 10 20:15:46 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?1 [1]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue......: 1/15 (6.67%)
Speed.#1.........:     1836 H/s (0.10ms) @ Accel:1024 Loops:62 Thr:32 Vec:1
Speed.#2.........:        0 H/s (0.00ms) @ Accel:64 Loops:62 Thr:256 Vec:1
Speed.#*.........:     1836 H/s
Recovered........: 0/1 (0.00%) Digests (total), 0/1 (0.00%) Digests (new)
```



```
Time.Started.....: Mon Apr 10 20:15:46 2023 (0 secs)
Time.Estimated...: Mon Apr 10 20:15:46 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?1 [1]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue......: 1/15 (6.67%)
Speed.#1.........:     1836 H/s (0.10ms) @ Accel:1024 Loops:62 Thr:32 Vec:1
Speed.#2.........:        0 H/s (0.00ms) @ Accel:64 Loops:62 Thr:256 Vec:1
Speed.#*.........:     1836 H/s
Recovered........: 0/1 (0.00%) Digests (total), 0/1 (0.00%) Digests (new)
Progress.........: 62/62 (100.00%)
Rejected.........: 0/62 (0.00%)
Restore.Point....: 0/1 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-62 Iteration:0-62
Restore.Sub.#2...: Salt:0 Amplifier:0-0 Iteration:0-62
Candidate.Engine.: Device Generator
Candidates.#1....: s -> X
Candidates.#2....: [Generating]
Hardware.Mon.#1..: Util: 57%
Hardware.Mon.#2..: Util:  0%

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 0 (MD5)
Hash.Target......: 202cb962ac59075b964b07152d234b70
Time.Started.....: Mon Apr 10 20:15:46 2023 (0 secs)
Time.Estimated...: Mon Apr 10 20:15:46 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?1?2 [2]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue......: 2/15 (13.33%)
Speed.#1.........:  1494.0 kH/s (0.10ms) @ Accel:512 Loops:62 Thr:32 Vec:1
Speed.#2.........:        0 H/s (0.00ms) @ Accel:512 Loops:62 Thr:32 Vec:1
Speed.#*.........:  1494.0 kH/s
Recovered........: 0/1 (0.00%) Digests (total), 0/1 (0.00%) Digests (new)
Progress.........: 2232/2232 (100.00%)
Rejected.........: 0/2232 (0.00%)
Restore.Point....: 0/36 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-62 Iteration:0-62
Restore.Sub.#2...: Salt:0 Amplifier:0-0 Iteration:0-62
Candidate.Engine.: Device Generator
Candidates.#1....: sa -> Xq
Candidates.#2....: [Generating]
Hardware.Mon.#1..: Util: 93%
Hardware.Mon.#2..: Util:  0%

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.
```

Password Cracking:



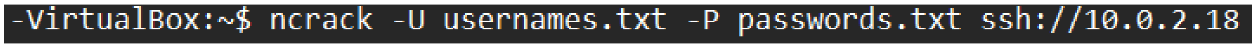
```
202cb962ac59075b964b07152d234b70:123

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
Hash.Target......: 202cb962ac59075b964b07152d234b70
Time.Started.....: Mon Apr 10 20:15:46 2023 (0 secs)
Time.Estimated...: Mon Apr 10 20:15:46 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?1?2?2 [3]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue......: 3/15 (20.00%)
Speed.#1.........:  7474.7 kH/s (0.10ms) @ Accel:512 Loops:62 Thr:32 Vec:1
Speed.#2.........: 24324.9 kH/s (0.00ms) @ Accel:256 Loops:62 Thr:64 Vec:1
Speed.#*.........: 31799.6 kH/s
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 77376/80352 (96.30%)
Rejected.........: 0/77376 (0.00%)
Restore.Point....: 0/1296 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-62 Iteration:0-62
Restore.Sub.#2...: Salt:0 Amplifier:0-62 Iteration:0-62
Candidate.Engine.: Device Generator
Candidates.#1....: sar -> Xhy
Candidates.#2....: s4j -> Xcx
Hardware.Mon.#1..: Util: 96%
Hardware.Mon.#2..: Util:  0%
```

Hence the password is cracked successfully.

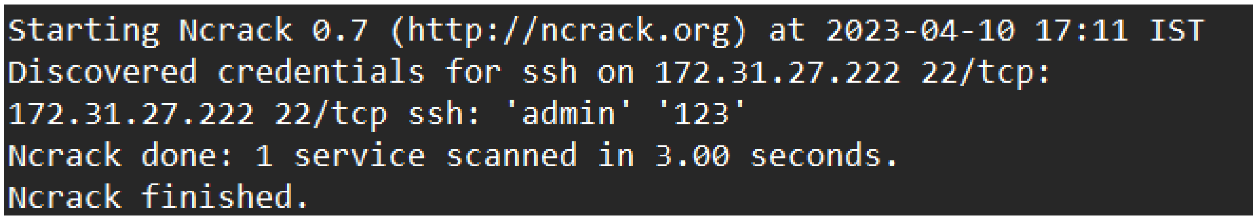
e) Ncrack:

To check for any potential vulnerabilities in the form of weak or commonly used passwords, a dictionary-based brute force attack is performed using ncrack on the Linux root credentials that are being utilized for SSH on a Linux virtual machine. To carry out this attack, a list of frequently used usernames is loaded from usernames.txt, while a default list of common passwords is loaded from passwords.txt. The goal is to determine if any security weaknesses can be exploited using this method.

The following is used to start with ncrack.



```
-VirtualBox:~$ ncrack -U usernames.txt -P passwords.txt ssh://10.0.2.18
```



```
Starting Ncrack 0.7 (http://ncrack.org) at 2023-04-10 17:11 IST
Discovered credentials for ssh on 172.31.27.222 22/tcp:
172.31.27.222 22/tcp ssh: 'admin' '123'
Ncrack done: 1 service scanned in 3.00 seconds.
Ncrack finished.
```

We can see ncrack successfully find the password of user ‘admin’

Comparison between them:

We will compare and contrast the features and capabilities of the following password cracking tools: John the Ripper, FcrackZip, Hashcat, DaveGrohl, and Ncrack. We will analyze the strengths and weaknesses of each tool based on various factors, including the type of attack, the type of password that can be guessed, the platform, and the time taken to crack a password.

John the Ripper: John the Ripper is a popular password cracking tool that can be used to crack passwords from various operating systems, including Unix, Linux, Windows, and macOS. John the Ripper supports several cracking modes, including dictionary attacks, brute force attacks, and hybrid attacks. It can also crack passwords encrypted with several hashing algorithms. John the Ripper is an excellent tool for offline password cracking.

FcrackZip: It's a command-line tool used for cracking password-protected ZIP files. It employs brute-force and dictionary-based attacks to guess the password of encrypted ZIP archives. The tool supports various attack modes, including dictionary attack, brute-force attack, and incremental brute-force attack, making it versatile for different scenarios. Fcrackzip is commonly used by security professionals and penetration testers to assess the strength of password protection on ZIP files and recover lost passwords. It is available for Unix-like operating systems and is often included in security-focused toolsets and distributions.

Hashcat: Hashcat is a powerful password cracking tool that can crack passwords from various operating systems and applications, including Windows, macOS, Unix, and Android. It supports several cracking modes, including dictionary attacks, brute force attacks, and mask attacks. Hashcat can also crack passwords encrypted with several hashing algorithms. It is one of the fastest password cracking tools available, and it can run on various hardware, including GPUs.

DaveGrohl: DaveGrohl is not a password cracking tool. Instead, it’s a famous musician and the lead vocalist of the Foo Fighters.

Ncrack: Ncrack is a network authentication cracking tool that can be used to crack passwords from various network services, including SSH, RDP, FTP, and Telnet. It supports several cracking modes, including dictionary attacks, brute force attacks, and hybrid attacks. Ncrack is available for Linux, Windows, and macOS. It is an excellent tool for network security auditing.

Conclusion: In conclusion, each of the password cracking tools we have analyzed has its strengths and weaknesses.John the Ripper is a versatile tool for offline password cracking, FcrackZip supports various attack modes, including dictionary attack, brute-force attack, and incremental brute-force attack, making it versatile for different scenarios, Hashcat is one of the fastest password cracking tools available, and Ncrack is an excellent tool for network security auditing. When choosing a password cracking tool, it is essential to consider the specific requirements of your project and select the tool that best meets those requirements