CS392 – Secure System Design

Assignment 2 - Report

Name: **M Maheeth Reddy**	Roll No.: **1801CS31**	Date: **11-Apr-2021**

Task 1.1: Implement a Honeyword Based System

I have implemented the Erguler's Honeyword based scheme in **Ubuntu 18.04.5 LTS** environment.

I have written two C source codes:

1. **login_server.c** – this is the implementation of Login Server
2. **honeychecker.c** – this is the implementation of Honeychecker

I have also written four text files:

1. **F1.txt** – for storing usernames and honeyindex set
2. **F2.txt** – for storing index and corresponding password hashes
3. **F3.txt** – for storing sugarindex of each username. Usernames are hashed to protect from security breach
4. **F4.txt** – for storing usernames of honeypot accounts, in hashed format

I have included a **makefile** in my submission. So, all the C source codes can be compiled at once by executing the command '*make*'.

rockyou.txt is the wordlist file I used for password cracking in Task 1.2. I can't attach the file since it is huge. It can be downloaded by executing:

wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt

Implementation details

Assumptions:

1. Each password is 8 to 12 characters long
2. Each username has five honeywords and an actual password. In other words, I chose k = 6

In this implementation, there are two main parts: **Login process** and **Registration process**.

The **Login Process** involves the following steps occurring in the Login Server:

1. Take username(u_i) and password(g) as input from user
2. Check whether g is correct password for u_i :
    a. Obtain honeyindex set X_i of u_i from F1.txt
    b. Compare hash(g) with hashes in F2.txt whose index ∈ X_i
    c. If there is no match, then login fails. Else
        i. Check whether account is a honeypot account
        ii. If it is, report a security breach. Else
            1. Send <u_i, j> to honeychecker. (j is the index ∈ X_i, that matched with hash(g))
            2. Receive honeychecker's response: whether j is sugarindex or not.
            3. If it is, Login is successful. Else, take necessary action

The Honeychecker server's role in Login Process is to determine whether the **index j** it receives from Login server as <u_i, j>, **is the sugarindex for u_i or not** and send the result back to Login server.

The **Registration Process** is used to add a new user to the system. It is relatively simpler than the Login Process. The Login server takes the username and password as input. It sends these two fields to honeychecker server. It is the Honeychecker server's duty to update the F1, F2, F3 files with details of the new username and password.

From the protocols mentioned above for the two processes, it is clear that the Login Server and the Honeychecker must have a way to communicate with each other. So, I have implemented this communication channel by creating a TCP connection between login_server.c and honeychecker.c

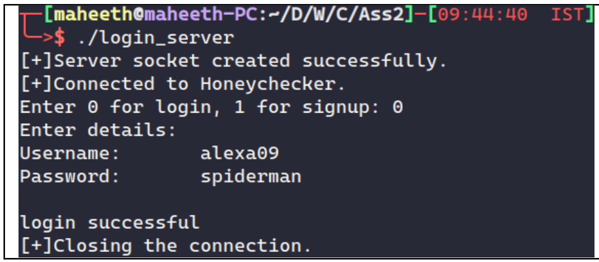
Instructions to run:

1. First, execute '*make*' command
2. Then, execute honeychecker program (./honeychecker)
3. Finally, execute login_server program (./login_server)

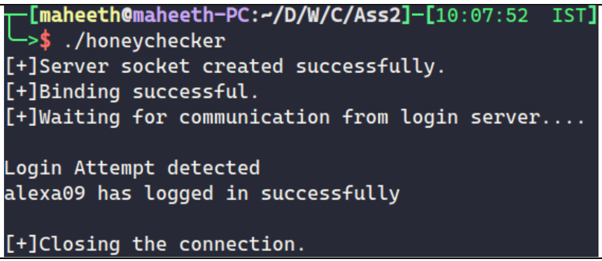
Note: Order is very important here.

Test Runs for Login Process:

1. Attempting to login from an existing username (but not honeypot account):
   a. By the right password (sugarword)



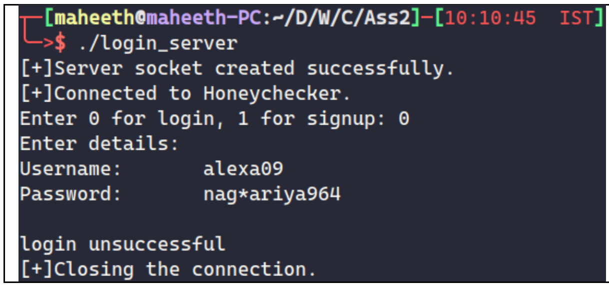
```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[09:44:40  IST]
└──>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 0
Enter details:
Username:       alexa09
Password:       spiderman

login successful
[+]Closing the connection.
```



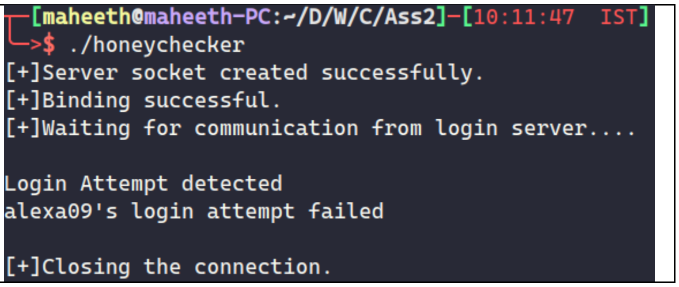
```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[10:07:52  IST]
└──>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Login Attempt detected
alexa09 has logged in successfully

[+]Closing the connection.
```

Note: User must first give either 0 or 1 as input to specify Login Process or Registration Process

   b. By a honeyword

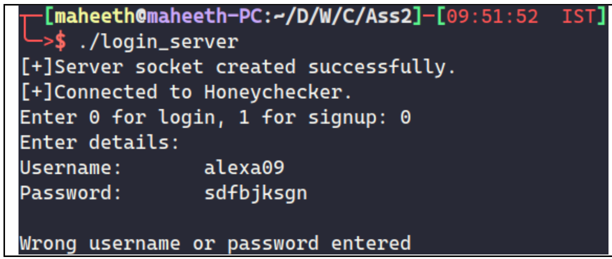


```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[10:10:45  IST]
└──>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 0
Enter details:
Username:       alexa09
Password:       nag*ariya964

login unsuccessful
[+]Closing the connection.
```

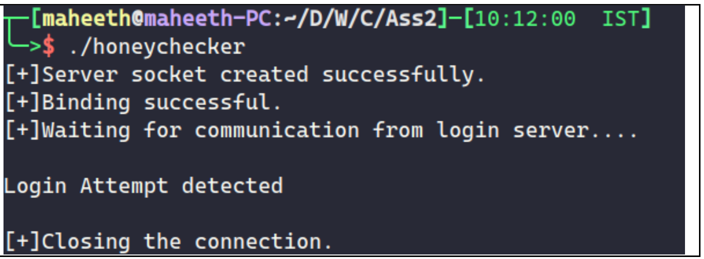


```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[10:11:47  IST]
└──>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Login Attempt detected
alexa09's login attempt failed

[+]Closing the connection.
```

   c. By a password that is not associated with any of the indices in the honeyindex set



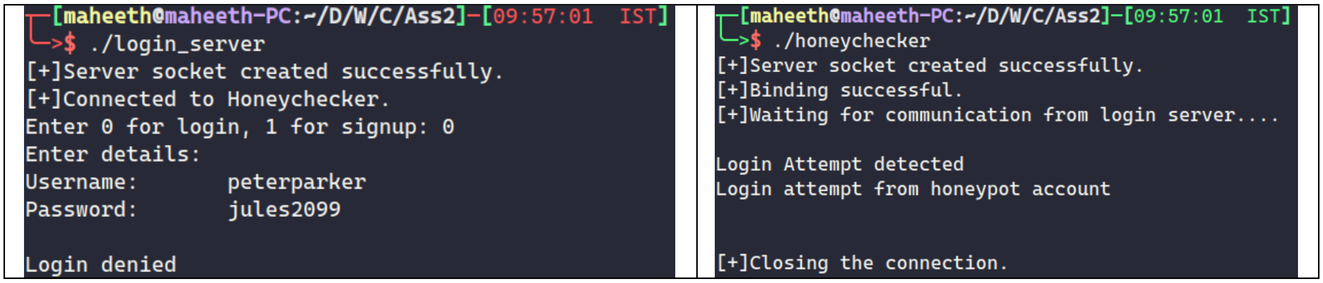
```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[09:51:52  IST]
└──>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 0
Enter details:
Username:       alexa09
Password:       sdfbjksgn

Wrong username or password entered
```



```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[10:12:00  IST]
└──>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Login Attempt detected

[+]Closing the connection.
```

Notice there is no other statement in the honeychecker execution except "Login attempt detected", unlike previous cases.

2. Attempting to login from a honeypot account

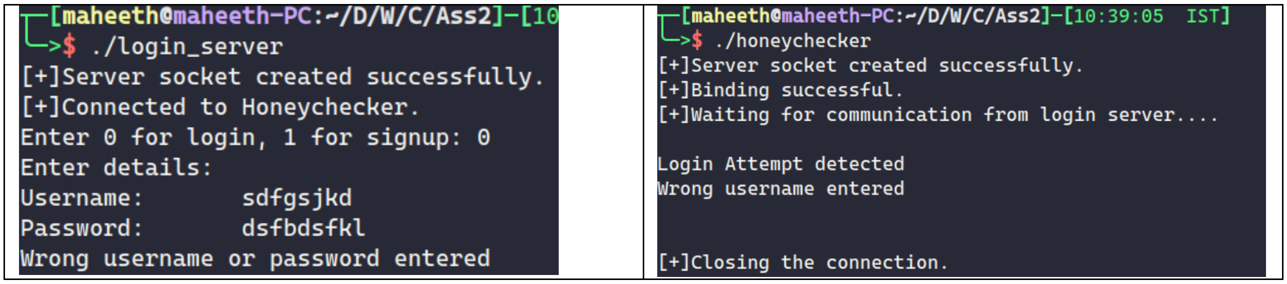


```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[09:57:01  IST]
└─>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 0
Enter details:
Username:       peterparker
Password:       jules2099

Login denied
```

```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[09:57:01  IST]
└─>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Login Attempt detected
Login attempt from honeypot account


[+]Closing the connection.
```

Notice that the login attempt has been blocked since it is a honeypot account

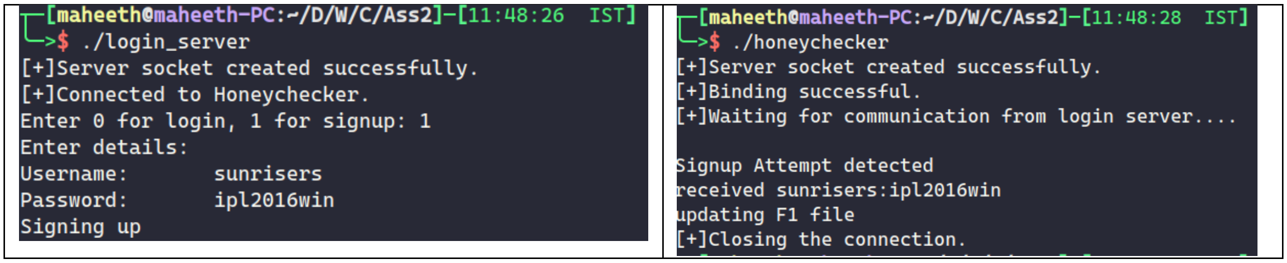
3. Attempting to login from a non-existing username



```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[10
└─>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 0
Enter details:
Username:       sdfgsjkd
Password:       dsfbdsfkl
Wrong username or password entered
```

```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[10:39:05  IST]
└─>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Login Attempt detected
Wrong username entered


[+]Closing the connection.
```

Test Run for Registration Process:



```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[11:48:26  IST]
└─>$ ./login_server
[+]Server socket created successfully.
[+]Connected to Honeychecker.
Enter 0 for login, 1 for signup: 1
Enter details:
Username:       sunrisers
Password:       ipl2016win
Signing up
```

```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]-[11:48:28  IST]
└─>$ ./honeychecker
[+]Server socket created successfully.
[+]Binding successful.
[+]Waiting for communication from login server....

Signup Attempt detected
received sunrisers:ipl2016win
updating F1 file
[+]Closing the connection.
```

Task 1.2: Use of Password Cracking Tool

Password Cracking is used by administrators to detect weak passwords in their organization. For this purpose, I installed "**John The Ripper**" tool using the command '*sudo apt install john*'. It is a commonly used password cracking tool. It comes preinstalled in penetration testing operating systems like Kali Linux.

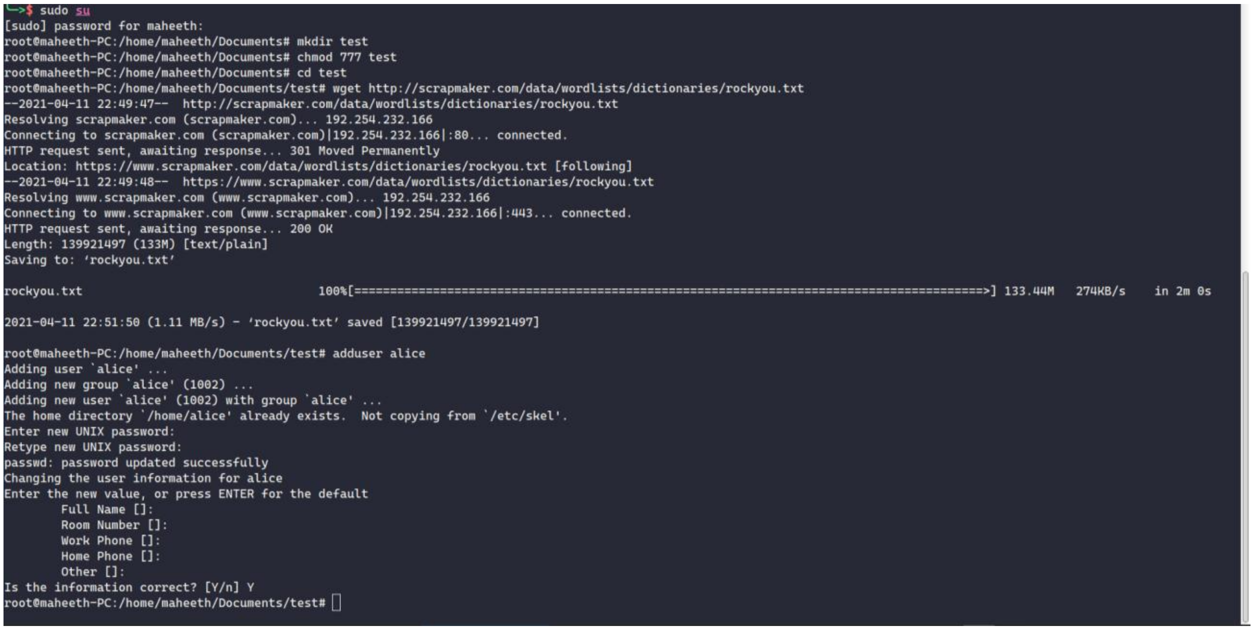
In this assignment, I first used John The Ripper to **detect any weak passwords in the /etc/shadow file present in my system**.

I executed the following commands in terminal, in order:

1. sudo su
2. mkdir test
3. chmod 777 test
4. cd test
5. wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt

By the above commands, I created a folder "test" with superuser privileges and downloaded a wordlist file called "rockyou.txt" into that "test" folder.

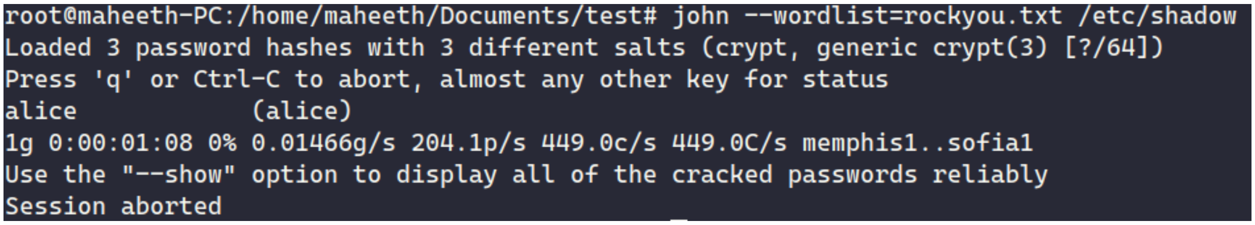
I executed the command adduser alice (when prompted, type the password as “alice”). By this, we are adding a user to the system with a weak password.



```
└─>$ sudo su
[sudo] password for maheeth:
root@maheeth-PC:/home/maheeth/Documents# mkdir test
root@maheeth-PC:/home/maheeth/Documents# chmod 777 test
root@maheeth-PC:/home/maheeth/Documents# cd test
root@maheeth-PC:/home/maheeth/Documents/test# wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
--2021-04-11 22:49:47--  http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
Resolving scrapmaker.com (scrapmaker.com)... 192.254.232.166
Connecting to scrapmaker.com (scrapmaker.com)|192.254.232.166|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.scrapmaker.com/data/wordlists/dictionaries/rockyou.txt [following]
--2021-04-11 22:49:48--  https://www.scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
Resolving www.scrapmaker.com (www.scrapmaker.com)... 192.254.232.166
Connecting to www.scrapmaker.com (www.scrapmaker.com)|192.254.232.166|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 139921497 (133M) [text/plain]
Saving to: ‘rockyou.txt’

rockyou.txt                                 100%[=====================================================================================>] 133.44M   274KB/s    in 2m 0s

2021-04-11 22:51:50 (1.11 MB/s) - ‘rockyou.txt’ saved [139921497/139921497]

root@maheeth-PC:/home/maheeth/Documents/test# adduser alice
Adding user `alice' ...
Adding new group `alice' (1002) ...
Adding new user `alice' (1002) with group `alice' ...
The home directory `/home/alice' already exists.  Not copying from `/etc/skel'.
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for alice
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
root@maheeth-PC:/home/maheeth/Documents/test# 
```

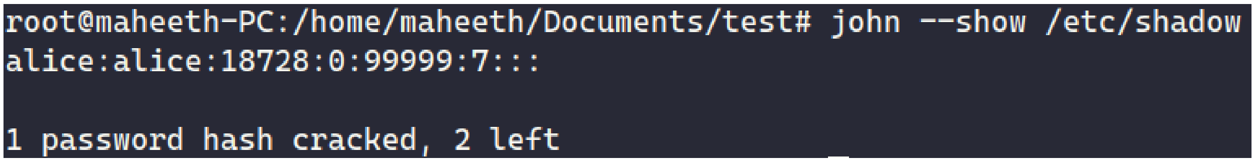
Now we can use John The Ripper to identify this weak password.

Command: john --wordlist=rockyou.txt /etc/shadow



```
root@maheeth-PC:/home/maheeth/Documents/test# john --wordlist=rockyou.txt /etc/shadow
Loaded 3 password hashes with 3 different salts (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
alice            (alice)
1g 0:00:01:08 0% 0.01466g/s 204.1p/s 449.0c/s 449.0C/s memphis1..sofia1
Use the "--show" option to display all of the cracked passwords reliably
Session aborted
```

Command: john --show /etc/shadow

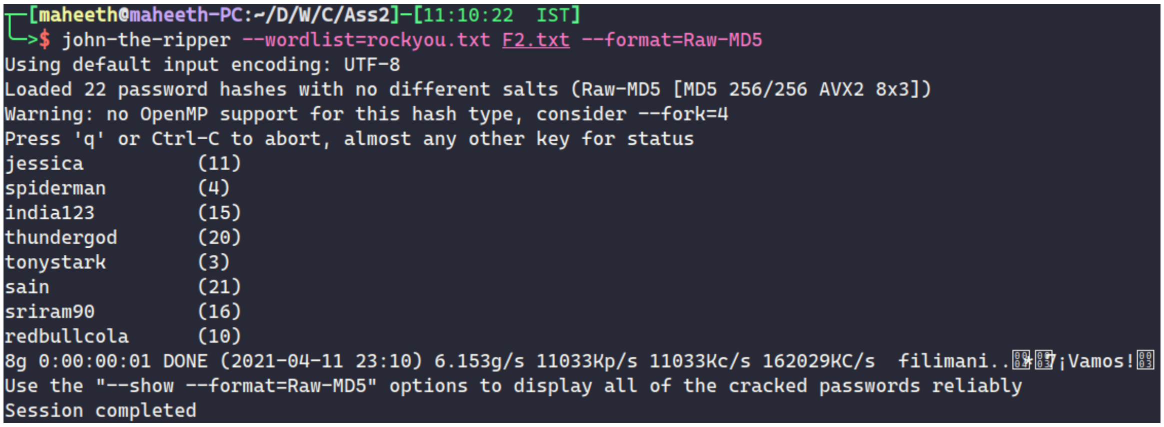


```
root@maheeth-PC:/home/maheeth/Documents/test# john --show /etc/shadow
alice:alice:18728:0:99999:7:::

1 password hash cracked, 2 left
```

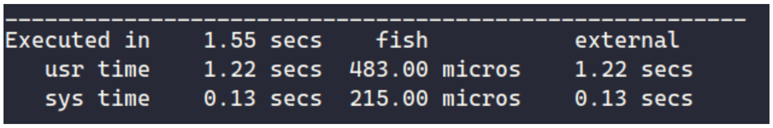
Using John The Ripper to detect weak passwords in honeyword system

I executed ‘sudo apt install john’ to install John The Ripper. Due to this version 1.8.0 got installed which unfortunately doesn’t support MD5 hashes. But I used MD5 hashing for my password file F2.txt. So I used ‘sudo snap install john-the-ripper’ to install the latest version 1.9.0 which supports MD5 hashing. I executed equivalents of above commands and their output is below.

Command: john-the-ripper --wordlist=rockyou.txt F2.txt --format=Raw-MD5

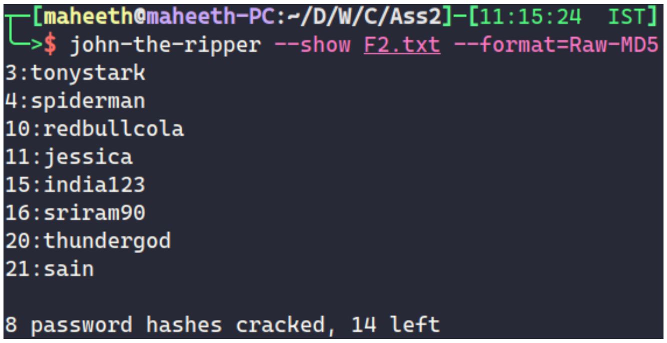


```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[11:10:22  IST]
└─>$ john-the-ripper --wordlist=rockyou.txt F2.txt --format=Raw-MD5
Using default input encoding: UTF-8
Loaded 22 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
jessica          (11)
spiderman        (4)
india123         (15)
thundergod       (20)
tonystark        (3)
sain             (21)
sriram90         (16)
redbullcola      (10)
8g 0:00:00:01 DONE (2021-04-11 23:10) 6.153g/s 11033Kp/s 11033Kc/s 162029KC/s  filimani..␄*␃7¡Vamos!␃
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed
```

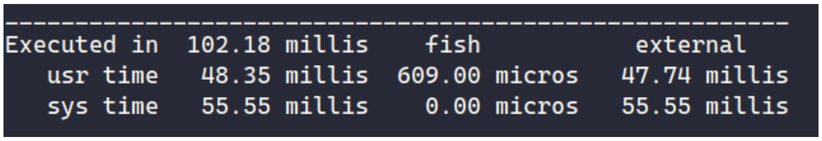


```
------------------------------------------------------------
Executed in    1.55 secs    fish           external
   usr time    1.22 secs  483.00 micros    1.22 secs
   sys time    0.13 secs  215.00 micros    0.13 secs
```

Command: john-the-ripper --show F2.txt --format=Raw-MD5



```
┌─[maheeth@maheeth-PC:~/D/W/C/Ass2]─[11:15:24  IST]
└─>$ john-the-ripper --show F2.txt --format=Raw-MD5
3:tonystark
4:spiderman
10:redbullcola
11:jessica
15:india123
16:sriram90
20:thundergod
21:sain

8 password hashes cracked, 14 left
```



```
------------------------------------------------------------
Executed in  102.18 millis    fish           external
   usr time   48.35 millis  609.00 micros   47.74 millis
   sys time   55.55 millis    0.00 micros   55.55 millis
```