EN4720: Security in Cyber-Physical Systems
Exercise — Authentication

Name: Thalagala B. P.
Index No: 180631J

May 15, 2023

This is an individual exercise!
Due Date: 19 May 2023 by 11.59 PM

Content adopted from the Udacity Security Engineer Nano-Degree and ECU CSI 1101.

This exercise has to be carried out using a Linux-based PC/virtual machine. Read all the instructions and questions before attempting the exercise. Add answers under each question in the Questions section and submit the resulting PDF. All files used in this exercise can be found in *ex3-resources* folder.

Instructions

1. Explore and understand how Unix systems store and manage passwords.
2. Download the shadow.hex file from the "files" folder. This file has been encrypted using RC4 encryption. 40 bits hexadecimal key: 5D 49 34 71 64. Decrypt the file using any available decryption tool (e.g., cryptool).
3. Download the "500_passwords.txt" file from the "files" folder. Run "hashcat" password cracking utility to crack the passwords in the decrypted shadow file with the help of the dictionary "500_passwords.txt". Please note that, to crack the passwords in the decrypted shadow file using hashcat, you need to take encrypted password fields of each user into a different file. You may rename that file as "encrypted_passwords.txt".
4. Download the "myhashes.txt", "mydictionary.txt", "myruleset.txt" files from the "files" folder. Note that, myhashes.txt containes MD5 hashes. Run a rule-based attack using "hashcat" password cracking utility to crack the passwords in the myhashes.txt file with the help of the dictionary "mydictionary.txt" and the ruleset myruleset.txt.
5. Answer the questions given below.

Questions

1. What is the difference between /etc/passwd and /etc/shadow files? Complete Table 1.

    /etc/passwd file stores the account details of the users on a given host machine. The content of the file is word-readable and everyone has the read permission for that file.

```
(user@host)-[/etc]
$ ls -lh | grep passwd
-rw-r--r--  1 root root   2.0K Apr  2 14:44 passwd
-rw-r--r--  1 root root   2.0K Apr  2 14:38 passwd-
```

    /etc/shadow file stores the details about the passwords of users and hashed passwords. The shadow file can only be read by the root account.

```
(user@host)-[/etc]
$ ls -lh | grep shadow
-rw-r-----  1 root shadow  770 Apr  2 14:44 gshadow
-rw-r-----  1 root shadow  761 Apr  2 14:44 gshadow-
-rw-r-----  1 root shadow  968 Apr  2 14:44 shadow
-rw-r-----  1 root shadow  949 Apr  2 14:38 shadow-
```

Table 1: Difference between /etc/passwd amd etc/shadow files

Attribute	**/etc/passwd**	**etc/shadow**
Number of fields	7	9
Accessible users	Anyone can read	Only `root` user and members of `shadow` group can read

2. You will find seven fields in the /etc/passwd file as indicated in Figure 1. Explain the purpose of each field.

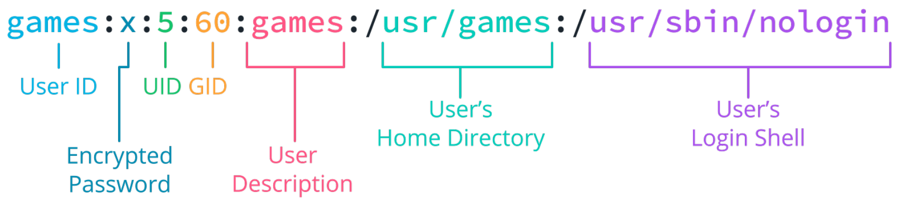


Figure 1: Example storage format of /etc/passwd file

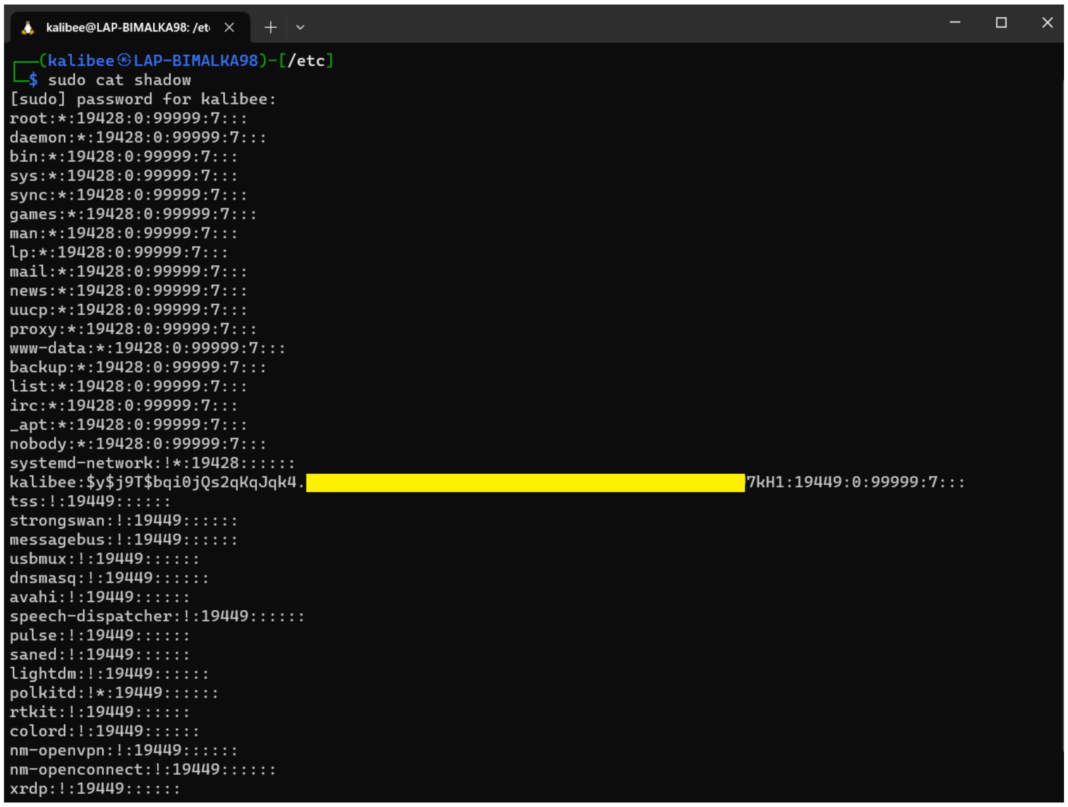
(a) `games`: *name of the user for whom this entry corresponds to.*
(b) `x` : *indicates that a password exists for the user. However, the password is stored in the /etc/shadow file. If instead of x it shows a ! symbol, this indicates that a password does not exist.*
(c) `5` : *User ID of this user. User IDs for created users begin from 1000 and run up to 59999*
(d) `60`: *Group ID of the group this user belongs to.*
(e) `games`: *can contain multiple comma separated sub-fields of information inclusive of full name and telephone numbers. Here, no telephone numbers have been provided.*

(f) `/usr/games` : *location of home directory assigned to this user.*

(g) `/usr/sbin/nologin` : *default shell assigned to this user.*

Note: Information adopted from,
`https://www.maketecheasier.com/how-linux-stores-manages-user-passwords/`

3. Dump the content of your /etc/shadow file to the terminal using sudo cat /etc/shadow and add a screenshot.



```
kalibee@LAP-BIMALKA98: /etc
┌──(kalibee㉿LAP-BIMALKA98)-[/etc]
└─$ sudo cat shadow
[sudo] password for kalibee:
root:*:19428:0:99999:7:::
daemon:*:19428:0:99999:7:::
bin:*:19428:0:99999:7:::
sys:*:19428:0:99999:7:::
sync:*:19428:0:99999:7:::
games:*:19428:0:99999:7:::
man:*:19428:0:99999:7:::
lp:*:19428:0:99999:7:::
mail:*:19428:0:99999:7:::
news:*:19428:0:99999:7:::
uucp:*:19428:0:99999:7:::
proxy:*:19428:0:99999:7:::
www-data:*:19428:0:99999:7:::
backup:*:19428:0:99999:7:::
list:*:19428:0:99999:7:::
irc:*:19428:0:99999:7:::
_apt:*:19428:0:99999:7:::
nobody:*:19428:0:99999:7:::
systemd-network:!*:19428::::::
kalibee:$y$j9T$bqi0jQs2qKqJqk4.[redacted]7kH1:19449:0:99999:7:::
tss:!:19449::::::
strongswan:!:19449::::::
messagebus:!:19449::::::
usbmux:!:19449::::::
dnsmasq:!:19449::::::
avahi:!:19449::::::
speech-dispatcher:!:19449::::::
pulse:!:19449::::::
saned:!:19449::::::
lightdm:!:19449::::::
polkitd:!*:19449::::::
rtkit:!:19449::::::
colord:!:19449::::::
nm-openvpn:!:19449::::::
nm-openconnect:!:19449::::::
xrdp:!:19449::::::
```

Figure 2: Content of /etc/shadow file

4. Identify the different fields present in one line in the shadow file and explain the purpose of each field. Clearly mark the fields and provide the explanation.

there are 9 fields in an entry in the shadow file, and each field can be identified as follows. Let us consider the entry of the user "kalibee"

(a) `kalibee` - *name of the user for whom this entry corresponds to*

(b) `$y$j9T$bqi0jQs2qKqJqk4...7kH1` - *hashed user password. Along with the information about, the hashing algorithm, used salt value.*

(c) `19449` - *last password change time (in days since Jan 1, 1970)*

(d) `0` - *the number of days after which the password can be changed. A value of 0 means the password can be changed at any time.*

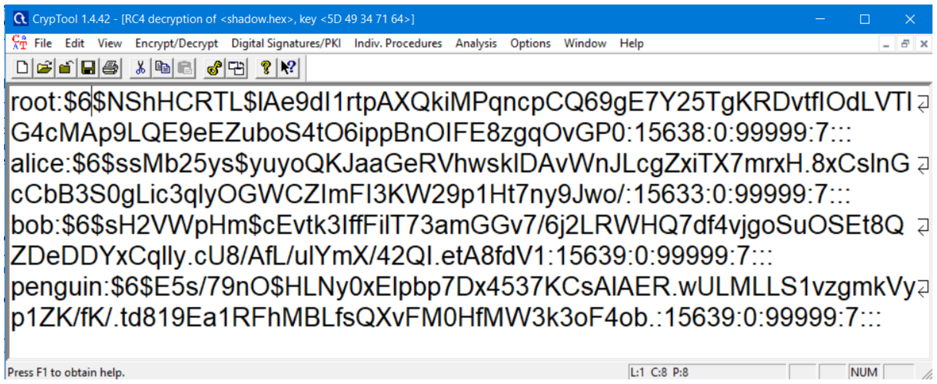
(e) `99999` - *the number of days after which the password must be changed. A value of 99999 indicates a user can retain the password as long as desired.*

(f) `7` - *if the password is set to expire, this fields indicates the number of days to warn the user about password expiry.*

(g) *::: - Three empty fields. The first one indicates the number of days to wait after password expiry, following which the account will be disabled. The second one indicates the number of days since January 1, 1970, that an account has been disabled. The third field is reserved for future use. The empty fields indicate that the existing password for this user has not expired and is not set to expire soon.*

Note: Information adopted from,
https://www.maketecheasier.com/how-linux-stores-manages-user-passwords/

5. Complete Step #2 mentioned under **Instructions**. Add a screenshot of the decrypted shadow.hex file content. What is the type of hash in the decrypted shadow file?

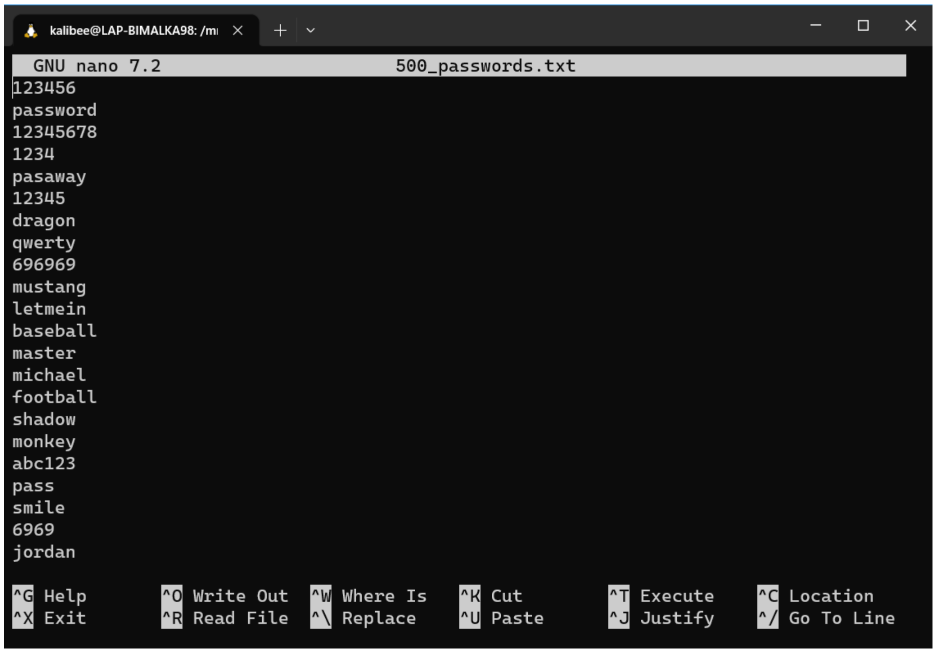

CrypTool 1.4.42 - [RC4 decryption of <shadow.hex>, key <5D 49 34 71 64>]
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help

```
root:$6$NShHCRTL$lAe9dI1rtpAXQkiMPqncpCQ69gE7Y25TgKRDvtflOdLVTI
G4cMAp9LQE9eEZuboS4tO6ippBnOIFE8zgqOvGP0:15638:0:99999:7:::
alice:$6$ssMb25ys$yuyoQKJaaGeRVhwskIDAvWnJLcgZxiTX7mrxH.8xCsInG
cCbB3S0gLic3qlyOGWCZImFI3KW29p1Ht7ny9Jwo/:15633:0:99999:7:::
bob:$6$sH2VWpHm$cEvtk3IffFiIT73amGGv7/6j2LRWHQ7df4vjgoSuOSEt8Q
ZDeDDYxCqlly.cU8/AfL/uIYmX/42QI.etA8fdV1:15639:0:99999:7:::
penguin:$6$E5s/79nO$HLNy0xElpbp7Dx4537KCsAIAER.wULMLLS1vzgmkVy
p1ZK/fK/.td819Ea1RFhMBLfsQXvFM0HfMW3k3oF4ob.:15639:0:99999:7:::
```

Figure 3: Content of the decrypted shadow.hex file using the key 5D 49 34 71 64

All the password hashes start with hash ID, 6. This indicates that they use SHA-512 hashing algorithm.

6. Complete Step #3 mentioned under **Instructions**. Add a screenshot of a part of the "500_passwords.txt" file content using vi/vim/nano.



```
  GNU nano 7.2                     500_passwords.txt
123456
password
12345678
1234
pasaway
12345
dragon
qwerty
696969
mustang
letmein
baseball
master
michael
football
shadow
monkey
abc123
pass
smile
6969
jordan

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line
```

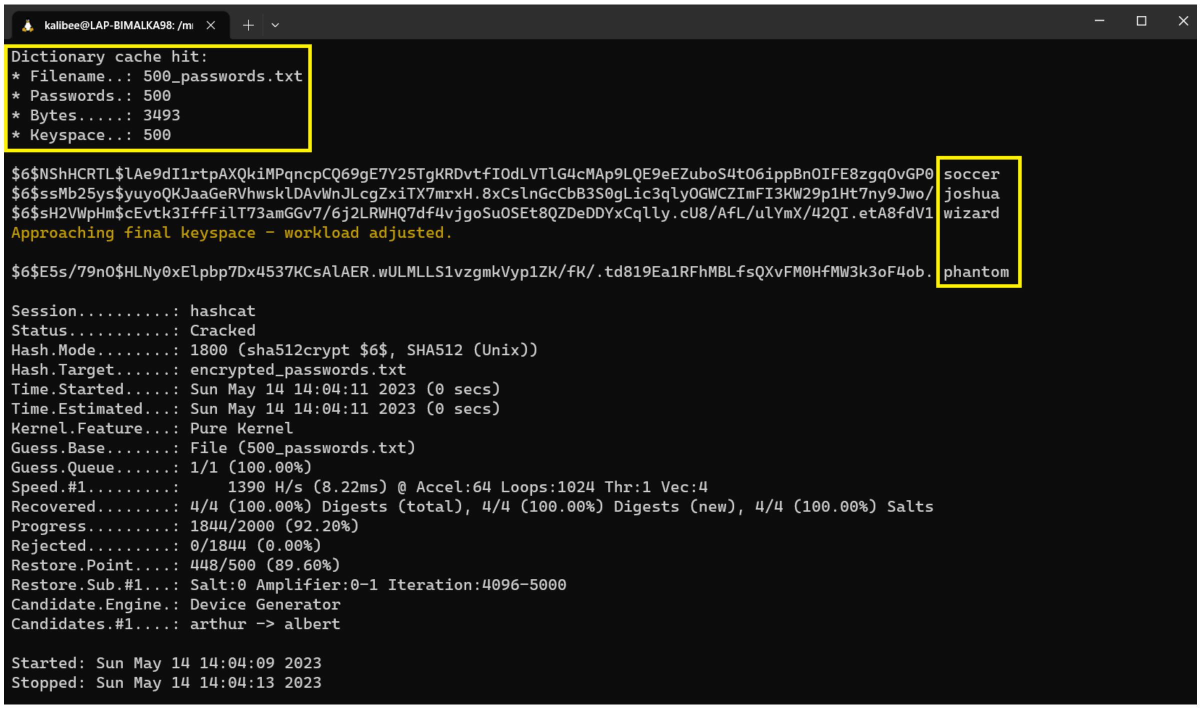
Figure 4: A part of the content of "500_passwords.txt" file

7. What is the command you used to crack the passwords in the shadow file? Briefly explain each field in the command.

```
hashcat -m 1800 -a 0 encrypted_passwords.txt 500_passwords.txt
    --potfile-path=cracked.pot
```

- `-m 1800` *specifies the code for the hash mode: 1800 - sha512crypt 6, SHA512 (Unix) in the category of 'Operating System'.*
- `-a 0` *specifies the 'Straight' attack mode that is dictionary attack.*
- *An optional parameter can be passed to specify the path to the '.pot' file, where the cracked passwords are saved.* `--potfile-path=cracked.pot`. *This will save the pot file in the same directory as the other input files.*

8. Add a screenshot of all the cracked passwords.



```
Dictionary cache hit:
* Filename..: 500_passwords.txt
* Passwords.: 500
* Bytes.....: 3493
* Keyspace..: 500

$6$NShHCRTL$lAe9dI1rtpAXQkiMPqncpCQ69gE7Y25TgKRDvtfIOdLVTlG4cMAp9LQE9eEZuboS4tO6ippBnOIFE8zgqOvGP0 soccer
$6$ssMb25ys$yuyoQKJaaGeRVhwsklDAvWnJLcgZxiTX7mrxH.8xCslnGcCbB3S0gLic3qlyOGWCZImFI3KW29p1Ht7ny9Jwo/ joshua
$6$sH2VWpHm$cEvtk3IffFilT73amGGv7/6j2LRWHQ7df4vjgoSuOSEt8QZDeDDYxCqlly.cU8/AfL/ulYmX/42QI.etA8fdV1 wizard
Approaching final keyspace - workload adjusted.

$6$E5s/79nO$HLNy0xElpbp7Dx4537KCsAlAER.wULMLLS1vzgmkVyp1ZK/fK/.td819Ea1RFhMBLfsQXvFM0HfMW3k3oF4ob. phantom

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1800 (sha512crypt $6$, SHA512 (Unix))
Hash.Target......: encrypted_passwords.txt
Time.Started.....: Sun May 14 14:04:11 2023 (0 secs)
Time.Estimated...: Sun May 14 14:04:11 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (500_passwords.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     1390 H/s (8.22ms) @ Accel:64 Loops:1024 Thr:1 Vec:4
Recovered........: 4/4 (100.00%) Digests (total), 4/4 (100.00%) Digests (new), 4/4 (100.00%) Salts
Progress.........: 1844/2000 (92.20%)
Rejected.........: 0/1844 (0.00%)
Restore.Point....: 448/500 (89.60%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4096-5000
Candidate.Engine.: Device Generator
Candidates.#1....: arthur -> albert

Started: Sun May 14 14:04:09 2023
Stopped: Sun May 14 14:04:13 2023
```

Figure 5: Cracked passwords using Hashcat tool

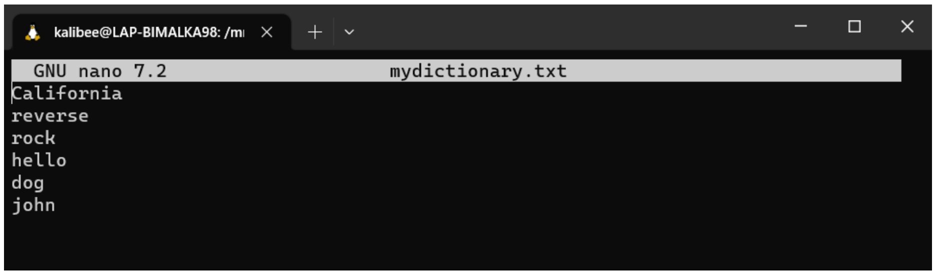
9. Provide recommendations to enhance the strength of the passwords.

- *When creating a password a combination of numbers, letters (upper case and/or lower case) and special characters must be used and the password must be sufficiently long to make it stronger.*
- *When creating a password common words (such as soccer, wizard, phantom (found above) and etc) and patterns (such as 123, 12345 and etc) must be strictly avoided. As these words can be easily cracked in no time as seen in the above scenario.*
- *A password must not contain personal information such as name, birth year and etc. as those can easily be combined to form a candidate set of passwords easily.*

10. Briefly explain about "Rule-based Attack" and how it uses time and memory for the hash cracking process using a diagram.

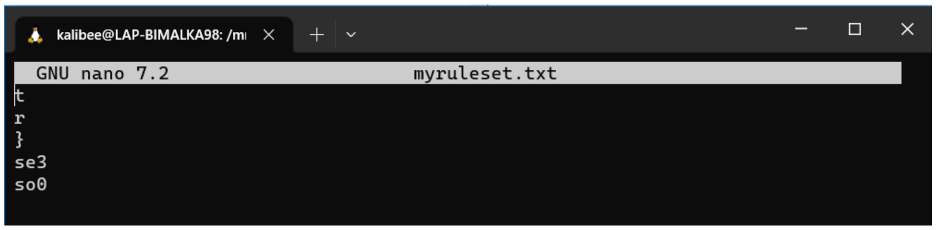
    While pure dictionary attacks use the bare passwords in a given file, rule-based attacks can generate new candidate passwords using a dictionary of passwords. In order to to do that, password cracking tools have functions to modify, cut or extend words and has conditional operators to skip some, etc. That makes it the most flexible, accurate and efficient attack.

11. Complete Step #4 mentioned under **Instructions**. Add a screenshot of the mydictionary.txt and myruleset.txt file contents. Define each rule in myrulset.txt using standard rules in the rule-based attack.



```
kalibee@LAP-BIMALKA98: /m
  GNU nano 7.2                         mydictionary.txt
California
reverse
rock
hello
dog
john
```

Figure 6: Content of the `mydictionary.txt` file



```
kalibee@LAP-BIMALKA98: /m
  GNU nano 7.2                           myruleset.txt
t
r
}
se3
so0
```

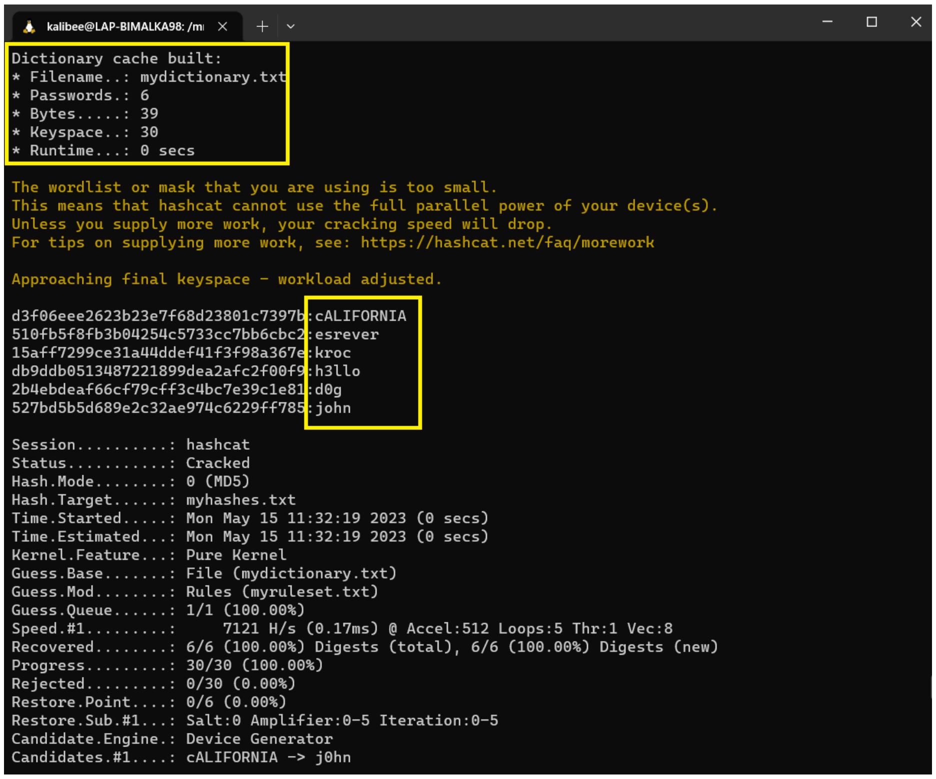
Figure 7: Content of the `myruleset.txt` file

Definitions of the rules shown in the Figure 7,

- `t` - *Toggle the case of all characters in word.*
- `r` - *Reverse the entire word*
- } - *Rotate the word right*
- `se3` *and* `so0`- *(sXY) Replace all instances of X with Y*

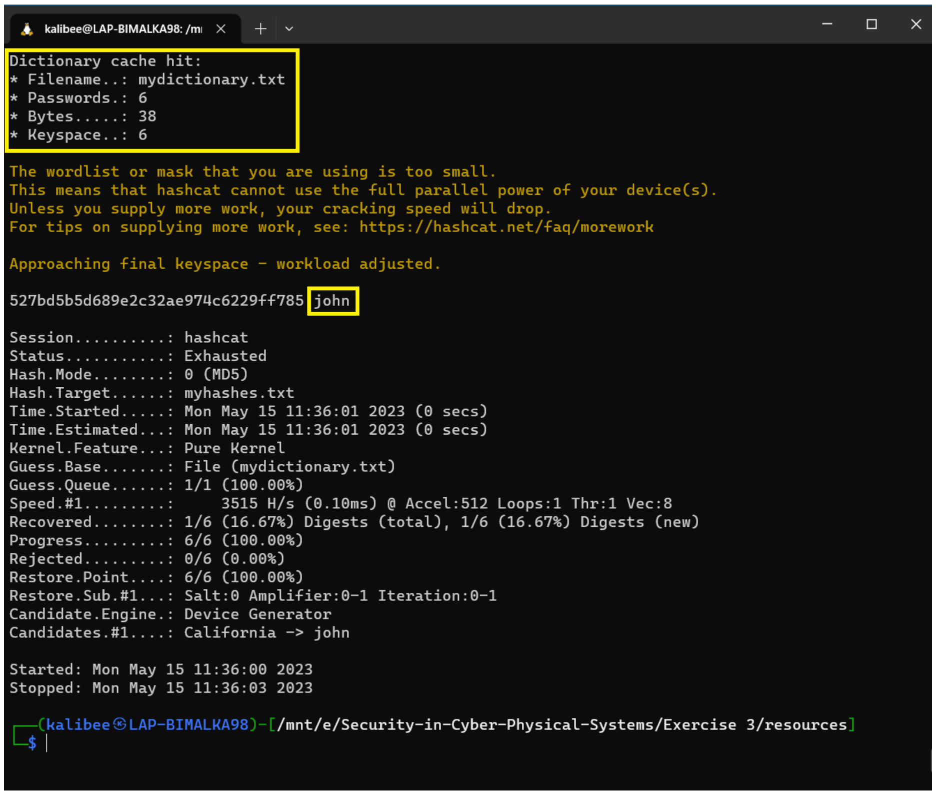
12. Add a screenshot of all the cracked passwords. Run a dictionary attack without myruleset.txt (using only the dictionary) and compare the two cracked files.

    Through the rule-based attack, the tool can generate new candidate passwords other than the original password set. This increases the keyspace to 30 passwords from only 6 passwords, which increases the effectiveness of the attack. Figure 8 illustrates that, through the rule-based attack, all the six passwords are cracked, whereas only one password is cracked through the bare dictionary attack (figure 9) .



```
Dictionary cache built:
* Filename..: mydictionary.txt
* Passwords.: 6
* Bytes.....: 39
* Keyspace..: 30
* Runtime...: 0 secs

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.

d3f06eee2623b23e7f68d23801c7397b:cALIFORNIA
510fb5f8fb3b04254c5733cc7bb6cbc2:esrever
15aff7299ce31a44ddef41f3f98a367e:kroc
db9ddb0513487221899dea2afc2f00f9:h3llo
2b4ebdeaf66cf79cff3c4bc7e39c1e81:d0g
527bd5b5d689e2c32ae974c6229ff785:john

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
Hash.Target......: myhashes.txt
Time.Started.....: Mon May 15 11:32:19 2023 (0 secs)
Time.Estimated...: Mon May 15 11:32:19 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (mydictionary.txt)
Guess.Mod........: Rules (myruleset.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     7121 H/s (0.17ms) @ Accel:512 Loops:5 Thr:1 Vec:8
Recovered........: 6/6 (100.00%) Digests (total), 6/6 (100.00%) Digests (new)
Progress.........: 30/30 (100.00%)
Rejected.........: 0/30 (0.00%)
Restore.Point....: 0/6 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-5 Iteration:0-5
Candidate.Engine.: Device Generator
Candidates.#1....: cALIFORNIA -> j0hn
```

Figure 8: Cracked passwords using the rule-based attack. **Command** : `hashcat -m 0 -a 0 myhashes.txt mydictionary.txt -r myruleset.txt --potfile-path=cracked.pot`



```
Dictionary cache hit:
* Filename..: mydictionary.txt
* Passwords.: 6
* Bytes.....: 38
* Keyspace..: 6

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.

527bd5b5d689e2c32ae974c6229ff785:john

Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 0 (MD5)
Hash.Target......: myhashes.txt
Time.Started.....: Mon May 15 11:36:01 2023 (0 secs)
Time.Estimated...: Mon May 15 11:36:01 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (mydictionary.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     3515 H/s (0.10ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered........: 1/6 (16.67%) Digests (total), 1/6 (16.67%) Digests (new)
Progress.........: 6/6 (100.00%)
Rejected.........: 0/6 (0.00%)
Restore.Point....: 6/6 (100.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: California -> john

Started: Mon May 15 11:36:00 2023
Stopped: Mon May 15 11:36:03 2023

┌──(kalibee㉿LAP-BIMALKA98)-[/mnt/e/Security-in-Cyber-Physical-Systems/Exercise 3/resources]
└─$ 
```

Figure 9: Cracked passwords using bare dictionary attack. **Command** : `hashcat -m 0 -a 0 myhashes.txt mydictionary.txt --potfile-path=cracked.pot`