

# TASK 4

In this digital world security and privacy is important because data is the nowadays asset. To protect our data from the third party members and attackers, we need to change the plain text into a cipher text. Changing the plain text into cipher text can be achieved with several processes like encryption and hashing.

## HASHING

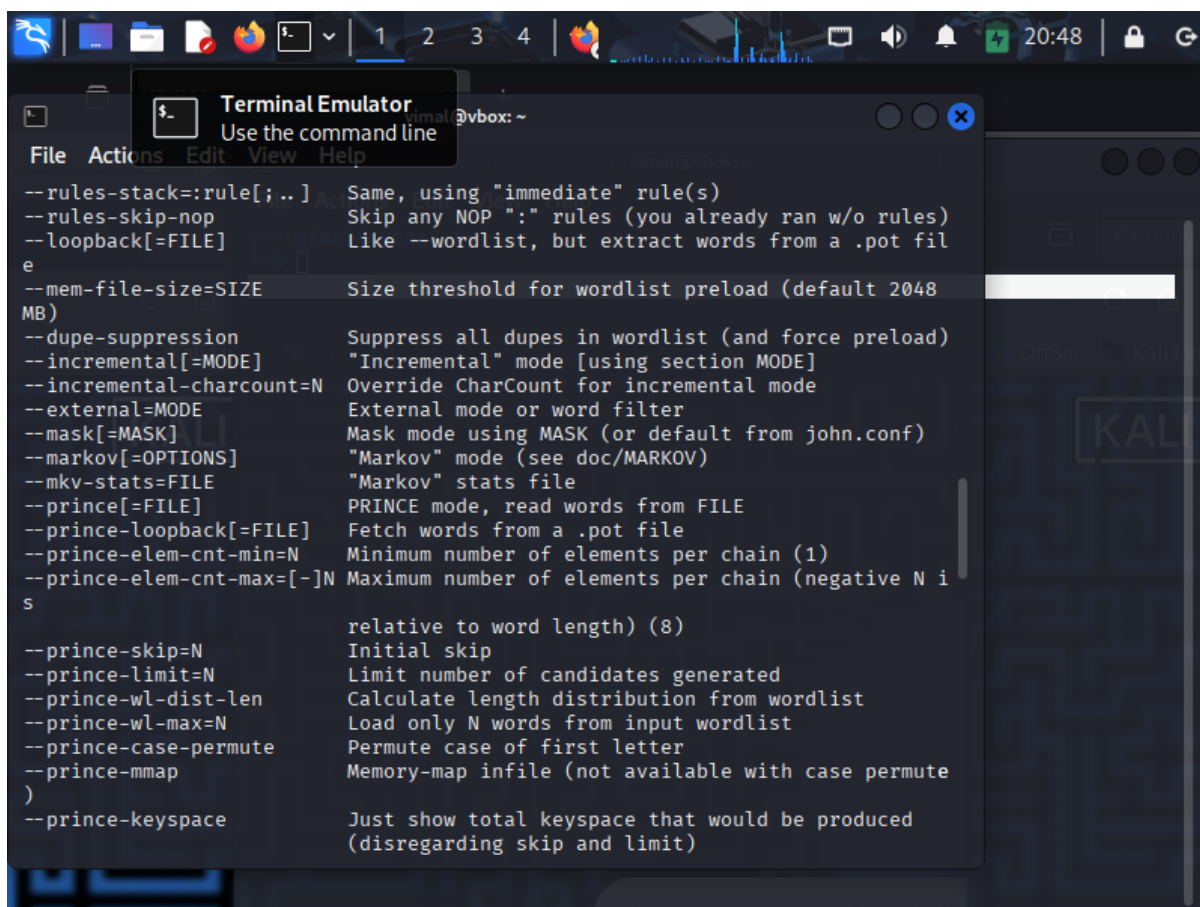
Hashing is a process of changing the text into a sequence of numbers, mostly in hexadecimal values. This process is fixed; we can't reverse it. This is useful to store the credentials at the server. Even the server doesn't know the original text or script.

Example **fad89cd3d010b686f0fb8aaf22d79576d589d29f648b42d917b7f5cafe6c0ed3** is the hash value of **WannaCry ransomware SHA256 hash.txt**

## ENCRYPTION

Encryption also does the same things like hashing, changing the plain text into a cipher text. But the working mechanism makes the difference here. In hashing, there is no reverse process, but in encryption, the both sides reverse process may occur. The changing process is achieved by the crypto keys like **public key and private key**.

- **HASHING** -ONE WAY PROCESS
- **ENCRYPTION** -TWO WAY PROCESS(crypto keys)



```
root@kali: ~  
--rules-stack=:rule[;..] Same, using "immediate" rule(s)  
--rules-skip-nop Skip any NOP ":" rules (you already ran w/o rules)  
--loopback[=FILE] Like --wordlist, but extract words from a .pot file  
e  
--mem-file-size=SIZE Size threshold for wordlist preload (default 2048  
MB)  
--dupe-suppression Suppress all dups in wordlist (and force preload)  
--incremental[=MODE] "Incremental" mode [using section MODE]  
--incremental-charcount=N Override CharCount for incremental mode  
--external=MODE External mode or word filter  
--mask[=MASK] Mask mode using MASK (or default from john.conf)  
--markov[=OPTIONS] "Markov" mode (see doc/MARKOV)  
--mkv-stats=FILE "Markov" stats file  
--prince[=FILE] PRINCE mode, read words from FILE  
--prince-loopback[=FILE] Fetch words from a .pot file  
--prince-elem-cnt-min=N Minimum number of elements per chain (1)  
--prince-elem-cnt-max=[-]N Maximum number of elements per chain (negative N i  
s  
relative to word length) (8)  
--prince-skip=N Initial skip  
--prince-limit=N Limit number of candidates generated  
--prince-wl-dist-len Calculate length distribution from wordlist  
--prince-wl-max=N Load only N words from input wordlist  
--prince-case-permute Permute case of first letter  
--prince-mmap Memory-map infile (not available with case permute  
)  
--prince-keyspace Just show total keyspace that would be produced  
(disregarding skip and limit)
```

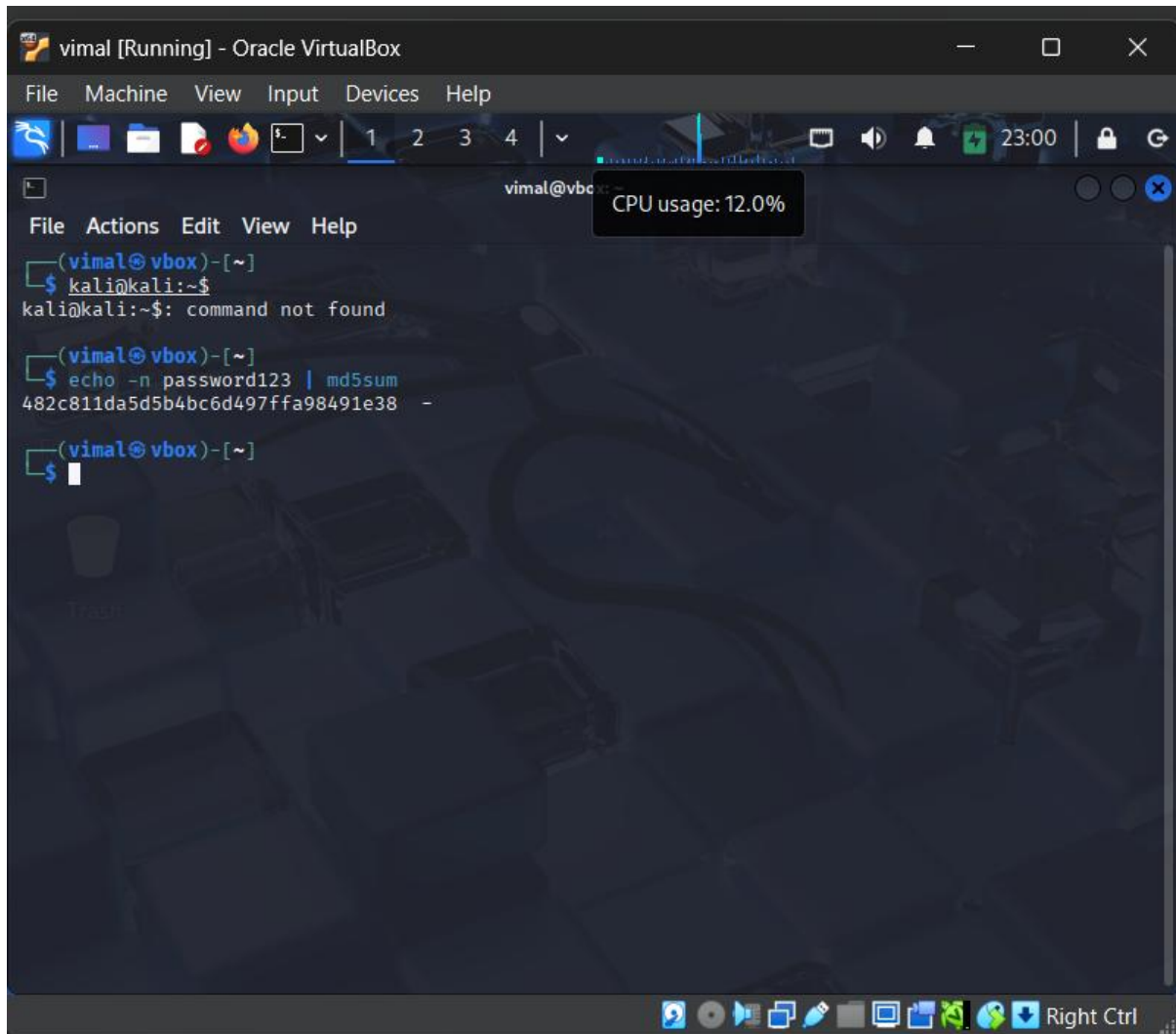
I have installed the john ripper in my vm, kali linux runn it in mu vm terminal and run the following commands

- `sudo apt update` : update the firmware.
- `sudo apt install john -y` : install the john ripper
- `john --help` : confirm installation
- `echo -n password123 | md5sum` : **Create a harsh using text**

Here the the password123 is the word to hashing

- **echo** : to print
- **-n** : to decleare there is no new line
- **Password123** : this is the text
- **Md5sum** : it is the command to hash the value

482c811da5d5b4bc6d497ffa98491e38 -this is the hash value



```
vimal [Running] - Oracle VirtualBox
File Machine View Input Devices Help
vimal@vbox: CPU usage: 12.0%
File Actions Edit View Help
(vimal@vbox)-[~]
$ kali@kali:~$
kali@kali:~$: command not found

(vimal@vbox)-[~]
$ echo -n password123 | md5sum
482c811da5d5b4bc6d497ffa98491e38 -

(vimal@vbox)-[~]
$
```