

CIS\_8080

# Password Cracking

Done by

Venkat Sai Nag Bhargav Sabbineni

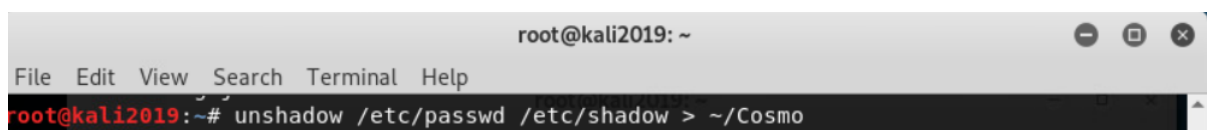
John The Ripper:

Observations:

Created user credentials in kali Linux 2019 as below

User Name	Password
Alpha	atlanta
Beta	colrodo@358
Gamma	system
Omega	car
Condo	Apple
Cosmo	file

Combined password and shadow files together into a Cosmo file using Unshadow Command

A screenshot of a terminal window titled 'root@kali2019: ~'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The command prompt shows 'root@kali2019:~# unshadow /etc/passwd /etc/shadow > ~/Cosmo'. The output of the command is not visible in the screenshot.

```
root@kali2019: ~
File Edit View Search Terminal Help
root@kali2019:~# unshadow /etc/passwd /etc/shadow > ~/Cosmo
```

Started running John- the ripper tool to perform dictionary attack and brute-force attack

Dictionary Attack:

After dictionary attack, it took less than 1 min to crack the passwords of Condo, Gamma, and Omega as the passwords were not strong at all and the password length was small

```

root@kali2019:~# john --wordlist=/usr/share/john/password.lst ~/Cosmo
Using default input encoding: UTF-8
Loaded 7 password hashes with 7 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
apple (Condo)
system (Gamma)
car (Omega)
3g 0:00:00:10 DONE (2022-09-21 10:40) 0.2824g/s 333.8p/s 1814c/s 1814C/s jussi..sss
Use the "-i show" option to display all of the cracked passwords reliably
Session completed
root@kali2019:~# john --show ~/Cosmo
Gamma:system:1002:1002::/home/Gamma:/bin/sh
Omega:car:1003:1003::/home/Omega:/bin/sh
Condo:apple:1004:1004::/home/Condo:/bin/sh
root@kali2019:~# john --show
3 password hashes cracked, 4 left

```

## Brute-Force Attack:

```

root@kali2019:~# john ~/Cosmo
Using default input encoding: UTF-8
Loaded 7 password hashes with 7 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Remaining 4 password hashes with 4 different salts
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Wordlist
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 2 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 6 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 2 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 4 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 7 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.

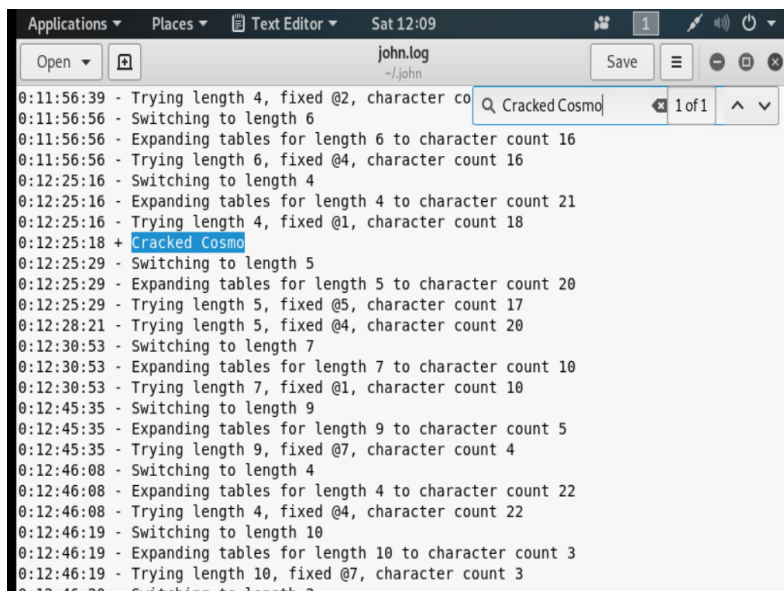
```

```

needed for performance.
Warning: Only 6 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 2 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 4 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 7 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Further messages of this type will be suppressed.
To see less of these warnings in the future, enable 'RelaxKPCWarningCheck'
in john.conf
Almost done: Processing the remaining buffered candidate passwords, if any
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
file (Cosmo)
atlanta:password h(Alpha)acked, 4 left
[1]+ Stopped john ~/Cosmo

```

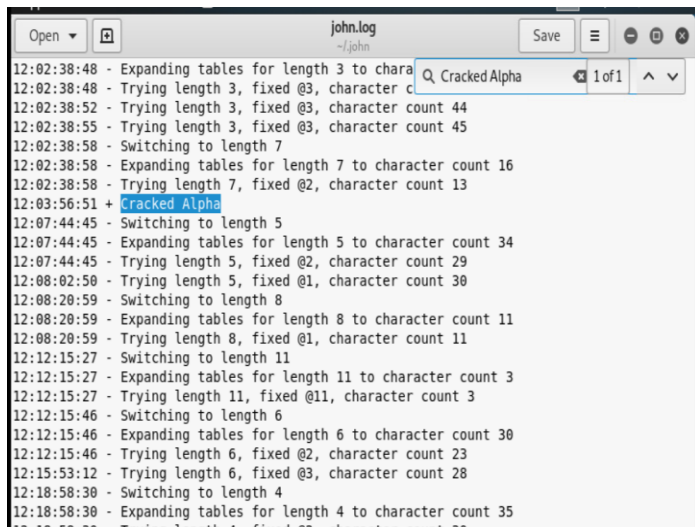
After running Brute-Force attack, it took 12 hours 25 minutes to crack the password of Cosmo, and took almost 12 days 4 hours to crack the password. But the password of Beta user didn't crack still as the password contains alpha numeric along with a special character, and the password length is long when compared to others. As per my understanding, the Brute-Force attack is a slow but efficient cracking method when compared to Dictionary Attack



```

Applications ▾ Places ▾ Text Editor ▾ Sat 12:09
john.log
~/.john
Open Save
0:11:56:39 - Trying length 4, fixed @2, character count 16
0:11:56:56 - Switching to length 6
0:11:56:56 - Expanding tables for length 6 to character count 16
0:11:56:56 - Trying length 6, fixed @4, character count 16
0:12:25:16 - Switching to length 4
0:12:25:16 - Expanding tables for length 4 to character count 21
0:12:25:16 - Trying length 4, fixed @1, character count 18
0:12:25:18 + Cracked Cosmo
0:12:25:29 - Switching to length 5
0:12:25:29 - Expanding tables for length 5 to character count 20
0:12:25:29 - Trying length 5, fixed @5, character count 17
0:12:28:21 - Trying length 5, fixed @4, character count 20
0:12:30:53 - Switching to length 7
0:12:30:53 - Expanding tables for length 7 to character count 10
0:12:30:53 - Trying length 7, fixed @1, character count 10
0:12:45:35 - Switching to length 9
0:12:45:35 - Expanding tables for length 9 to character count 5
0:12:45:35 - Trying length 9, fixed @7, character count 4
0:12:46:08 - Switching to length 4
0:12:46:08 - Expanding tables for length 4 to character count 22
0:12:46:08 - Trying length 4, fixed @4, character count 22
0:12:46:19 - Switching to length 10
0:12:46:19 - Expanding tables for length 10 to character count 3
0:12:46:19 - Trying length 10, fixed @7, character count 3
0:12:46:20 - Switching to length 2

```



```
john.log
~/john
Open Save
12:02:38:48 - Expanding tables for length 3 to character count 44
12:02:38:48 - Trying length 3, fixed @3, character count 44
12:02:38:52 - Trying length 3, fixed @3, character count 44
12:02:38:55 - Trying length 3, fixed @3, character count 45
12:02:38:58 - Switching to length 7
12:02:38:58 - Expanding tables for length 7 to character count 16
12:02:38:58 - Trying length 7, fixed @2, character count 13
12:03:56:51 + Cracked Alpha
12:07:44:45 - Switching to length 5
12:07:44:45 - Expanding tables for length 5 to character count 34
12:07:44:45 - Trying length 5, fixed @2, character count 29
12:08:02:50 - Trying length 5, fixed @1, character count 30
12:08:20:59 - Switching to length 8
12:08:20:59 - Expanding tables for length 8 to character count 11
12:08:20:59 - Trying length 8, fixed @1, character count 11
12:12:15:27 - Switching to length 11
12:12:15:27 - Expanding tables for length 11 to character count 3
12:12:15:27 - Trying length 11, fixed @11, character count 3
12:12:15:46 - Switching to length 6
12:12:15:46 - Expanding tables for length 6 to character count 30
12:12:15:46 - Trying length 6, fixed @2, character count 23
12:15:53:12 - Trying length 6, fixed @3, character count 28
12:18:58:30 - Switching to length 4
12:18:58:30 - Expanding tables for length 4 to character count 35
12:18:58:30 - Trying length 4, fixed @2, character count 30
```

## Learnings:

I have attempted using the same user credential list with Hashcat, seems John the ripper is more efficient when compared to Hashcat(done analysis privately). As per my observation, John the ripper was better in finding and cracking the passwords. I think John the Ripper can crack more character length passwords than Hashcat does.