NCL Password Cracking--Key

Getting Started

You can install a pre-built VMware VM from here. You just need to open the VM, not install it. The login credentials are kali, kali.

https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/

Then read this excellent blog from NCL.

https://cryptokait.com/2019/09/24/everything-you-need-to-know-about-password-cracking-for-the-national-cyber-league-games/

A Note about Hashcat

Hashcat, and the older John the Ripper, keep a list of hashes they have already cracked. If the hash is in the list, it will not appear in the output. This can be confusing if you are having problems and run the same hash file repeatedly. If you think some cracked hashes are missing, look in `.hashcat/hashcat.potfile` in your home directory, or perhaps where you ran hashcat from. If you want to start from scratch, just delete `.hashcat/hashcat.potfile`

A Crack to Start With

This is from the paragraph in the NCL Blog, "Using a Pre-Made Wordlist on Kali." Follow the procedure to unzip (actually Gnu unzip or gunzip) the password list from the rockyou.com breach (I moved rockyou.txt to my home directory instead of Downloads.) Then run the hashcat line, hashcat -m 0 {means the hashes are MD5} -a 0 {attack will be wordlist only} -o outputfile {where the output goes} hashlist pwlist

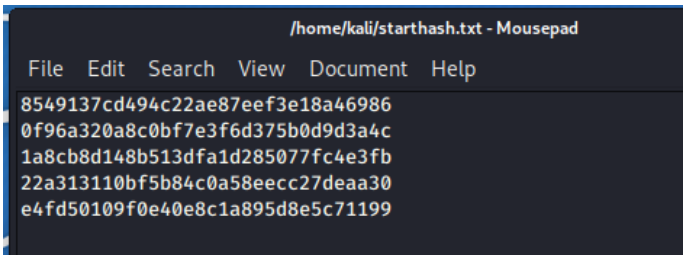
Here are the hashes.

```
8549137cd494c22ae87eef3e18a46986
0f96a320a8c0bf7e3f6d375b0d9d3a4c
1a8cb8d148b513dfa1d285077fc4e3fb
22a313110bf5b84c0a58eecc27deaa30
e4fd50109f0e40e8c1a895d8e5c71199
```

These are easy and should crack in under a minute.

Solution

Save the hashes in a file on Kali.



Remember to unzip rockyou.gz and put it in your home directory.

```
kali@kali:/usr/share/wordlists$ gunzip rockyou.gz
kali@kali:/usr/share/wordlists$ mv rockyou.txt ~
mv: cannot move 'rockyou.txt' to '/home/kali/rockyou.txt': Permission
denied
kali@kali:/usr/share/wordlists$ sudo !!
sudo mv rockyou.txt ~
```

Crack the hashes

```
hashcat -m 0 -a 0 -o startout.txt starthash.txt rockyou.txt --force
```

```
kali@kali:~$ hashcat -m 0 -a 0 -o startout.txt starthash.txt rockyou.txt --force
hashcat (v5.1.0) starting...

OpenCL Platform #1: The pocl project
====================================
* Device #1: pthread-Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2048/5911 MB allocatable, 2MCU

Hashes: 5 digests; 5 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash
```

<snip>

```
Session..........: hashcat
Status...........: Cracked
Hash.Type........: MD5
Hash.Target......: starthash.txt
Time.Started.....: Wed Mar 18 09:38:12 2020 (1 sec)
Time.Estimated...: Wed Mar 18 09:38:13 2020 (0 secs)
Guess.Base.......: File (rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    66539 H/s (0.38ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered........: 5/5 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 8192/14344385 (0.06%)
Rejected.........: 0/8192 (0.00%)
Restore.Point....: 6144/14344385 (0.04%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1....: horoscope -> whitetiger

Started: Wed Mar 18 09:38:10 2020
Stopped: Wed Mar 18 09:38:14 2020
kali@kali:~$ cat startout.txt
0f96a320a8c0bf7e3f6d375b0d9d3a4c:hustler
1a8cb8d148b513dfa1d285077fc4e3fb:loved1
22a313110bf5b84c0a58eecc27deaa30:trivium
8549137cd494c22ae87eef3e18a46986:chasity
e4fd50109f0e40e8c1a895d8e5c71199:poppy123
kali@kali:~$
```

A Simple Custom Wordlist

In this one (Creating an Enumerated Wordlist on Kali, from the blog) you are told the passwords are of the form SKY-KAIT-####, where # is one digit. She uses crunch in Kali to generate a list. So, generate a list and crack the hashes. This one should go quickly if your list is good. Remember to replace rockyou.txt from you hashcat command with the name of the new list.

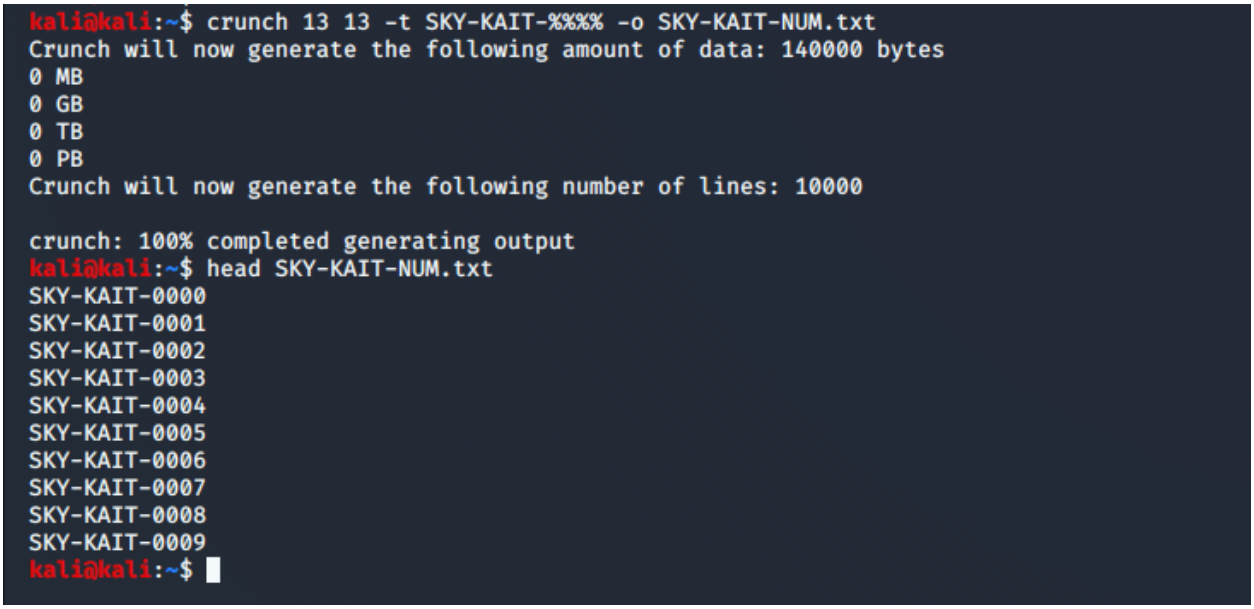
```
c38d29e8899455c85ee03d11abbd262b
```

```
ff8f9efad5c9f106ac39e5290d810c91
425206344bd204933a38236b715c498f
ab37c335e51b2855cb5a11ca89041733
82dcf30f8c7c8d4f23961f7e0c1d3cee
```

Solution

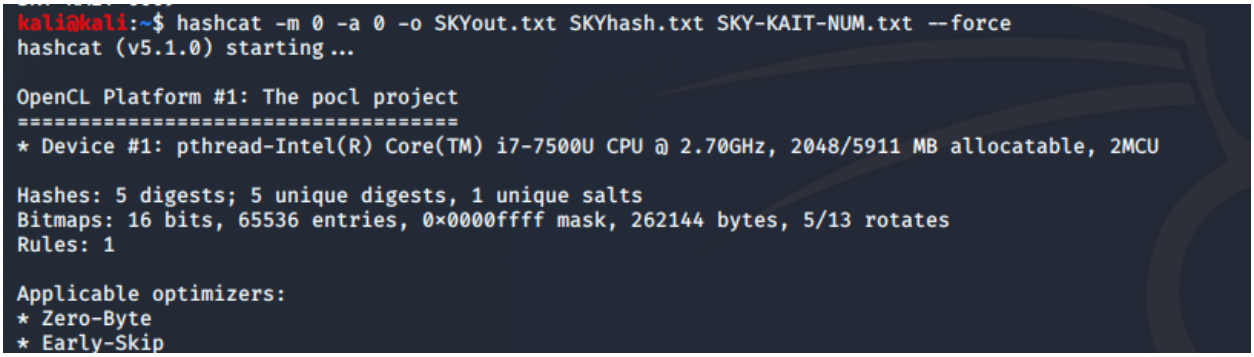
Make the list.

```
crunch 13 13 -t SKY-KAIT-%%%% -o SKY-KAIT-NUM.txt
```



Crack

```
hashcat -m 0 -a 0 -o SKYout.txt SKYhash.txt SKY-KAIT-NUM.txt –force
```



<snip>

```
Session..........: hashcat
Status...........: Cracked
Hash.Type........: MD5
Hash.Target......: SKYhash.txt
Time.Started.....: Wed Mar 18 09:45:28 2020 (0 secs)
Time.Estimated...: Wed Mar 18 09:45:28 2020 (0 secs)
Guess.Base.......: File (SKY-KAIT-NUM.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    50000 H/s (1.35ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered........: 5/5 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 10000/10000 (100.00%)
Rejected.........: 0/10000 (0.00%)
Restore.Point....: 8192/10000 (81.92%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1....: SKY-KAIT-8192 → SKY-KAIT-9999

Started: Wed Mar 18 09:45:26 2020
Stopped: Wed Mar 18 09:45:29 2020
kali@kali:~$ cat SKYout.txt
425206344bd204933a38236b715c498f:SKY-KAIT-3572
ab37c335e51b2855cb5a11ca89041733:SKY-KAIT-4130
82dcf30f8c7c8d4f23961f7e0c1d3cee:SKY-KAIT-5892
c38d29e8899455c85ee03d11abbd262b:SKY-KAIT-6823
ff8f9efad5c9f106ac39e5290d810c91:SKY-KAIT-9965
425206344bd204933a38236b715c498f:SKY-KAIT-3572
ab37c335e51b2855cb5a11ca89041733:SKY-KAIT-4130
82dcf30f8c7c8d4f23961f7e0c1d3cee:SKY-KAIT-5892
c38d29e8899455c85ee03d11abbd262b:SKY-KAIT-6823
ff8f9efad5c9f106ac39e5290d810c91:SKY-KAIT-9965
kali@kali:~$
```

A Pokémon Wordlist

In the blog, Kait found a Wikipedia list of all the Pokémon characters, pasted them into Excel, and then edited the list. I'll use a slightly different method for this. The site https://pokemondb.net/pokedex/national has a good list of Pokémon characters. We can scrape them off the site with a program called cewl (by @digininja) that is included in Kali. The syntax for what we want is simple.

```
cewl -d 0 -w list.txt "https://pokemondb.net/pokedex/national"
```

The "`-d 0`" is critical. It is the depth you want to spider the web site. If the words you want are in the top level of the URL you give, set the depth to zero and cewl will be fast. On a big site, any other choice (even just 1) cat be extremely slow. The "`-w list.txt`" just tells where to put the output.

If you look at the list that is generated by cewl, you will find a lot of extra words besides Pokémon names. Our list is small and hashcat is fast, so it is not worth the time to edit it.

If you run this like we did before, you will get zero hashes cracked.

```
hashcat -m 0 -a 0 -o pokeout.txt pokehashes.txt list.txt
```

Ugh. If we look at the blog, we'll see that one of the hashes was for Charizard6. It appears they are appending the Pokémon character's number to the name. Rather than spend the time to edit list.txt by adding the number, let's just try all possible numbers. Use the hybrid attack (dictionary plus mask) at the end of this site. https://laconicwolf.com/2018/09/29/hashcat-tutorial-the-basics-of-cracking-passwords-with-hashcat/ A look at the list of Pokemon characters show that the highest number is 890, so we'll use three digits. The mask for three decimal digits is ?d?d?d . If we use that it will try all numbers 000 to 999. We want the one and two digits without the leading zeros, 0 to 999. To do that, we use the `--increment` flag. Hybrid mode is -a 6.

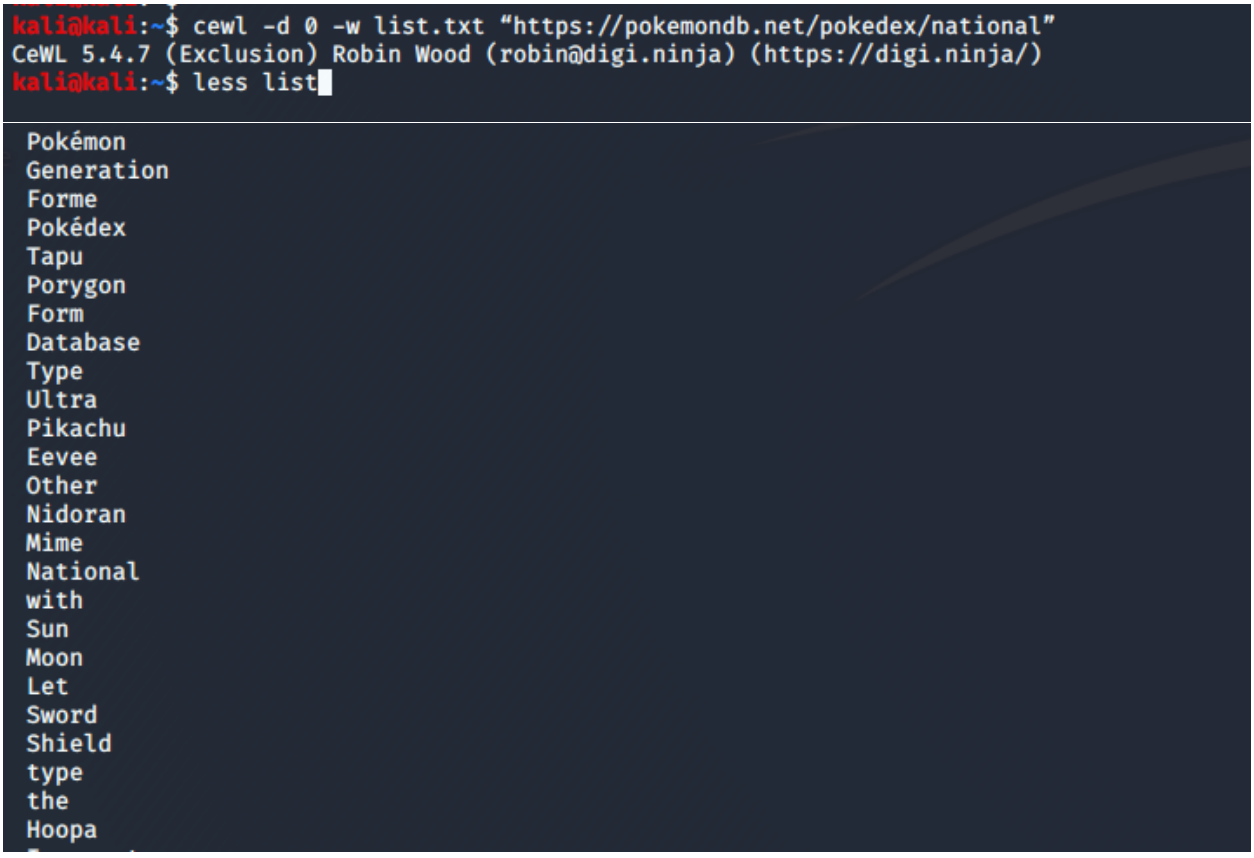
```
hashcat -m 0 -o pokeout.txt -a 6 pokehashes.txt list.txt ?d?d?d --
increment --force
```

```
3546576a03c2c8229175eede8c02f89
a19d7a52bff83b0e4012d2c766e2f731
5a31b6b31f92c8f797505ca26af4b9de
857875c031fce47b2d40be0ce3ffd0bf
dc6054fbe36c8a2bd49b1d05b3b872ee
```

Solution

Grab the list.

```
cewl -d 0 -w list.txt "https://pokemondb.net/pokedex/national
```



As you can see, we have a mix of character names and other words.

Crack the hashes, adding numbers at the end and the --increment switch.

```
hashcat -m 0 -o pokeout.txt -a 6 pokehashes.txt list ?d?d?d --
increment --force
```

```
kali@kali:~$ hashcat -m 0 -o pokeout.txt -a 6 pokehashes.txt list ?d?d?d --increment --force
hashcat (v5.1.0) starting...

OpenCL Platform #1: The pocl project
====================================
* Device #1: pthread-Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2048/5911 MB allocatable, 2MCU

Hashfile 'pokehashes.txt' on line 1 (3546576a03c2c8229175eede8c02f89): Token length exception
Hashes: 4 digests; 4 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Applicable optimizers:
* Zero-Byte
* Early-Skip
```

<snip>

```
Session..........: hashcat
Status...........: Cracked
Hash.Type........: MD5
Hash.Target......: pokehashes.txt
Time.Started.....: Wed Mar 18 09:55:00 2020 (1 sec)
Time.Estimated...: Wed Mar 18 09:55:01 2020 (0 secs)
Guess.Base.......: File (list), Left Side
Guess.Mod........: Mask (?d?d?d) [3], Right Side
Guess.Queue.Base.: 1/1 (100.00%)
Guess.Queue.Mod..: 3/3 (100.00%)
Speed.#1.........:  8916.5 kH/s (13.98ms) @ Accel:512 Loops:125 Thr:1 Vec:8
Recovered........: 4/4 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 896000/1064000 (84.21%)
Rejected.........: 0/896000 (0.00%)
Restore.Point....: 0/1064 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:750-875 Iteration:0-125
Candidates.#1....: sprite138 → page958

Started: Wed Mar 18 09:54:58 2020
Stopped: Wed Mar 18 09:55:02 2020
kali@kali:~$ cat pokeout.txt
857875c031fce47b2d40be0ce3ffd0bf:Milotic350
dc6054fbe36c8a2bd49b1d05b3b872ee:Absol359
a19d7a52bff83b0e4012d2c766e2f731:Beedrill15
5a31b6b31f92c8f797505ca26af4b9de:Weezing110
857875c031fce47b2d40be0ce3ffd0bf:Milotic350
dc6054fbe36c8a2bd49b1d05b3b872ee:Absol359
a19d7a52bff83b0e4012d2c766e2f731:Beedrill15
5a31b6b31f92c8f797505ca26af4b9de:Weezing110
857875c031fce47b2d40be0ce3ffd0bf:Milotic350
dc6054fbe36c8a2bd49b1d05b3b872ee:Absol359
kali@kali:~$
```

We had some repeats, probably due to increment, but we'll take it.

It looks like the first hash, 3546576a03c2c8229175eede8c02f89, is missing a character. I tried appending and prepending [0-9a-f] but didn't find the missing character. I guess I could hash all the

entries in my list, with 0-890 appended, and see if any are close to the hash missing the digit.

```
Hashfile 'pokehashes.txt' on line 1 (3546576a03c2c8229175eede8c02f89): Token length exception
Hashes: 4 digests; 4 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13 rotates
```