Password Attacks

this room is an introduction to the types and techniques used in password attacks. we will discuss the way to get and generate custom password lists.

TOPICS:

password profiling

password attacks techniques

online password attacks

Password Attacking Techniques

various tools used for password cracking, including **Hashcat** and **John the Ripper**.

Password guessing is a technique used to target online protocols and services.

Password cracking is a technique performed locally or on systems controlled by the attacker.

Task 3

Password Profiling #1 - Default, Weak, Leaked, Combined , and Username Wordlists

Default Passwords

Before performing password attacks, it is worth trying a couple of default passwords against the targeted service

try out admin:admin, admin:123456, etc.

Here are some website lists that provide default passwords for various products.

- **https://cirt.net/passwords**
- **https://default-password.info/**
- **https://datarecovery.com/rd/default-passwords/**
- **https://wiki.skullsecurity.org/index.php?title=Passwords - This includes the most well-known collections of passwords.**
- **SecLists - A huge collection of all kinds of lists, not only for password cracking.**

Leaked Passwords: SecLists/Passwords/Leaked-Databases

Combined wordlists:

```
cat file1.txt file2.txt file3.txt > combined_list.txt
```

To clean up the generated combined list to remove duplicated words, we can use sort and uniq as follows:

```
sort combined_list.txt | uniq -u > cleaned_combined_list.txt
```

Customized Wordlists

Tools such as **Cewl** can be used to effectively crawl a website and extract strings or keywords. **Cewl** is a powerful tool to generate a wordlist specific to a given company or target.

```
$ cewl -w list.txt -d 5 -m 5 http://thm.labs
```

- w will write the contents to a file. In this case, list.txt.
- m 5 gathers strings (words) that are 5 characters or more
- d 5 is the depth level of web crawling/spidering (default 2)
- http://thm.labs is the URL that will be used

Username Wordlists

Gathering employees' names in the enumeration stage is essential. We can generate username lists from the target's website. For the following example, we'll assume we have a **{first name} {last name} (ex: John Smith)** and a method of generating usernames.

- **{first name}:** john
- **{last name}:** smith
- **{first name}{last name}: johnsmith**
- **{last name}{first name}: smithjohn**
- first letter of the **{first name}{last name}: jsmith**
- first letter of the **{last name}{first name}: sjohn**
- first letter of the **{first name}.{last name}: j.smith**
- first letter of the **{first name}-{last name}: j-smith**
- and so on

Thankfully, there is a tool username_generator that could help create a list with most of the possible combinations if we have a first name and last name.

1. **What are the default login credentials (in the format of `username:password`) for a Juniper Networks ISG 2000 device?**

ans: **netscreen:netscreen**

Taks 4

Keyspace Technique

Another way of preparing a wordlist is by using the key-space technique. In this technique, we specify a range of characters, numbers, and symbols in our wordlist. **crunch** is one of many powerful tools for creating an offline wordlist. With **crunch**, we can specify numerous options, including min, max, and options as follows

$ crunch 2 3 all_the_words_for_password -o password.txt

-o is for creating the password

2 3 is for the length of the password

crunch 8 8 0123456789abcdefABCDEF -o crunch.txt the file generated is 459 GB and contains 54875873536 words.

crunch also lets us specify a character set using the -t option to combine words of our choice.

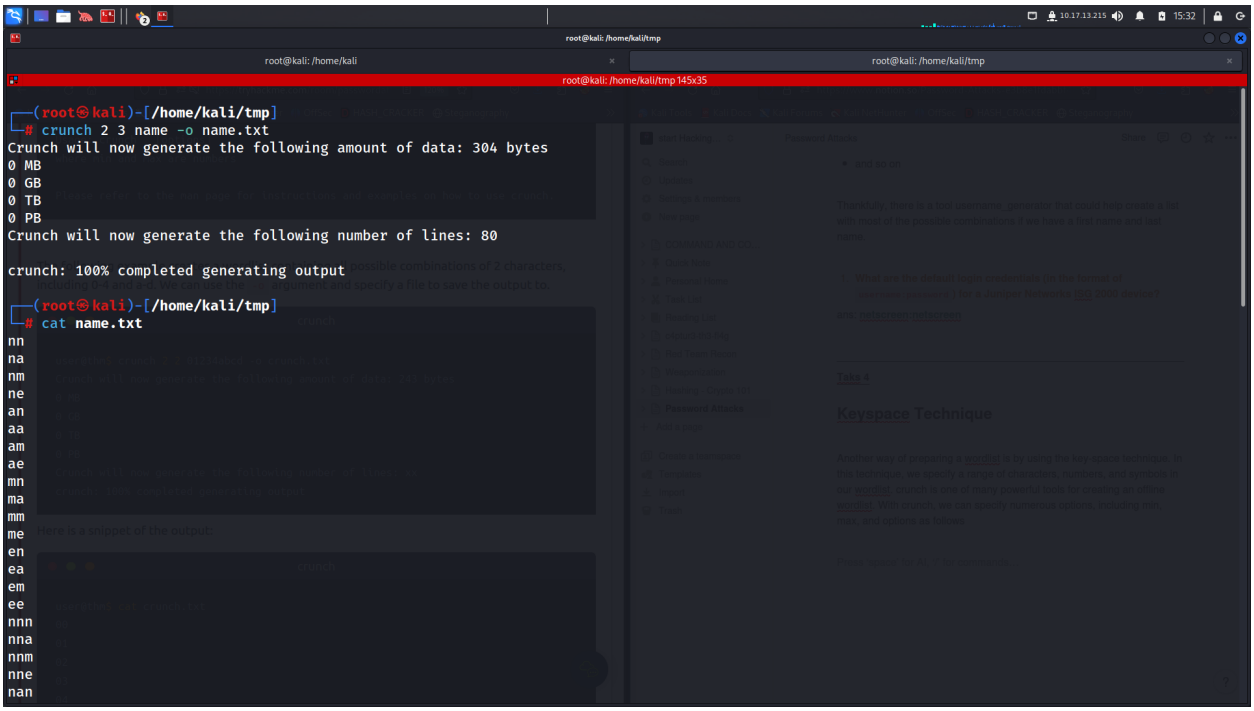
@ - lower case alpha characters

, - upper case alpha characters

% - numeric characters

^ - special characters including space

```
$  crunch 6 6 -t pass%%
```



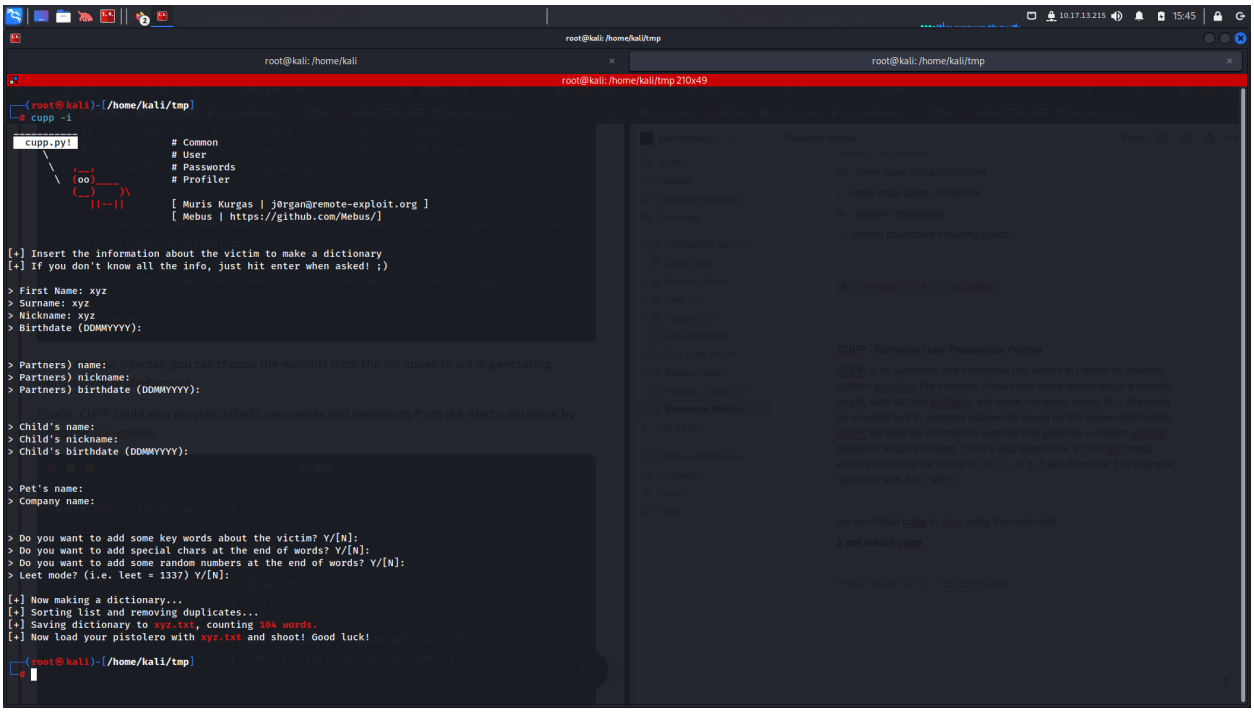
CUPP - Common User Passwords Profiler

CUPP is an automatic and interactive tool written in Python for creating custom wordlists. For instance, if you know some details about a specific target, such as their birthdate, pet name, company name, etc., this could be a helpful tool to generate passwords based on this known information. CUPP will take the information supplied and generate a custom wordlist based on what's provided. There's also support for a 1337/leet mode, which substitutes the letters a, i,e, t, o, s, g, z with numbers. For example, replace a with 4 or i with 1.

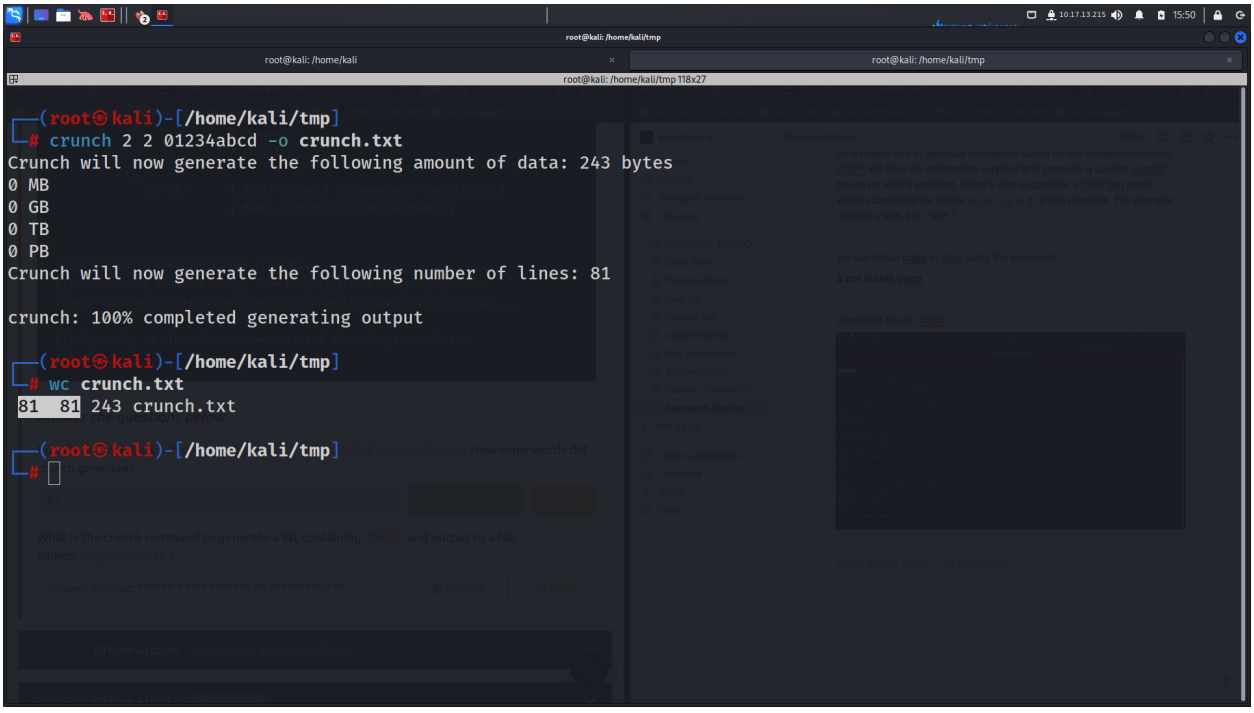
we can install **cupp** in linux using the command

$ apt install cupp

command to use: **cupp -i**



1. **Run the following crunch command:crunch 2 2 01234abcd -o crunch.txt. How many words did crunch generate?**



ans: **81**

2. **What is the crunch command to generate a list containing THM@% and output to a file named tryhackme.txt?**

ans: **crunch 5 5 -t “THM^^” -o tryhackme.txt**

Task 5

Offline Attacks - Dictionary and Brute-Force

Dictionary attack

A dictonary attack is a technique used to guess passwords bu using well-known words or phrases. the dictionary attack relies entirely on pre-gathered wordlists that were previously generated or found, it is important to choose or create the best candidate wordlist for your target in order to succeed in this attack.

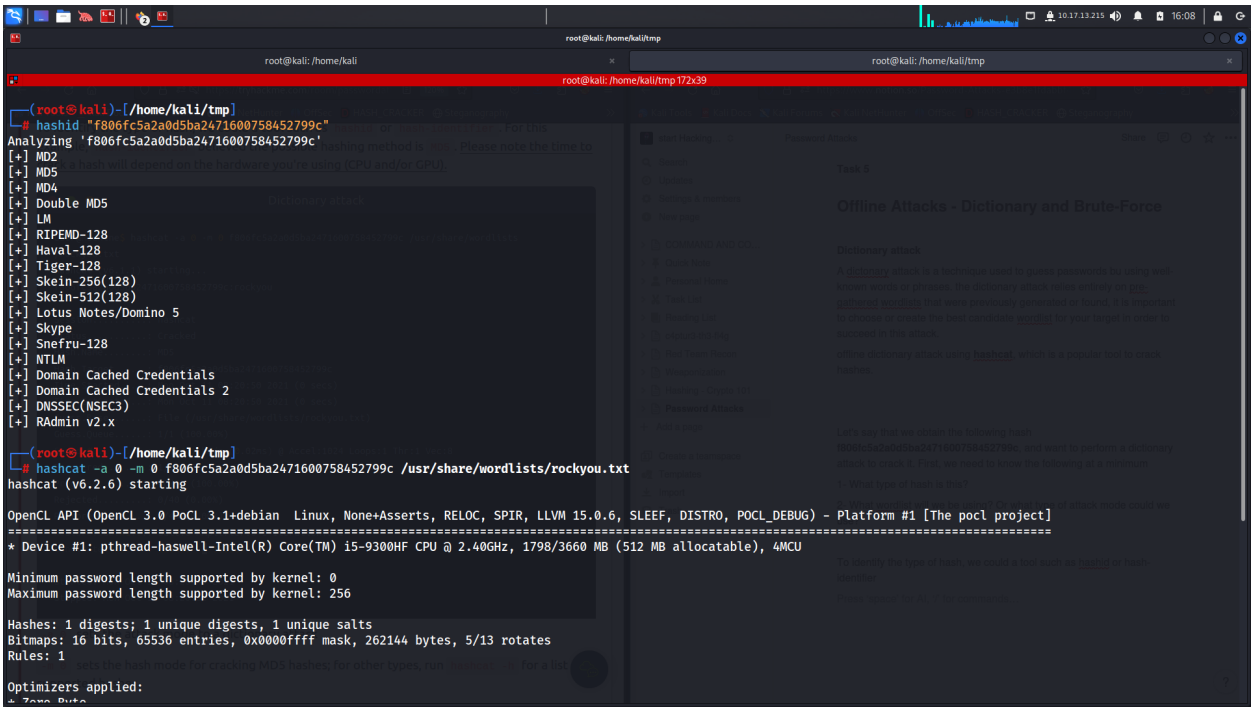
offline dictionary attack using **hashcat**, which is a popular tool to crack hashes.

Let's say that we obtain the following hash **f806fc5a2a0d5ba2471600758452799c**, and want to perform a dictionary attack to crack it. First, we need to know the following at a minimum

1- What type of hash is this?

2- What wordlist will we be using? Or what type of attack mode could we use?

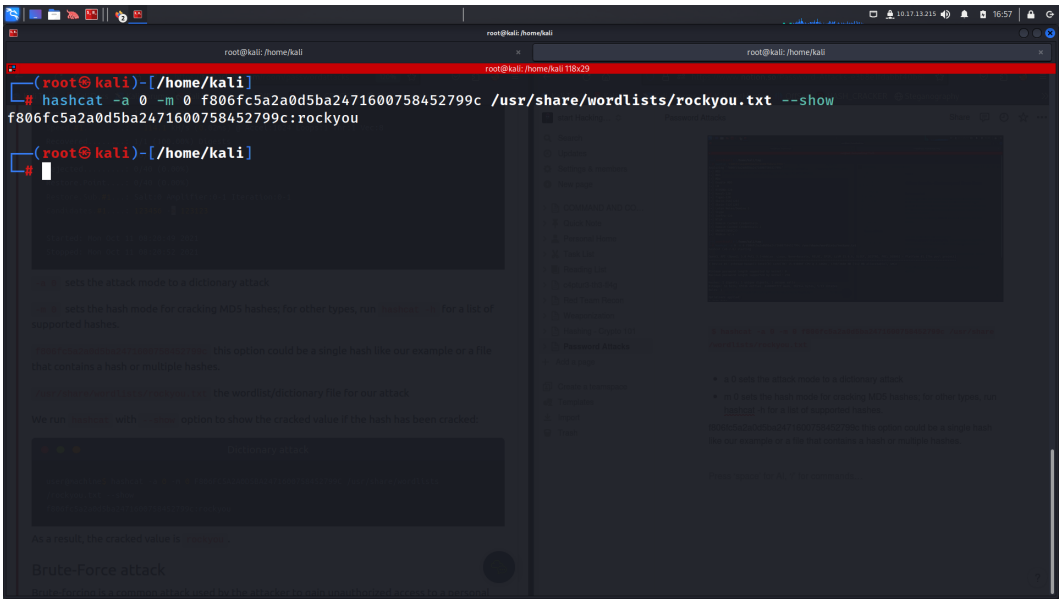
To identify the type of hash, we could a tool such as hashid or hash-identifier




```
$ hashcat -a 0 -m 0 f806fc5a2a0d5ba2471600758452799c /usr/share/wordlists/rockyou.txt
```

- a 0 sets the attack mode to a dictionary attack
- m 0 sets the hash mode for cracking MD5 hashes; for other types, run hashcat -h for a list of supported hashes.

f806fc5a2a0d5ba2471600758452799c this option could be a single hash like our example or a file that contains a hash or multiple hashes.




```
$ hashcat -a 0 -m 0 F806FC5A2A0D5BA2471600758452799C /usr/share/wordlists/rockyou.txt --sh
owf806fc5a2a0d5ba2471600758452799c:rockyou
```

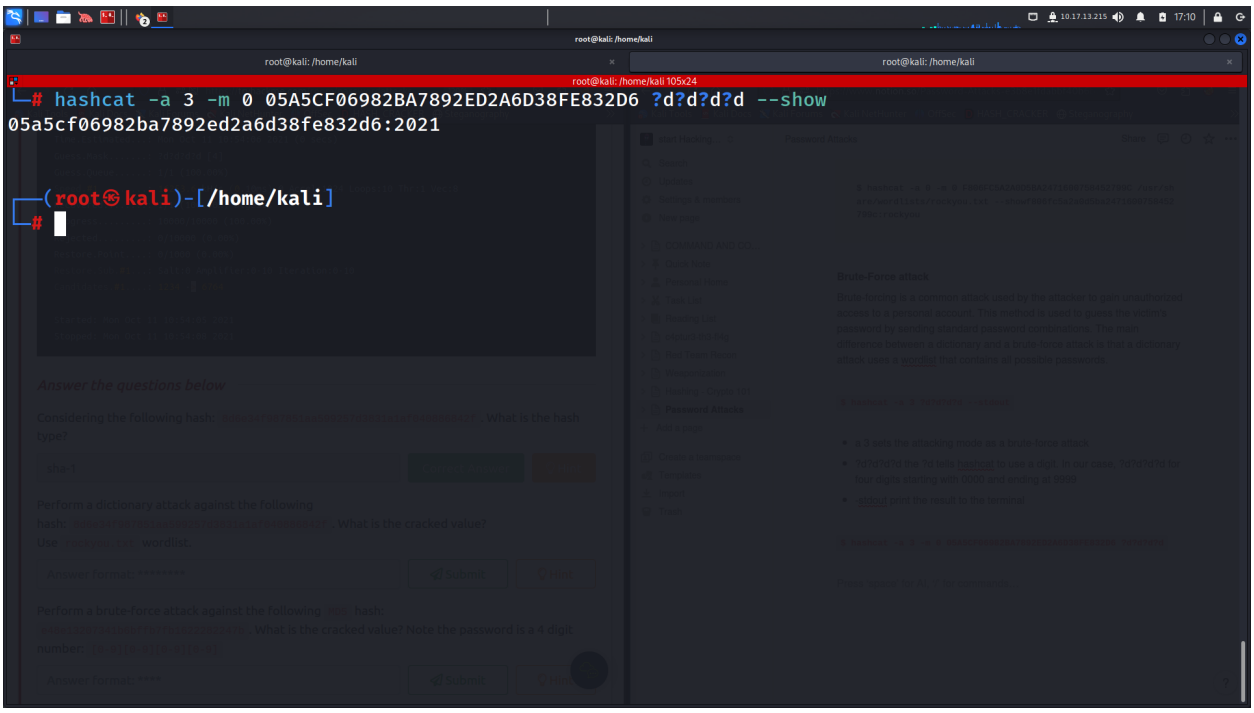
Brute-Force attack

Brute-forcing is a common attack used by the attacker to gain unauthorized access to a personal account. This method is used to guess the victim's password by sending standard password combinations. The main difference between a dictionary and a brute-force attack is that a dictionary attack uses a wordlist that contains all possible passwords.

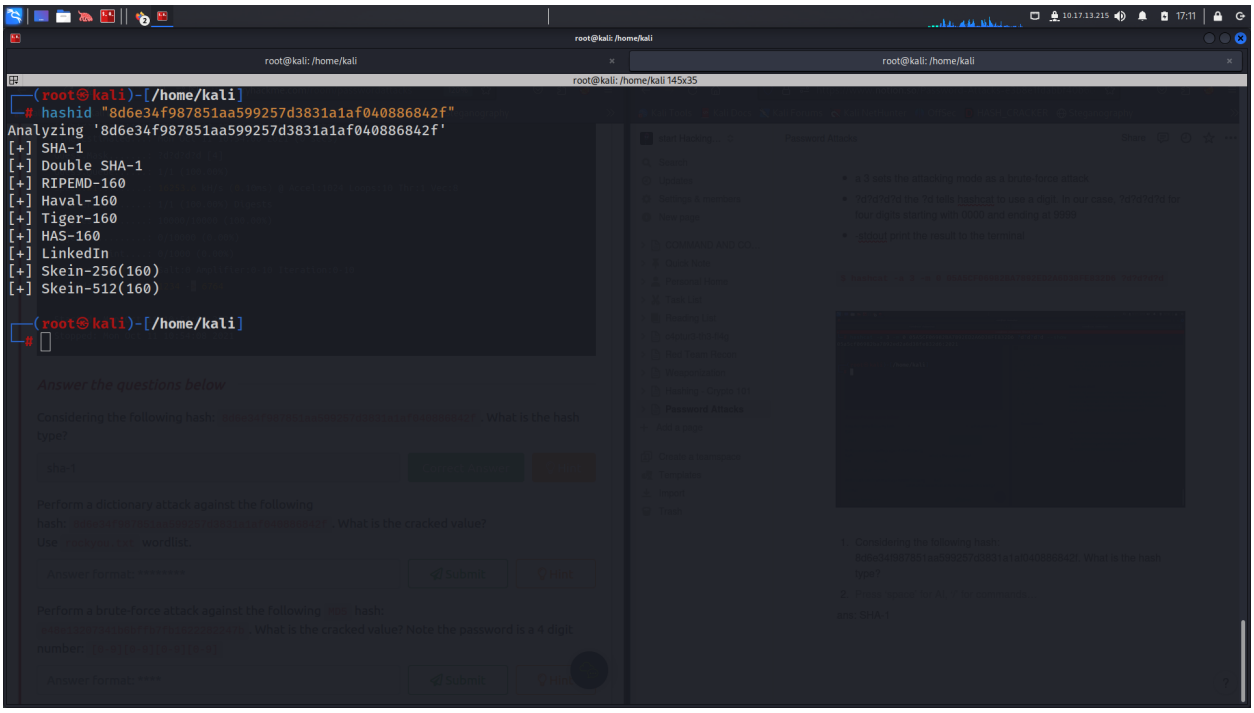
`$ hashcat -a 3 ?d?d?d?d --stdout`

- a 3 sets the attacking mode as a brute-force attack
- ?d?d?d?d the ?d tells hashcat to use a digit. In our case, ?d?d?d?d for four digits starting with 0000 and ending at 9999
- -stdout print the result to the terminal

```
$ hashcat -a 3 -m 0 05A5CF06982BA7892ED2A6D38FE832D6 ?d?d?d?d
```

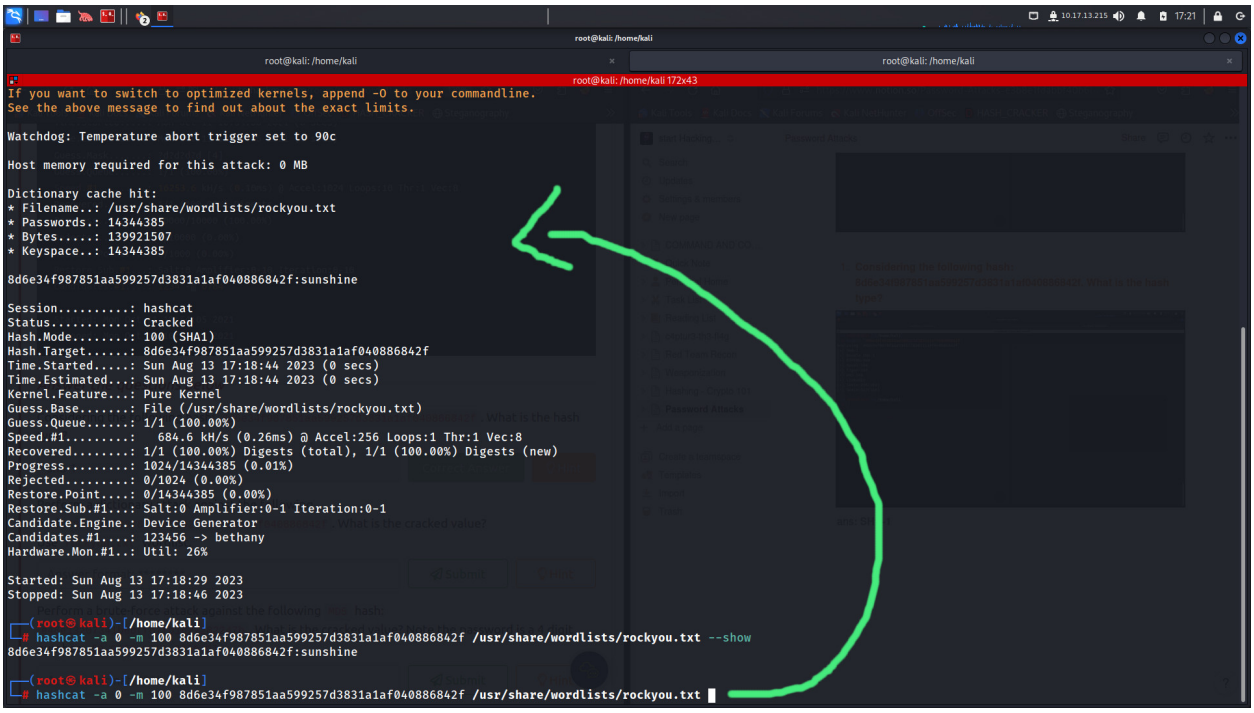


1. **Considering the following hash: 8d6e34f987851aa599257d3831a1af040886842f. What is the hash type?**



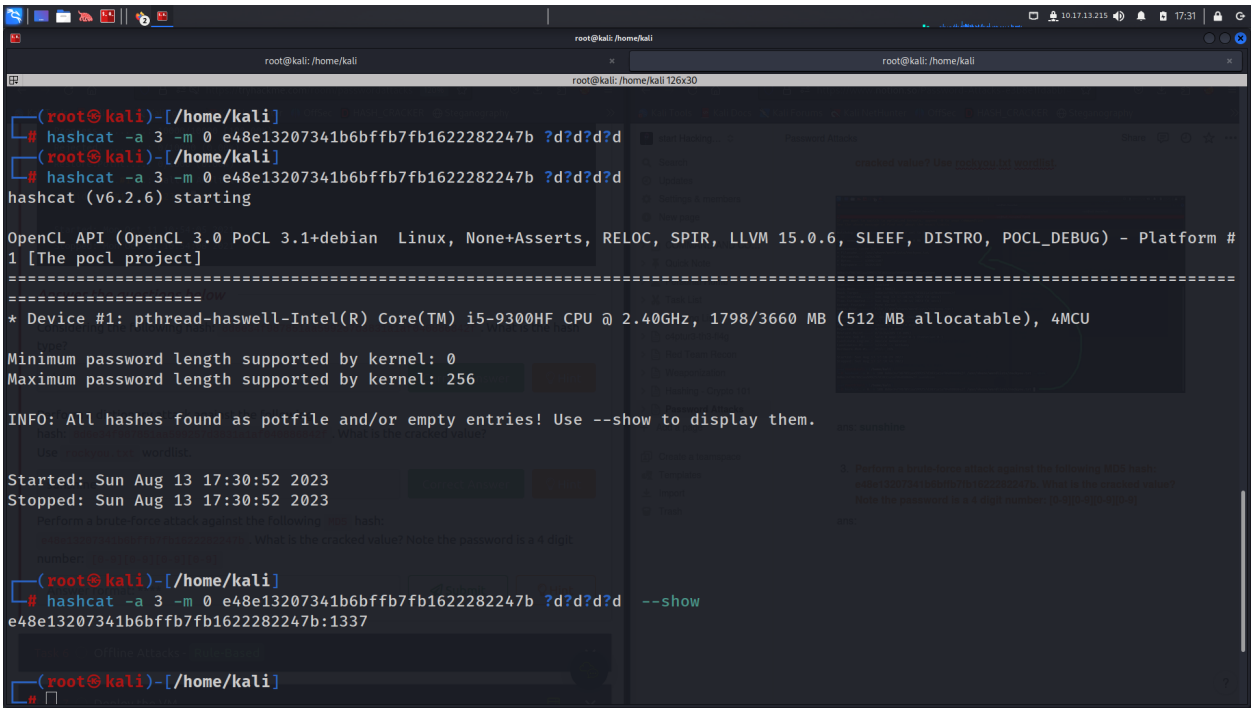
ans: **SHA-1**

2. **Perform a dictionary attack against the following hash: 8d6e34f987851aa599257d3831a1af040886842f. What is the cracked value? Use rockyou.txt wordlist.**



ans: **sunshine**

3. **Perform a brute-force attack against the following MD5 hash: e48e13207341b6bffb7fb1622282247b. What is the cracked value? Note the password is a 4 digit number: [0-9][0-9][0-9][0-9]**



ans: **1337**

Task 6

Offline Attacks - Rule-Based

Rule-Based attacks

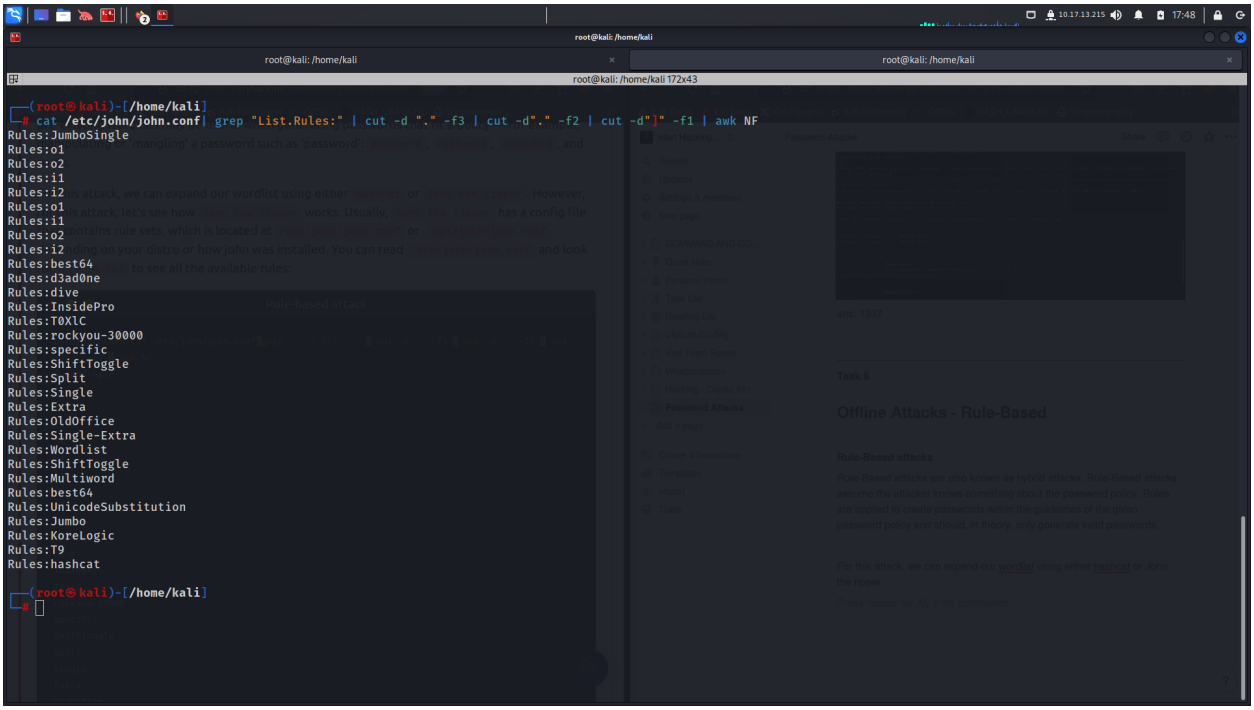
Rule-Based attacks are also known as hybrid attacks. Rule-Based attacks assume the attacker knows something about the password policy. Rules are applied to create passwords within the guidelines of the given password policy and should, in theory, only generate valid passwords.

For this attack, we can expand our wordlist using either hashcat or John the ripper

let's see how John the ripper works. Usually, John the ripper has a config file that contains rule sets, which is located at /etc/john/john.conf or /opt/john/john.conf

depending on your distro or how john was installed. You can read /etc/john/john.conf and look for List.Rules to see all the available rules:

```
$ john --wordlist=/tmp/single-password-list.txt --rules=best64 --stdout | wc -l
```



- -wordlist= to specify the wordlist or dictionary file.
- -rules to specify which rule or rules to use.
- -stdout to print the output to the terminal.

|wc -l to count how many lines John produced.

Custom Rules

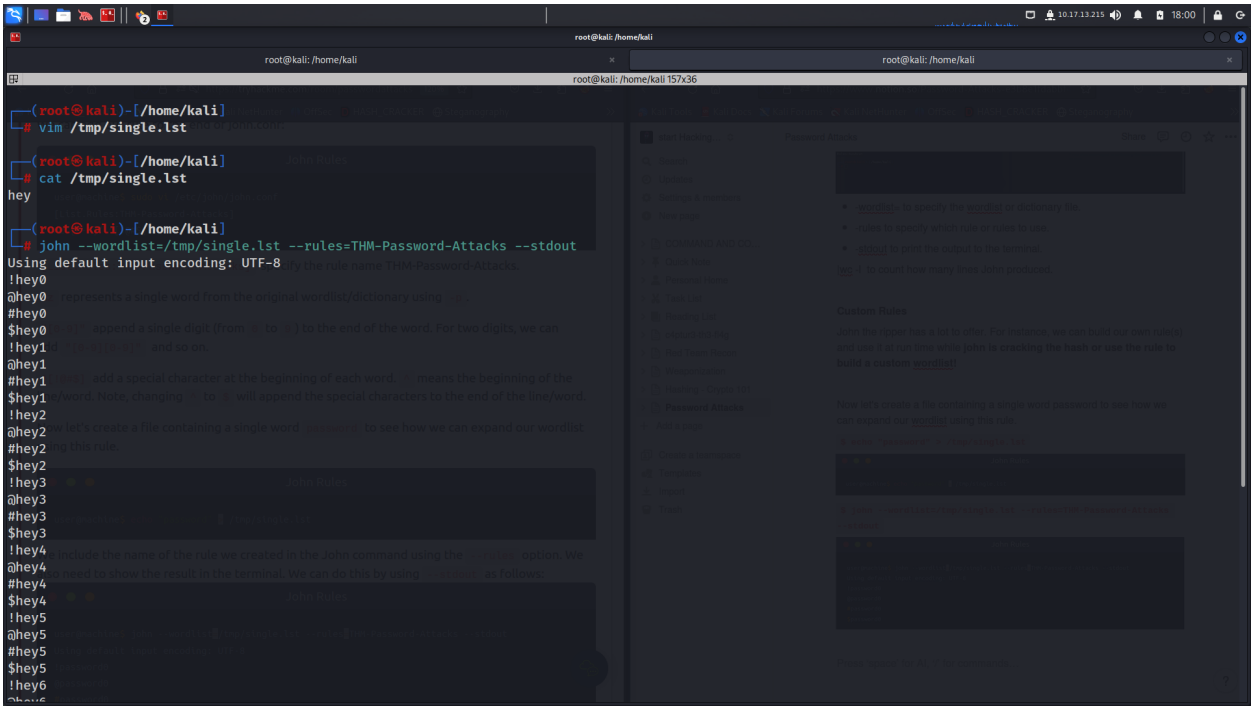
John the ripper has a lot to offer. For instance, we can build our own rule(s) and use it at run time while **john is cracking the hash or use the rule to build a custom wordlist!**

Now let's create a file containing a single word password to see how we can expand our wordlist using this rule.

```
$ echo "password" > /tmp/single.lst
```


```
$ john --wordlist=/tmp/single.lst --rules=THM-Password-Attacks --stdout
```

in my case:



1. **What syntax would you use to create a rule to produce the following: "S[Word]NN where N is Number and S is a symbol of !@?**

ans: **Az”[0–9][0–9]” ^[!@]**

Task 8

Online password attacks involve guessing passwords for networked services that use a username and password authentication scheme, including services such as HTTP, SSH, VNC, FTP, SNMP, POP3, etc. This section showcases using hydra which is a common tool used in attacking logins for various network services.

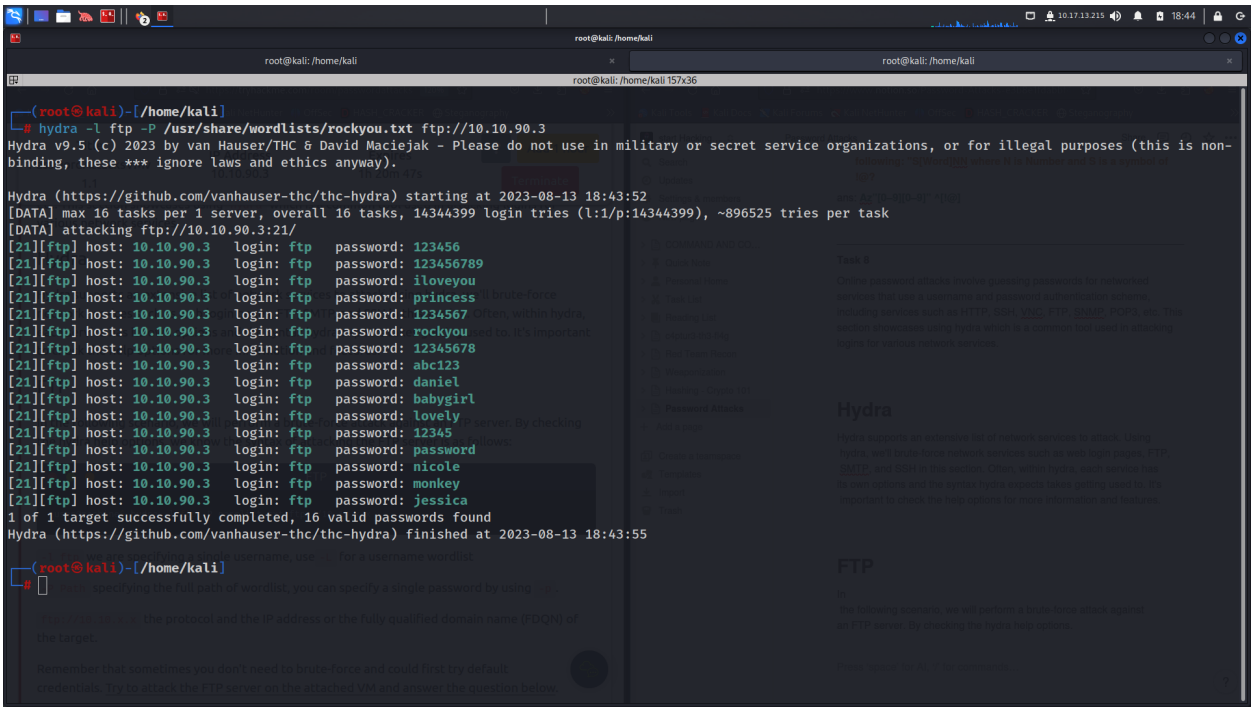
Hydra

Hydra supports an extensive list of network services to attack. Using hydra, we'll brute-force network services such as web login pages, FTP, SMTP, and SSH in this section. Often, within hydra, each service has its own options and the syntax hydra expects takes getting used to. It's important to check the help options for more information and features.

FTP

In the following scenario, we will perform a brute-force attack against an FTP server. By checking the hydra help options.

```
$ hydra -l ftp -P passlist.txt ftp://10.10.x.x
```



SMTP

Similar to FTP servers, we can also brute-force SMTP servers using hydra.

```
$ hydra -l email@company.xyz -P /path/to/wordlist.txt smtp://10.10.x.x -v
```

SSH

SSH brute-forcing can be common if your server is accessible to the Internet. Hydra supports many protocols, including SSH.

```
$ hydra -L users.lst -P /path/to/wordlist.txt ssh://10.10.x.x -v
```

HTTP login pages

To brute-force HTTP login pages we need to understand what you are burte-forcing. Using hydra, it is important to specify the type of HTTP request, whether **GET** or **POST**. checking hydra options: hydra **http-get-form -U,** we can see that hydra has the following syntax for the **http-get-form** option:

<url>:<form parameters>:<condition string>[:<optional>[:<optional>]

```
$ hydra -l admin -P 500-worst-passwords.txt 10.10.x.x http-get-form "/login-get/index.php:username=^USER^&password=^PASS^:S=logout.php" -f
```

-l admin we are specifying a single username, use **-L** for a username wordlist

-P Path specifying the full path of wordlist, you can specify a single passowrd by using -p

10.10.*.* the IP address of the full qualified domain name (FQDN) of the target.

http-get-form the type of HTTP request, which can be either **http-get-form** or **http-post-form**

next we specify the URL, path, and conditions that are split using **:**

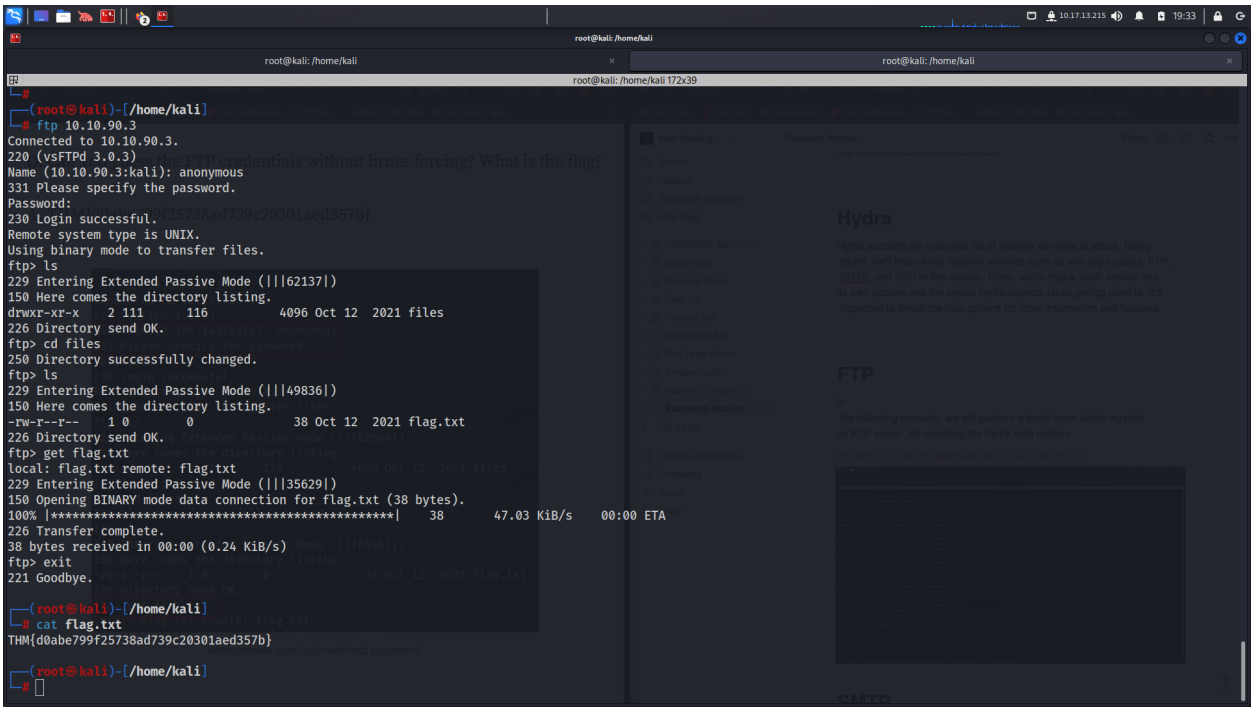
login-get/index.php the path of the login page on the target websrver

username=^USER^&password=^PASS^ the perameters to brute-force, we inject ^USER^ to brute force usernames and ^PASS^ for passwords from the specified dictionary.

The following section is important to eliminate false positive by specifying the failed condition with F=

1. **Can you guess the FTP credentials without brute-forcing? What is the flag?**

$ ftp 10.10.*.*



ans: **THM{d0abe799f25738ad739c20301aed357b}**

2. **In this question, you need to generate a rule-based dictionary from the wordlist clinic.lst in the previous task. email: pittman@clinic.thmredteam.com against 10.10.*.*:465 (SMTPS).What is the password? Note that the password format is as follows: [symbol][dictionary word][0-9][0-9].**

ans: **!multidisciplinary00**

3. **Perform a brute-forcing attack against the phillips account for the login page at http://10.10.90.3/login-get using hydra? What is the flag?**

ans: **THM{33c5d4954da881814420f3ba39772644}**

4. **Perform a rule-based password attack to gain access to the burgess account. Find the flag at the following website: http://10.10.*.*/login-post/. What is the flag? Note: use the clinic.lst dictionary in generating and expanding the wordlist!**

ans: **THM{33c5d4954da881814420f3ba39772644}**

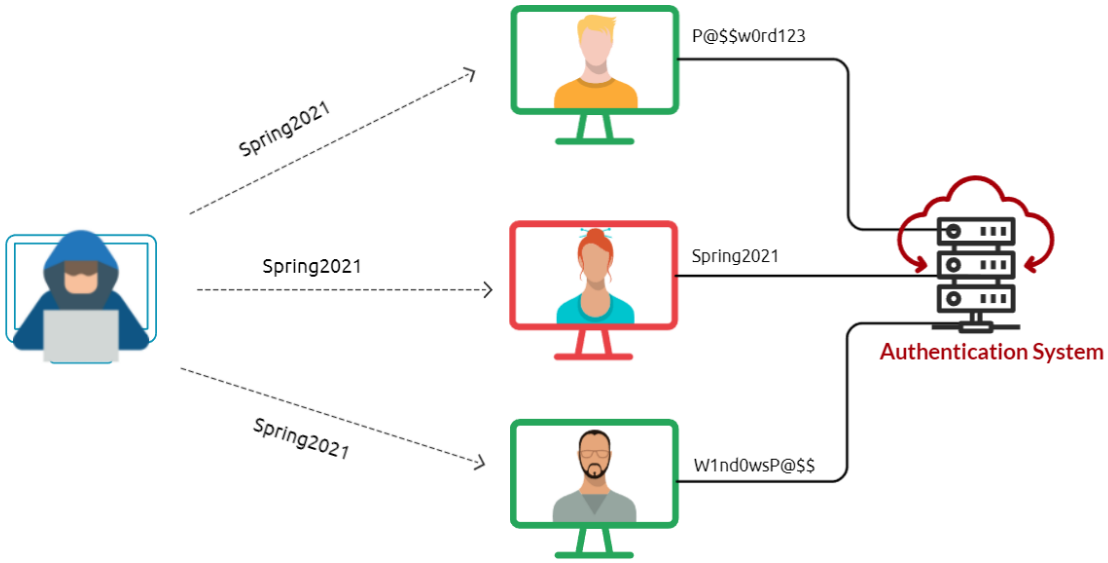
Task 9

Password spray attack

Password Spraying is an effective technique used to identify valid credentials. Nowadays, password spraying is considered one of the common password attacks for discovering weak passwords.

This technique can be used against various online services and authentication systems, such as SSH, SMB, RDP, SMTP, Outlook Web Application, etc.

A brute-force attack targets a specific username to try many weak and predictable passwords. While a password spraying attack targets many usernames using one common weak password, which could help avoid an account lockout policy.



Common and weak passwords often follow a pattern and format. Some commonly used passwords and their overall format can be found below.

- The current season followed by the current year (SeasonYear). For example, **Fall2020**, **Spring2021**, etc.
- The current month followed by the current year (MonthYear). For example, **November2020**, **March2021**, etc.
- Using the company name along with random numbers (CompanyNameNumbers). For example, **TryHackMe01**, **TryHackMe02**.

Hashcat

```
user@THM:~$ hydra -L usernames-list.txt -p Spring2021 ssh://10.1.1.10
[INFO] Successful, password authentication is supported by ssh://10.1.1.10:22
[22][ssh] host: 10.1.1.10 login: victim password: Spring2021
[STATUS] attack finished for 10.1.1.10 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
```

RDP

Let's assume that we found an exposed RDP service on port 3026. We can use a tool such as RDPassSprayto password spray against RDP.

lets try using the -u option to specify the victim as a username and the -p option set the Spring2021!. the -y option is to select a single host to attack.

```
Hashcat

user@THM:~# python3 RDPassSpray.py -u victim -p Spring2021! -t 10.100.10.240:3026
[13-02-2021 16:47] - Total number of users to test: 1
[13-02-2021 16:47] - Total number of password to test: 1
[13-02-2021 16:47] - Total number of attempts: 1
[13-02-2021 16:47] - [*] Started running at: 13-02-2021 16:47:40
[13-02-2021 16:47] - [+] Cred successful (maybe even Admin access!): victim ::
Spring2021!
```

The above output shows that we successfully found valid credentials victim:Spring2021!. Note that we can specify a domain name using the -d option if we are in an Active Directory environment.

```
Hashcat

user@THM:~# python3 RDPassSpray.py -U usernames-list.txt -p Spring2021! -d THM-labs
-T RDP_servers.txt
```

Outlook web access (OWA) portal

Tools:

- SprayingToolkit (atomizer.py)
- MailSniper

SMB

- Tool: Metasploit (auxiliary/scanner/smb/smb_login)

1. **Perform a password spraying attack to get access to the SSH://10.10.90.3 server to read /etc/flag. What is the flag?**

THM{a97a26e86d09388bbea148f4b870277d}