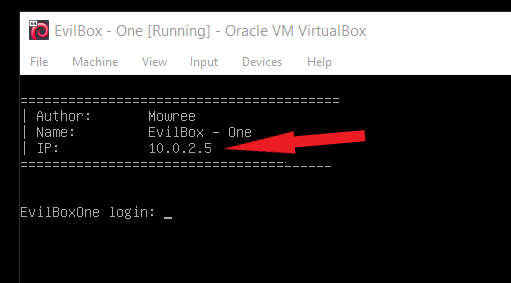
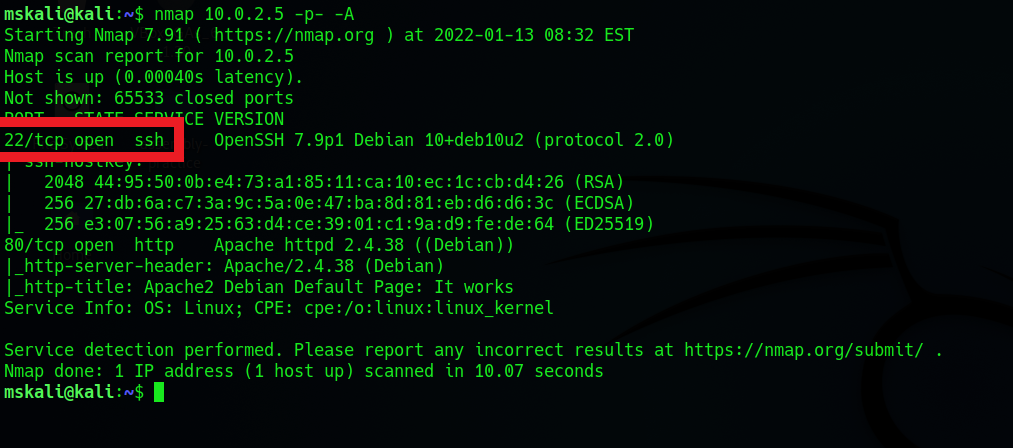
Our first step was detecting the machine IP by lunching the machine and reading the IP.



By getting the machine IP our next step was scanning it using NMAP.



We found 2 open ports:

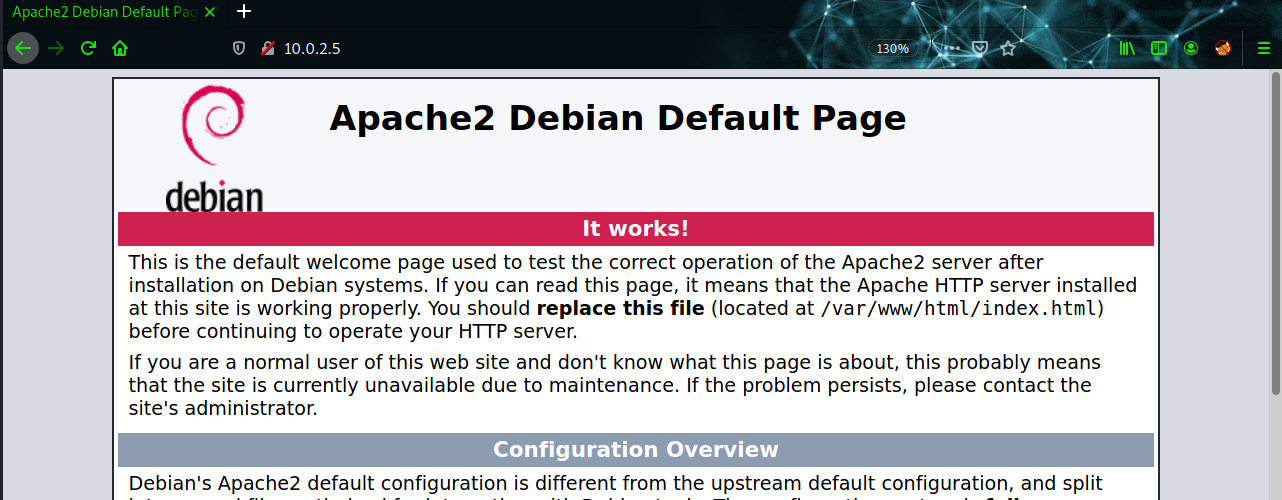
22/SSH – 7.9p1 Debian

80/HTTP – Apache/2.4.38 Debian

The SSH port will be used in further steps.

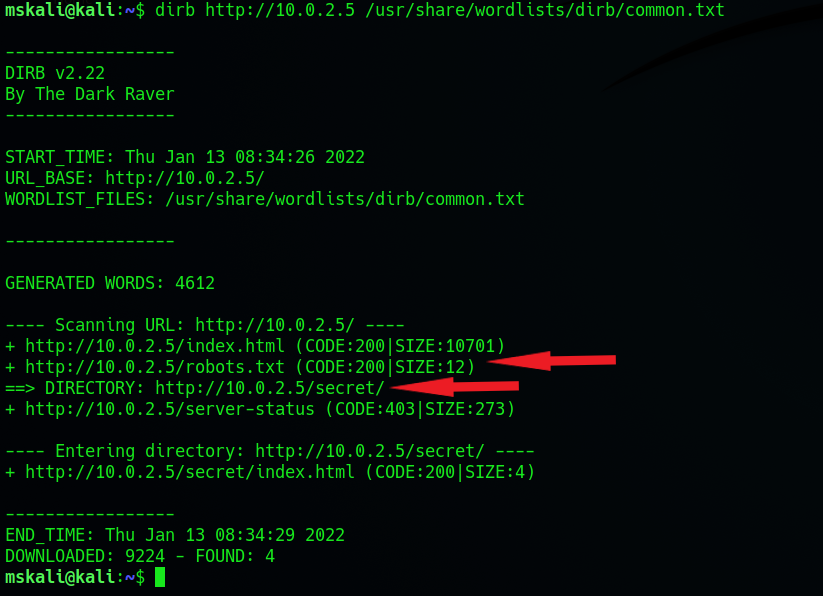
Meanwhile we used port 80 to check the web application.

Visiting the web application on the browser:



Nothing interesting regards the web application is running on Apache web server. As seen on the NMAP scan.

In order to discover the web application, we used Dirb & Gobuster to enumerate all the directories that can be accessed on this application.



Two interesting directories have been found (200 – can be accessed)

Notice that server-status page can’t be accessed (403 – forbidden)

* ” Server-status” page can have a lot of value and useful information about the application and the server its running on.

Checking the /robots.txt page:



H4x0r might be a username that can be used in further steps.

Meanwhile this page doesn’t help much

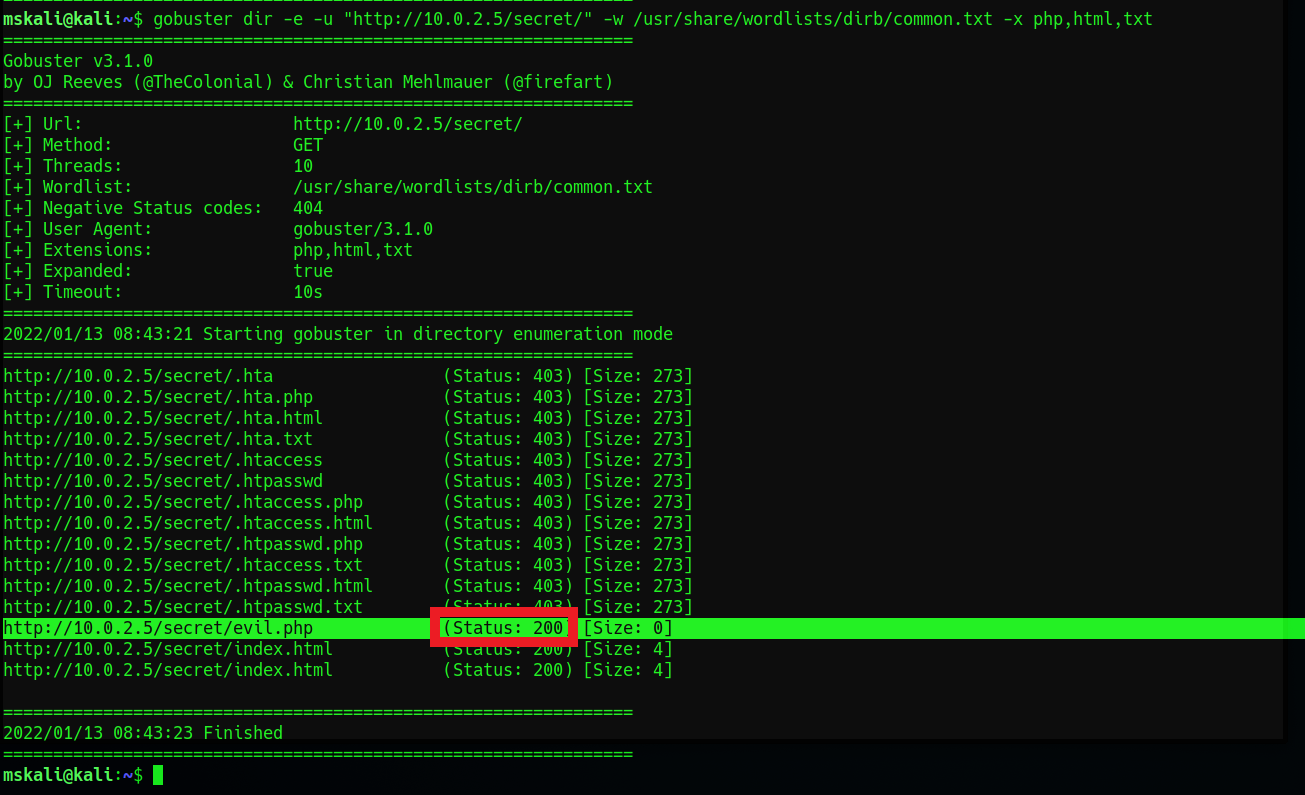
Checking the other 200 status directory discovered by Dirb.



Seems that nothing interesting to be found.

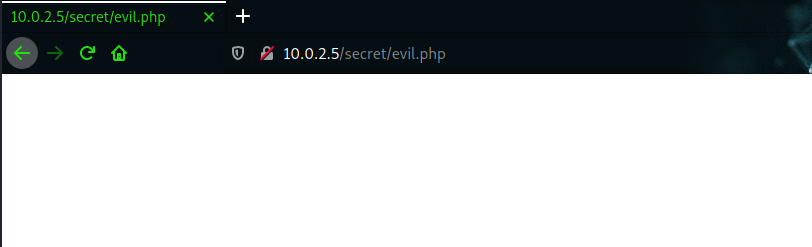
Because it is a Directory, it is possible to enumerate all pages on it.

Gobuster tool is used in this case:



“evil.php” page was found with status 200. (index.html) doesn’t help much.

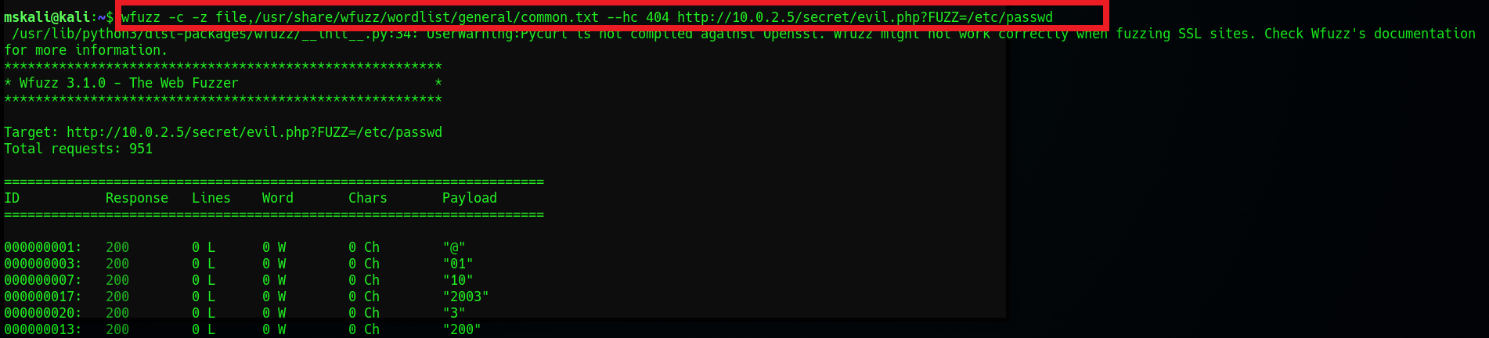
Checking the evil.php page:



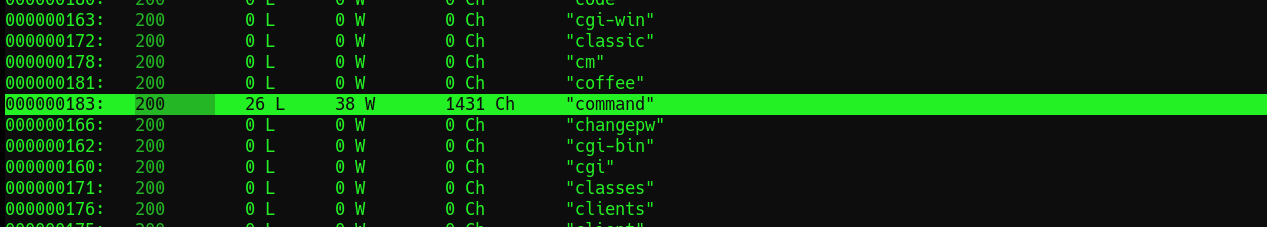
Nothing to be seen on this page.

Usually a .php page have functions to handle instructions on the webserver/page.

In order to enumerate the .php file and extract information from it. WFUZZ tool is used:

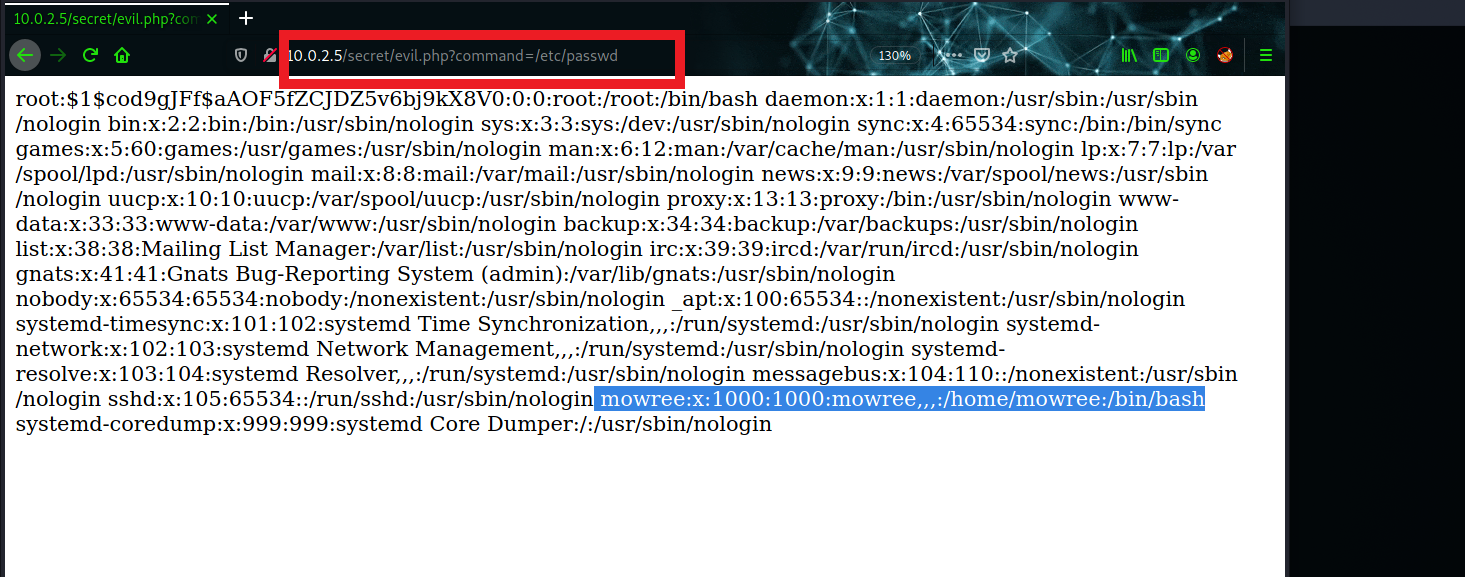


Notice the syntax of the command, this command syntax checks if there is a possible LFI injection vulnerability.



LFI vulnerability found using the variable “command”.

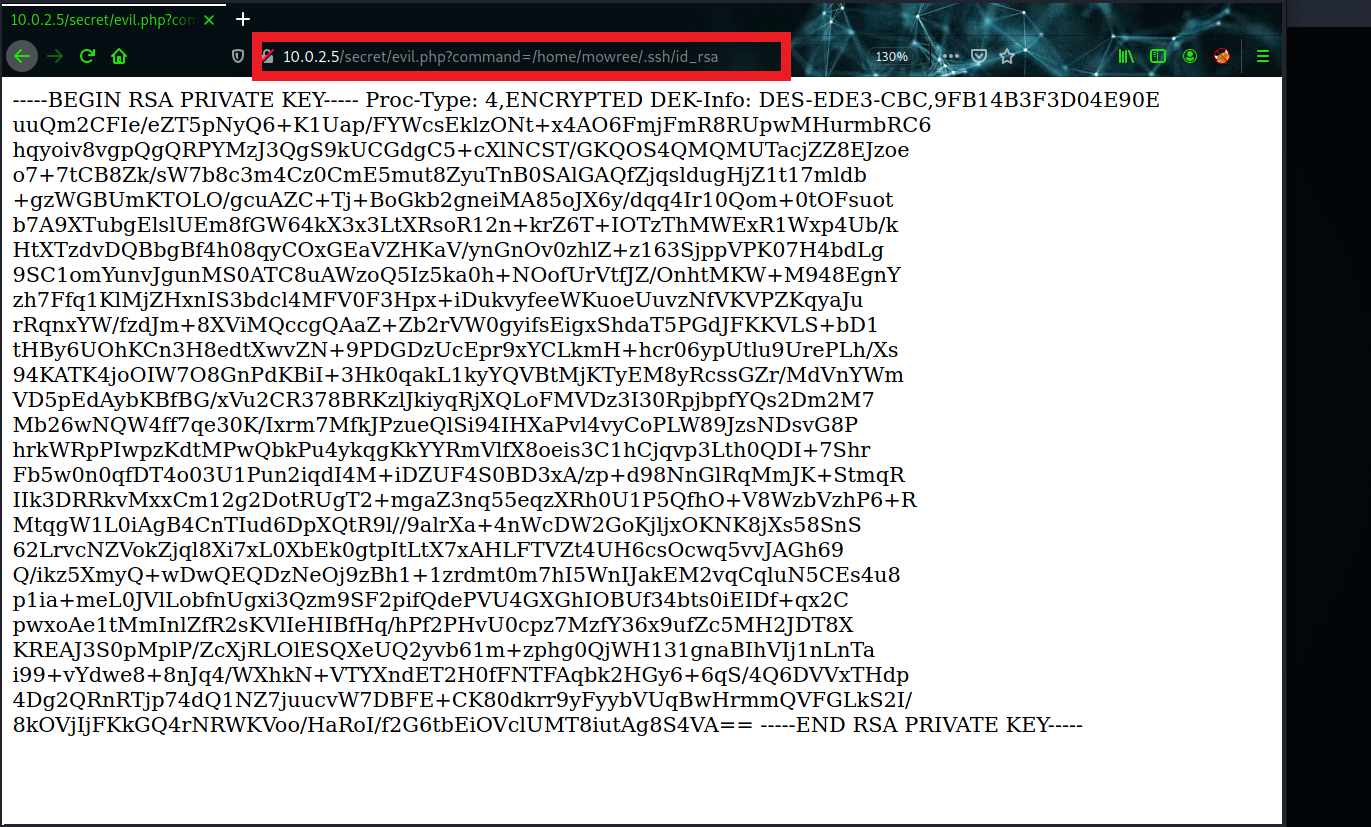
This means an unauthorized user can access certain files on the web server.



Accessing /etc/passwd file and reading it remotely allows us to discover the users on the system, an interesting user is “mowree” UID=1000 and it has a home directory.

This information and the information we received previously in the NMAP scan (SSH) can be connected together in order to access this user.

The LFI vulnerability allows us to access the users SSH keys that are placed on the home directory.



Copying the key to file on our kali machine in order to use it for the SSH connection.

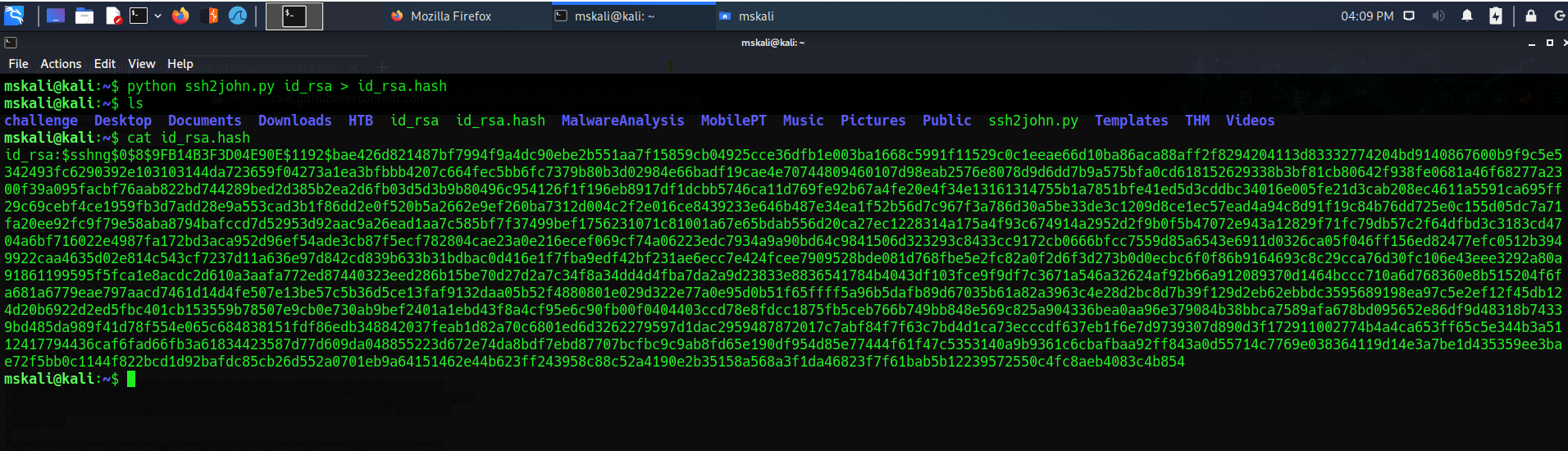
Notice that new key file must be protected in order to successfully connect to the server. This means that only the owner of the file has full read and write access to it.

The file can be protected by applying the following command to it :

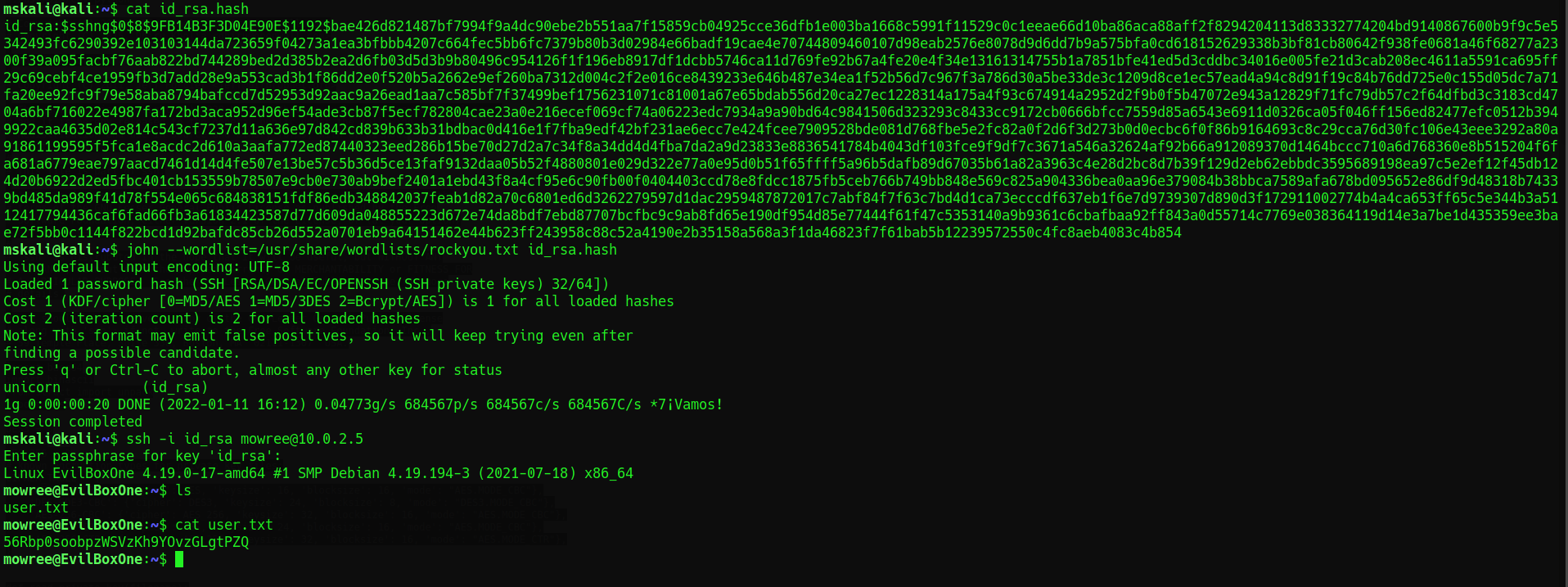
“chmod 600 [filename]”

Our next step was trying to connect to the sever using SSH.

Unfortenlty the SSH key was protected with passphrase. Which forces us to Brute-force the SSH key using SSH2John script. ([github-Raw](https://raw.githubusercontent.com/openwall/john/bleeding-jumbo/run/ssh2john.py))

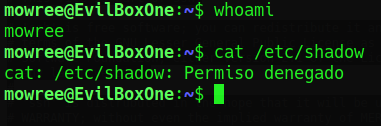


Revealing the hash of the id\_rsa (SSH key) file then Brute-force it using Rockyou.txt wordlist:



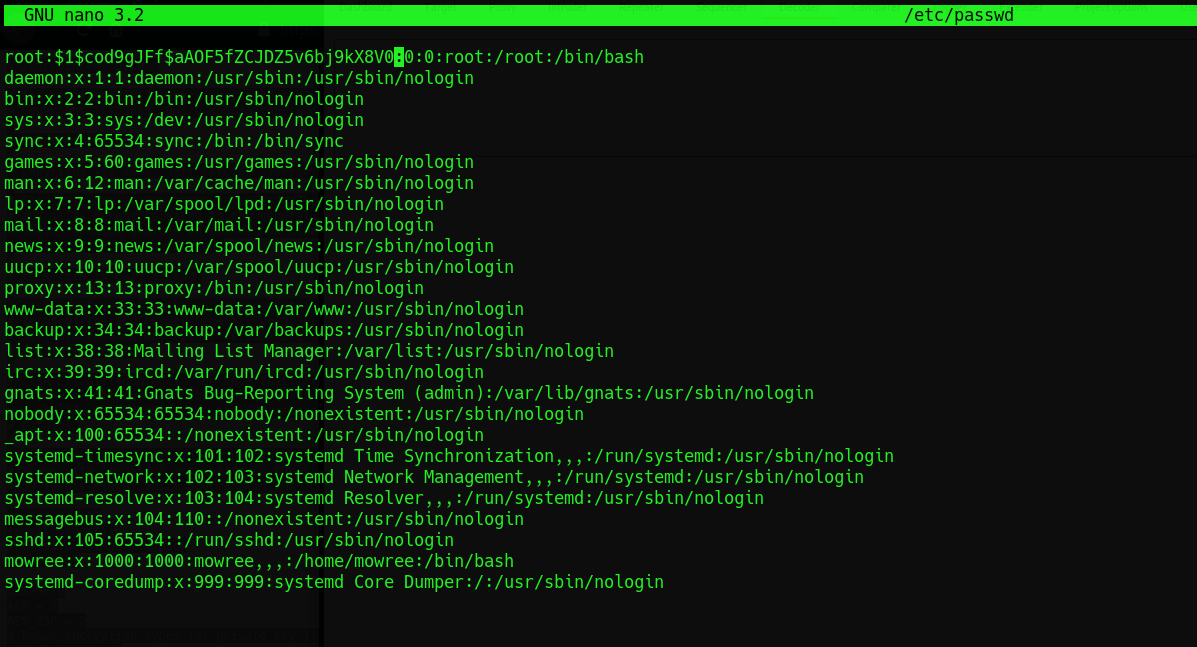
The passphrase was “unicorn” as u can see in the figure above.

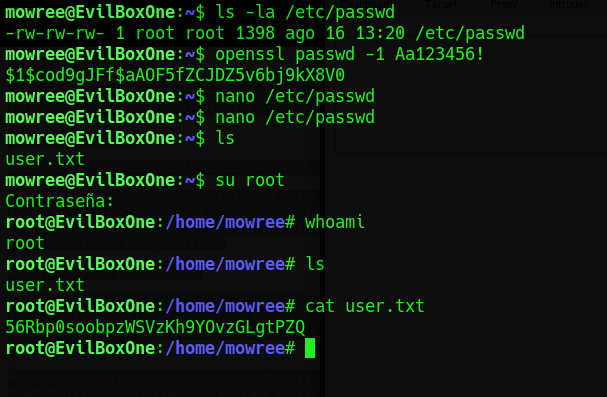
Now after connecting to the server using SSH, our next step is checking the system for important information and the flag.



Notice that we don’t have permissions to get the /etc/shadow file. But enumerating the system to check files that have write permissions with the command:

“find / -writable -type f 2>/dev/null” shows us that /etc/passwd have read and write permission for the current user which means that we can edit it, and this is extremely dangerous because we can edit the root password (hash)





We used OPENSSL tool in order to generate a hash for the root user then injecting the /etc/password with our hash.

A simple privilege escalation step.

Accessing root then capturing the flag.