Security+ Lab Series

Lab 13: Secure Network Administration Principles Log Analysis

Document Version: 2018-08-28

Contents

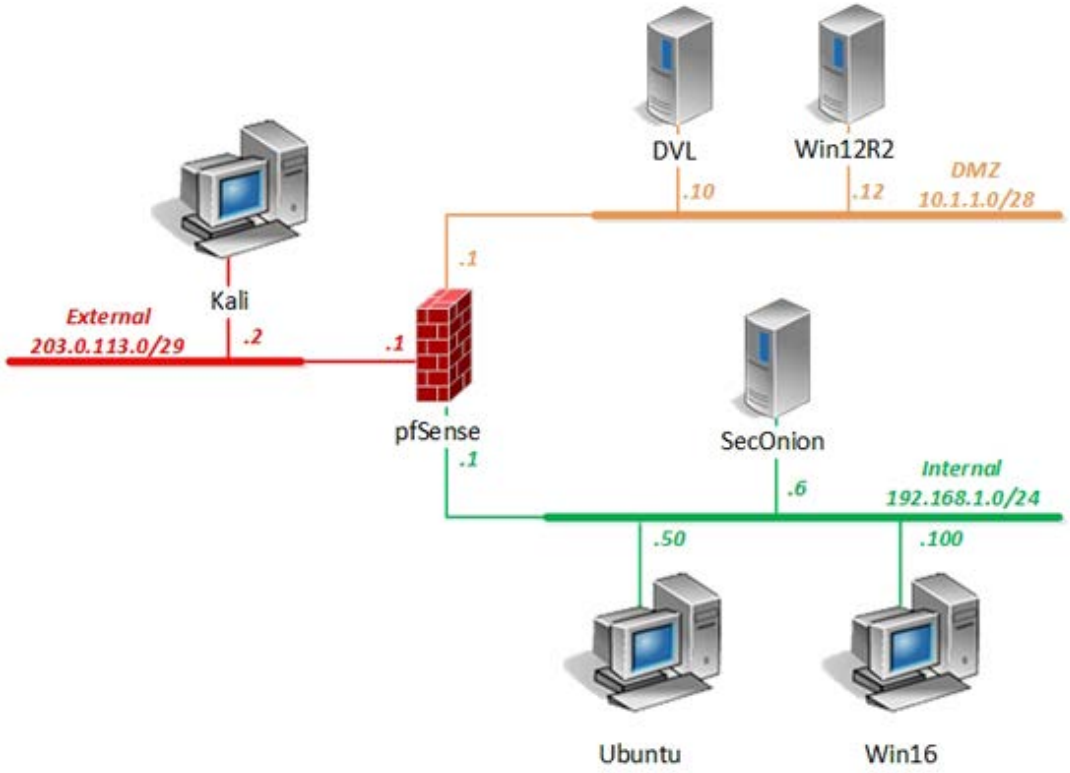
Introduction

In this lab, you will be conducting network log analysis practices using various tools.

Objectives

- Given a scenario, troubleshoot common security issues

Lab Topology



Lab Settings

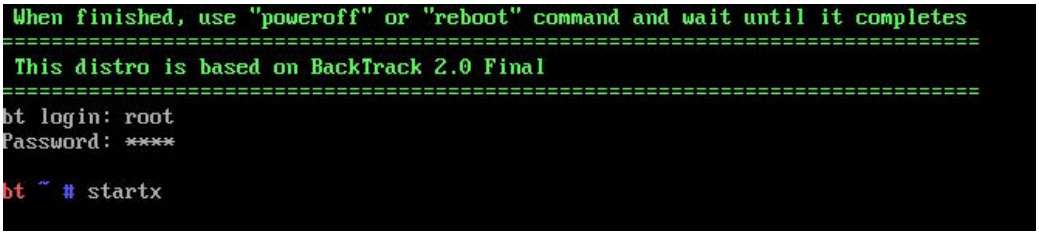
The information in the table below will be needed to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
DVL	10.1.1.10 /28	root	toor
Kali	203.0.113.2 /29	root	toor
pfSense	eth0: 192.168.1.1 /24 eth1: 10.1.1.1 /28 eth2: 203.0.113.1 /29	admin	pfsense
SecOnion	192.168.1.6 /24	soadmin	mypassword
		root	mypassword
Ubuntu	192.168.1.50 /24	student	securepassword
		root	securepassword
Win12R2	10.1.1.12 /28	administrator	Train1ng$
Win16	192.168.1.100 /24	lab-user	Train1ng$
		Administrator	Train1ng$

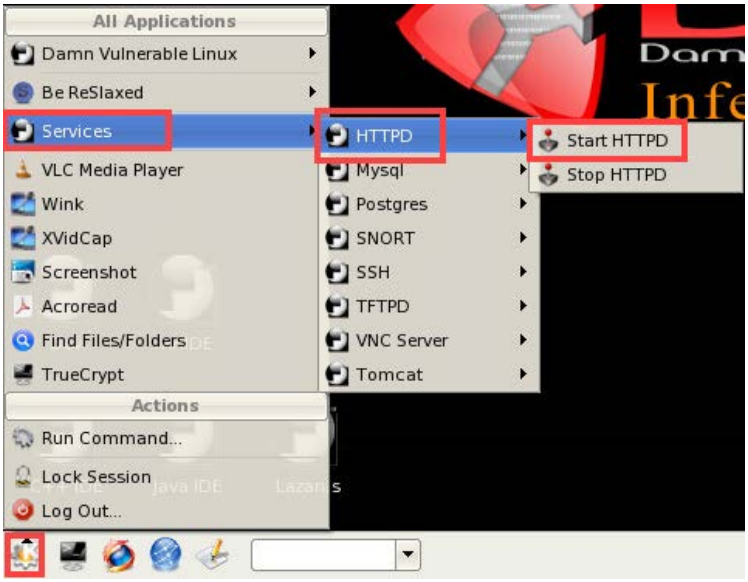
1 Nmap Analysis Using grep

1.1 Analyzing Different Nmap Reports

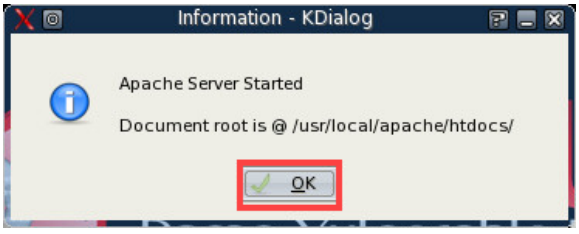
1. Launch the **DVL** virtual machine.
2. On the login screen, type **`root`** followed by pressing the **Enter** key.
3. When prompted for a password, type **`toor`** and press **Enter** again.
4. When presented with the user prompt, type **`startx`** and then press **Enter**.



5. Once logged in, click on the **Application Menu** icon located towards the bottom-left corner and navigate to **Services > HTTPD > Start HTTPD** to initialize the *HTTP* service on the server.



6. If a dialog message appears, click **OK**.



7. In the bottom taskbar, click on the **terminal** icon.

8. Start the FTP service by typing the command below followed by pressing the **Enter** key.

```
bt~# proftpd
```

Wait 1 minute for the service to start. Once the prompt comes back, the service is started. You may ignore the *IPv6* error and continue to the next step.

9. Launch the **Kali** virtual machine to access the graphical login screen.
10. Log in as **root** with **toor** as the password. Open the **Kali** *PC Viewer*.
11. Click on the icon located in the top menu bar.

12. Navigate to the **/tmp/reports** directory by entering the command below.

```
root@Kali-Attacker:~# cd /tmp/reports
```

13. Enter the command below to open a *Nmap* report in the *Leafpad GUI* text editor.

```
root@Kali-Attacker:/tmp/reports# leafpad dvlscan1.xml
```

```
<?xml version="1.0"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 6.47 scan initiated Wed Apr 15 11:27:46 2015 as: nmap -sS -oA dvlscan1 10.1.1.10 -->
<nmaprun scanner="nmap" args="nmap -sS -oA dvlscan1 10.1.1.10" start="1429111666" startstr="Wed
Apr 15 11:27:46 2015" version="6.47" xmloutputversion="1.04">
<scaninfo type="syn" protocol="tcp" numservices="1000"
services="1,3-4,6-7,9,13,17,19-26,30,32-33,37,42-43,49,53,70,79-85,88-90,99-100,106,109-111,113,119,
>
<verbose level="0"/>
<debugging level="0"/>
<host starttime="1429111666" endtime="1429111666"><status state="up" reason="echo-reply"
reason_ttl="63"/>
<address addr="10.1.1.10" addrtype="ipv4"/>
<hostnames>
</hostnames>
<ports><extraports state="closed" count="988">
<extrareasons reason="resets" count="988"/>
</extraports>
<port protocol="tcp" portid="21"><state state="open" reason="syn-ack" reason_ttl="63"/><service
```


14. Based on this report, we can see what ports/services are open on the date listed in the report. Notice that this format can be difficult to read. **Close** the text editor window.
15. Open a similar *dvlscan* report, but this time the format will be in *.gnmap*. Enter the command below.

```
root@Kali-Attacker:/tmp/reports# leafpad dvlscan1.gnmap
```

```
# Nmap 6.47 scan initiated Wed Apr 15 11:27:46 2015 as: nmap -sS -oA dvlscan1 10.1.1.10
Host: 10.1.1.10 ()      Status: Up
Host: 10.1.1.10 ()      Ports: 21/open/tcp//ftp///, 22/open/tcp//ssh///, 80/open/tcp//http///, 139/
open/tcp//netbios-ssn///, 199/open/tcp//smux///, 445/open/tcp//microsoft-ds///, 631/open/tcp//
ipp///, 3306/open/tcp//mysql///, 5801/open/tcp//vnc-http-1///, 5901/open/tcp//vnc-1///, 6000/open/
tcp//X11///, 6001/open/tcp//X11:1///    Ignored State: closed (988)
# Nmap done at Wed Apr 15 11:27:46 2015 -- 1 IP address (1 host up) scanned in 0.19 seconds
```

16. This is the same output from the previously opened *XML* file except that this format (*GNMAP*) is considered a *grep*-able *Nmap* output. **Close** the window.
17. Leave the *Kali* viewer open to continue with the next task.

1.2 Parsing Nmap Reports with CLI

1. Enter the command below to grep the first field of the **dvlscan1.gnmap** file.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut –d” “ –f1
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut -d" " -f1
Host:
root@Kali-Attacker:/tmp/reports#
```

Notice that the text “*Host:*” appears in the output. When using the cut command with the (*-d” “*) option, we are cutting out the spaces in the file. Adding the “*-f1*” option to that, we are cutting everything out except for the first field, which in this case was “*Host:*”.

2. Type the same command from the previous step, except for this time we will cut the second field.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut –d” “ –f2
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut -d" " -f2
10.1.1.10
root@Kali-Attacker:/tmp/reports#
```

Notice that we now were able to parse the live host IP from the *Nmap* report.

3. Issue the same command as before, but this time we will redirect the output to a file called *livehosts.txt*.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut –d” “ –f2
> livehosts.txt
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.gnmap | grep open | cut -d" " -f2 > livehosts.txt
root@Kali-Attacker:/tmp/reports#
```

4. Note that no confirmation appears from the command above. Type the **`ls -l`** command to verify the *livehosts.txt* file is created.

```
root@Kali-Attacker:/tmp/reports# ls -l
total 84
-rw-r--r-- 1 root root   561 Apr 15  2015 dvlscan1.gnmap
-rw-r--r-- 1 root root   553 Apr 15  2015 dvlscan1.nmap
-rw-r--r-- 1 root root  6631 Apr 15  2015 dvlscan1.xml
-rw-r--r-- 1 root root  1892 Apr 15  2015 livehostscan.txt
-rw-r--r-- 1 root root    10 Aug  3 15:31 livehosts.txt
-rw-r--r-- 1 root root    50 Dec 23  2017 liveports.txt
-rw-r--r-- 1 root root  1382 Apr 15  2015 networkscan1.gnmap
-rw-r--r-- 1 root root  1502 Apr 15  2015 networkscan1.nmap
-rw-r--r-- 1 root root 10652 Apr 15  2015 networkscan1.xml
-rw-r--r-- 1 root root  1714 Apr 15  2015 networkscan2.gnmap
-rw-r--r-- 1 root root  2145 Apr 15  2015 networkscan2.nmap
-rw-r--r-- 1 root root 18764 Apr 15  2015 networkscan2.xml
-rw-r--r-- 1 root root  1693 Dec 23  2017 parsed.txt
-rw-r--r-- 1 root root    68 Apr 15  2015 targets.txt
```

5. Enter the command below to view the output from the *dvlscan1.nmap* file.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap
# Nmap 6.47 scan initiated Wed Apr 15 11:27:46 2015 as: nmap -sS -oA dvlscan1 10.1.1.10
Nmap scan report for 10.1.1.10
Host is up (0.0014s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
199/tcp  open  smux
445/tcp  open  microsoft-ds
631/tcp  open  ipp
3306/tcp open  mysql
5801/tcp open  vnc-http-1
5901/tcp open  vnc-1
6000/tcp open  X11
6001/tcp open  X11:1

# Nmap done at Wed Apr 15 11:27:46 2015 -- 1 IP address (1 host up) scanned in 0.19 seconds
```

Notice how this output closely resembles the output we usually get from a *Nmap* scan.

6. Type the command below to grep lines that include the word **open**.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
199/tcp  open  smux
445/tcp  open  microsoft-ds
631/tcp  open  ipp
3306/tcp open  mysql
5801/tcp open  vnc-http-1
5901/tcp open  vnc-1
6000/tcp open  X11
6001/tcp open  X11:1
root@Kali-Attacker:/tmp/reports#
```

7. Include the cut command now as shown below to cut the "/" delimiter character and the first field, as we are mostly interested in grep-ing a list of port numbers.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open | cut –d"/" –f1
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open | cut -d"/" -f1
21
22
80
139
199
445
631
3306
5801
5901
6000
6001
root@Kali-Attacker:/tmp/reports#
```

8. Issue the same command as before, but this time save the output to a file called **liveports.txt**.

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open | cut –d"/" –f1
> liveports.txt
```

```
root@Kali-Attacker:/tmp/reports# cat dvlscan1.nmap | grep open | cut -d"/" -f1 > liveports.txt
root@Kali-Attacker:/tmp/reports#
```


9. View the contents of the **liveports.txt** file by typing the command below, followed by pressing the **Enter** key to confirm the contents of the file.

```
root@Kali-Attacker:/tmp/reports# cat liveports.txt
```

```
root@Kali-Attacker:/tmp/reports# cat liveports.txt
21
22
80
139
199
445
631
3306
5801
5901
6000
6001
root@Kali-Attacker:/tmp/reports#
```

1.3 Parsing Nmap Reports with Scripts

1. View the output of **livehostscan.txt** by issuing the command below.

```
root@Kali-Attacker:/tmp/reports# cat livehostscan.txt
```

```
root@Kali-Attacker:/tmp/reports# cat livehostscan.txt
# Nmap 6.47 scan initiated Wed Apr 15 15:29:06 2015 as: nmap -sV -oG livehostscan.txt -iL targe
ts.txt
Host: 192.168.1.1 ()    Status: Up
Host: 192.168.1.1 ()    Ports: 53/open/tcp//domain//NLNet Labs Unbound/, 80/open/tcp//http//lig
httpd 1.4.35/, 3128/open/tcp//http-proxy//Squid http proxy 2.7.STABLE9/ Ignored State: filtered
 (997)
Host: 192.168.1.6 ()    Status: Up
Host: 192.168.1.6 ()    Ports: 22/open/tcp//ssh//OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux;
 protocol 2.0)/, 25/open/tcp//smtp//Postfix smtpd/, 80/open/tcp//http//nginx 1.1.19/, 514/open/
tcp//shell?///  Ignored State: closed (996)
Host: 192.168.1.50 ()   Status: Up
Host: 192.168.1.50 ()   Ports: 21/open/tcp//ftp//ProFTPD 1.3.4a/, 22/open/tcp//tcpwrapped///, 2
3/open/tcp//telnet//Linux telnetd/, 80/open/tcp//http//Apache httpd 2.2.22 ((Ubuntu))/  Ignored
 State: closed (996)
Host: 10.1.1.1 ()       Status: Up
Host: 10.1.1.1 ()       Ports: 53/open/tcp//domain//NLNet Labs Unbound/, 80/open/tcp//http//lig
httpd 1.4.35/   Ignored State: filtered (998)
Host: 10.1.1.10 ()      Status: Up
Host: 10.1.1.10 ()      Ports: 21/open/tcp//ftp//ProFTPD 1.3.0/, 22/open/tcp//ssh//OpenSSH 4.4
(protocol 1.99)/, 80/open/tcp//http//Apache httpd 1.3.37 ((Unix) PHP|4.4.4)/, 139/open/tcp//net
bios-ssn//Samba smbd 3.X (workgroup: WORKGROUP)/, 199/open/tcp//smux//Linux SNMP multiplexer/,
445/open/tcp//netbios-ssn//Samba smbd 3.X (workgroup: WORKGROUP)/, 631/open/tcp//ipp//CUPS 1.1/
, 3306/open/tcp//mysql//MySQL (unauthorized)/, 5801/open/tcp//http-proxy//sslstrip/, 5901/open/
tcp//vnc//VNC (protocol 3.7)/, 6000/open/tcp//X11//(access denied)/, 6001/open/tcp//X11//(acces
s denied)/      Ignored State: closed (988)
Host: 203.0.113.1 ()    Status: Up
Host: 203.0.113.1 ()    Ports: 53/open/tcp//domain//NLNet Labs Unbound/, 80/open/tcp//http//lig
httpd 1.4.35/   Ignored State: filtered (998)
# Nmap done at Wed Apr 15 15:32:07 2015 -- 6 IP addresses (6 hosts up) scanned in 180.63 second
s
```

Notice how this scan report includes multiple targets.

2. Use the **scanreport.sh** script to automatically parse the **livehostscan.txt** file. Enter the command below into the *terminal*.

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh –f
livehostscan.txt
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f livehostscan.txt
# Nmap 6.47 scan initiated Wed Apr 15 15:29:06 2015 as: nmap -sV -oG livehostscan.txt -iL targets.txt

Host: 192.168.1.1 ()
53      open    tcp             domain          NLNet Labs Unbound
80      open    tcp             http            lighttpd 1.4.35
3128    open    tcp             http-proxy              Squid http proxy 2.7.STABLE9            Ignored State: f
iltered (997)

Host: 192.168.1.6 ()
22      open    tcp             ssh             OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux; protocol 2.0)
25      open    tcp             smtp            Postfix smtpd
80      open    tcp             http            nginx 1.1.19

Host: 192.168.1.50 ()
21      open    tcp             ftp             ProFTPD 1.3.4a
22      open    tcp             tcpwrapped
23      open    tcp             telnet          Linux telnetd

Host: 10.1.1.1 ()
53      open    tcp             domain          NLNet Labs Unbound
80      open    tcp             http            lighttpd 1.4.35         Ignored State: filtered (998)

Host: 10.1.1.10 ()
21      open    tcp             ftp             ProFTPD 1.3.0
22      open    tcp             ssh             OpenSSH 4.4 (protocol 1.99)
80      open    tcp             http            Apache httpd 1.3.37 ((Unix) PHP|4.4.4)
139     open    tcp             netbios-ssn             Samba smbd 3.X (workgroup: WORKGROUP)
199     open    tcp             smux            Linux SNMP multiplexer
445     open    tcp             netbios-ssn             Samba smbd 3.X (workgroup: WORKGROUP)
631     open    tcp             ipp             CUPS 1.1
3306    open    tcp             mysql           MySQL (unauthorized)
5801    open    tcp             http-proxy              sslstrip
```

Notice the output in a nice readable format.

3. We can parse the file even more by taking out the lines that begin with a comment **(#)** character. Enter the command below.

```
root@Kali-Attacker:/tmp/reports# /grep –v ^# livehostscan.txt > parsed.txt
```

```
root@Kali-Attacker:/tmp/reports# grep -v ^# livehostscan.txt > parsed.txt
root@Kali-Attacker:/tmp/reports#
```


4. Use the **scanrreport.sh** script again to see results. Type the command below.

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh –f parsed.txt
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f parsed.txt

Host: 192.168.1.1 ()
53      open    tcp             domain          NLNet Labs Unbound
80      open    tcp             http            lighttpd 1.4.35
3128    open    tcp             http-proxy              Squid http proxy 2.7.STABLE9            Ignored State: f
iltered (997)

Host: 192.168.1.6 ()
22      open    tcp             ssh             OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux; protocol 2.0)
25      open    tcp             smtp            Postfix smtpd
80      open    tcp             http            nginx 1.1.19

Host: 192.168.1.50 ()
21      open    tcp             ftp             ProFTPD 1.3.4a
22      open    tcp             tcpwrapped
23      open    tcp             telnet          Linux telnetd

Host: 10.1.1.1 ()
53      open    tcp             domain          NLNet Labs Unbound
80      open    tcp             http            lighttpd 1.4.35         Ignored State: filtered (998)

Host: 10.1.1.10 ()
21      open    tcp             ftp             ProFTPD 1.3.0
22      open    tcp             ssh             OpenSSH 4.4 (protocol 1.99)
80      open    tcp             http            Apache httpd 1.3.37 ((Unix) PHP|4.4.4)
139     open    tcp             netbios-ssn             Samba smbd 3.X (workgroup: WORKGROUP)
199     open    tcp             smux            Linux SNMP multiplexer
445     open    tcp             netbios-ssn             Samba smbd 3.X (workgroup: WORKGROUP)
631     open    tcp             ipp             CUPS 1.1
3306    open    tcp             mysql           MySQL (unauthorized)
```

5. Parse even further by only showing output for a specific IP address. Issue the command below to show the output for **192.168.1.50**.

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh –f parsed.txt –i
192.168.1.50
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f parsed.txt -i 192.168.1.50

Host: 192.168.1.50 ()
21      open    tcp             ftp             ProFTPD 1.3.4a
22      open    tcp             tcpwrapped
23      open    tcp             telnet          Linux telnetd
root@Kali-Attacker:/tmp/reports#
```

6. If you are interested in knowing which targets have a specific port opened, execute the command below the show results for **port 21** only.

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh –f parsed.txt –p
21
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f parsed.txt -p 21

Host: 192.168.1.50 ()
21      open    tcp             ftp             ProFTPD 1.3.4a

Host: 10.1.1.10 ()
21      open    tcp             ftp             ProFTPD 1.3.0
root@Kali-Attacker:/tmp/reports#
```


7. We can also parse by protocol name.

```
root@Kali-Attacker:/tmp/reports# /home/scritps/scanreport.sh –f parsed.txt –s
ftp
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f parsed.txt -s ftp

Host: 192.168.1.50 ()
21      open    tcp             ftp             ProFTPD 1.3.4a

Host: 10.1.1.10 ()
21      open    tcp             ftp             ProFTPD 1.3.0
root@Kali-Attacker:/tmp/reports#
```

8. See which network system(s) have **port 80** open. Enter the command below into the terminal.

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh –f parsed.txt –p
80
```

```
root@Kali-Attacker:/tmp/reports# /home/scripts/scanreport.sh -f parsed.txt -p 80

Host: 192.168.1.1 ()
80      open    tcp             http            lighttpd 1.4.35

Host: 192.168.1.6 ()
80      open    tcp             http            nginx 1.1.19

Host: 192.168.1.50 ()

Host: 10.1.1.1 ()
80      open    tcp             http            lighttpd 1.4.35         Ignored State: filtered (998)

Host: 10.1.1.10 ()
80      open    tcp             http            Apache httpd 1.3.37 ((Unix) PHP|4.4.4)

Host: 203.0.113.1 ()
80      open    tcp             http            lighttpd 1.4.35         Ignored State: filtered (998)
root@Kali-Attacker:/tmp/reports#
```

Observe the five systems from the output.

9. Leave the *Kali* viewer open to continue with the next task.

2 Log Analysis Using grep

2.1 Using grep With Curl

1. While logged into the *Kali* system, use the **curl** command to pull an *HTML* webpage from the potential web server on **192.168.1.50**.

```
root@Kali-Attacker:/tmp/reports# curl http://192.168.1.50
```

2. If the results from the *curl* command are large, it will be helpful to filter through the output using the grep command. See if we can find an email address on the web page.

```
root@Kali-Attacker:/tmp/reports# curl http://192.168.1.6 | grep @
```

```
root@Kali-Attacker:/tmp/reports# curl http://192.168.1.50 | grep @
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   252  100   252    0     0  75607      0 --:--:-- --:--:-- --:--:--  123k
<p>For help and support, please contact: admin@example.com</p>
root@Kali-Attacker:/tmp/reports#
```

The "@" symbol helps signify that an email address has been found when observing the contents.

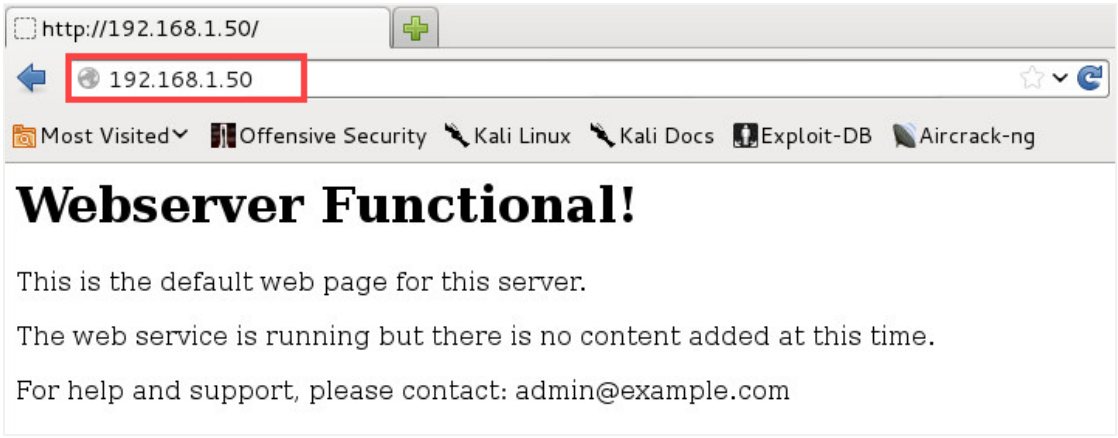
3. Next, we will generate some noise by initiating an intense *Nmap* scan. Type the command below into the **terminal**.

```
root@Kali-Attacker:/tmp/reports# nmap -T4 -A -v 192.168.1.50
```

Scan will take approximately 2-3 minutes to complete. Move on to the next step while the *Nmap* scan is running.

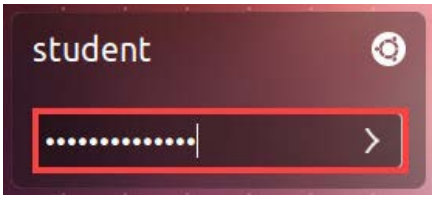
4. Generate more traffic by opening a web browser. Click on the **Iceweasel** icon located on the top menu pane.

5. Enter **`http://192.168.1.50`** into the address bar. Press **Enter**.



2.2 Using grep With Logs

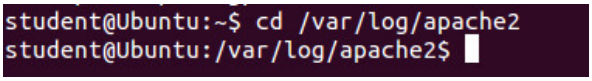
1. Launch the **Ubuntu** virtual machine to access the graphical login screen.
2. Log in as **`student`** with **`securepassword`** as the password.



3. Open a terminal window by clicking on the **terminal** icon located in the left menu pane.

4. In the terminal, change the current directory to **/var/log/apache2** .

```
student@Ubuntu:~$ cd /var/log/nginx
```



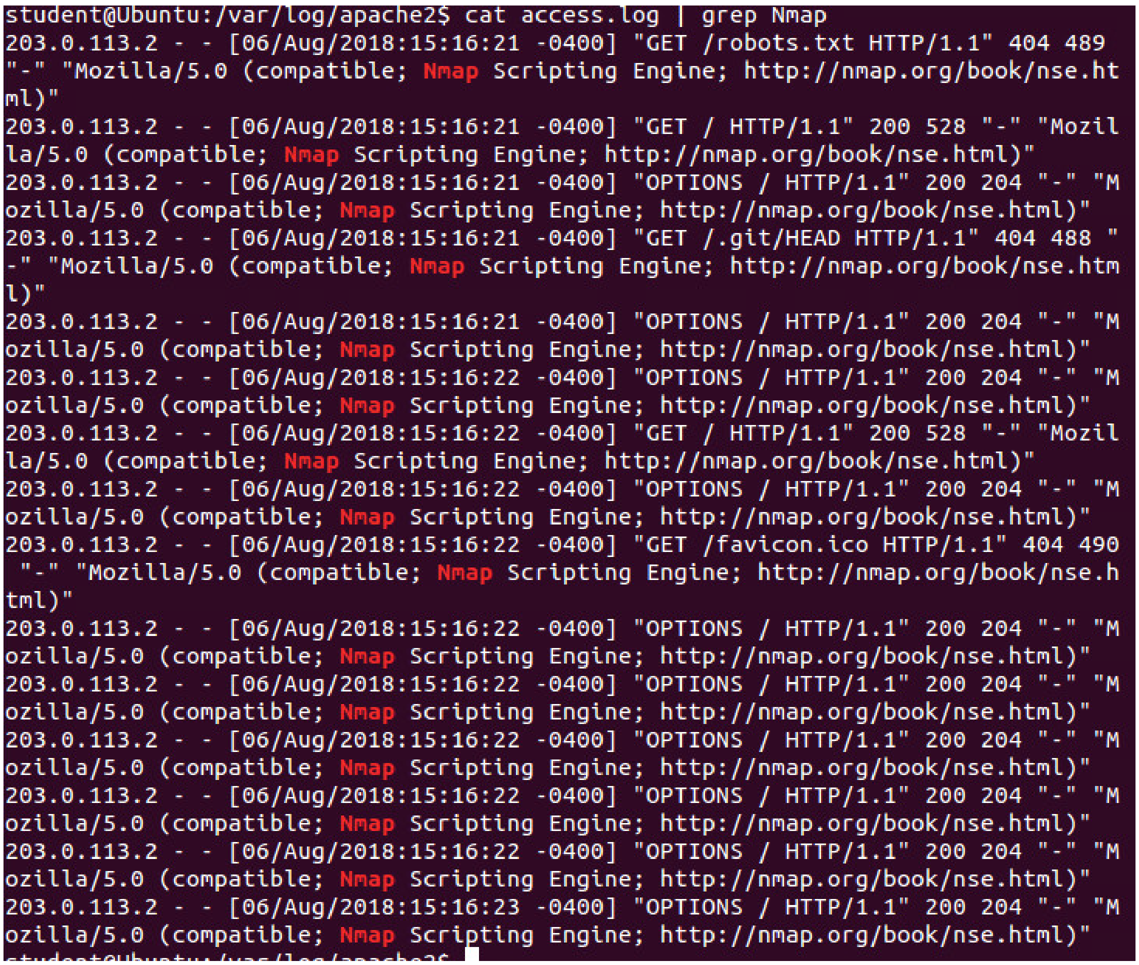
5. View the **access_log** by typing the command below.

```
student@Ubuntu:/var/log/apache2$ cat access.log
```

Notice that the output can be quite long.

6. Cut this down and only analyze potential *Nmap* scans that were initiated on this system (case sensitive) by typing the command below.

```
student@Ubuntu:/var/log/apache2$ cat access.log | grep Nmap
```



7. Type another command to only show entries made with *Firefox* (case sensitive).

```
student@Ubuntu:/var/log/apache2$ cat access.log | grep Firefox
```

```
student@Ubuntu:/var/log/apache2$ cat access.log | grep Firefox
203.0.113.2 - - [06/Aug/2018:15:15:40 -0400] "GET / HTTP/1.1" 200 526 "-" "Mozilla/5.0 (X11; Linux i686; r
v:24.0) Gecko/20140925 Firefox/24.0 Iceweasel/24.8.1"
203.0.113.2 - - [06/Aug/2018:15:15:40 -0400] "GET /favicon.ico HTTP/1.1" 404 502 "-" "Mozilla/5.0 (X11; Li
nux i686; rv:24.0) Gecko/20140925 Firefox/24.0 Iceweasel/24.8.1"
203.0.113.2 - - [06/Aug/2018:15:15:40 -0400] "GET /favicon.ico HTTP/1.1" 404 502 "-" "Mozilla/5.0 (X11; Li
nux i686; rv:24.0) Gecko/20140925 Firefox/24.0 Iceweasel/24.8.1"
student@Ubuntu:/var/log/apache2$
```

8. Type the following command to filter the **access_log** file for the word **curl**.

```
student@Ubuntu:/var/log/apache2$ cat access.log | grep curl
```

```
student@Ubuntu:/var/log/apache2$ cat access.log | grep curl
203.0.113.2 - - [06/Aug/2018:15:09:15 -0400] "GET / HTTP/1.1" 200 535 "-" "curl/7.26.0"
203.0.113.2 - - [06/Aug/2018:15:10:11 -0400] "GET / HTTP/1.1" 200 535 "-" "curl/7.26.0"
student@Ubuntu:/var/log/apache2$
```


3 Log Analysis Using gawk

3.1 Creating Groups and Users Remotely

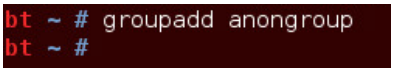
1. Change focus back to the **Kali** system.
2. Within a *terminal* window, enter the following command to *SSH* into a remote system, in this case, the **DVL Server**. When prompted for a password, enter **`toor`** followed by pressing the **Enter** key.

```
root@Kali-Attacker:/tmp/reports# ssh 10.1.1.10
```

Notice the prompt change to *bt~#*. You are now logged in remotely as the *root* user for the *DVL Server*.

3. With root privileges, create a group called **anongroup**.

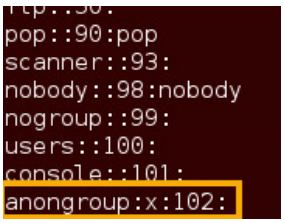
```
bt~# groupadd anongroup
```



No confirmation is given when a group is added like this.

4. View the list of groups on the *DVL Server*. Scroll towards the bottom and confirm that the *anongroup* appears in the list.

```
bt~# cat /etc/group
```



5. Create a new user **ben** and put him in the **anongroup**.

```
bt~# useradd ben –g anongroup
```

   a. Add another user **jerry** using the same command.
   b. Add a third user **katy** using the same command

```
bt ~ # useradd ben -g anongroup
bt ~ # useradd jerry -g anongroup
bt ~ # useradd katy -g anongroup
```

6. Assign the user **ben** a new password. When prompted, type **passb1** for the password. If a warning message displays that the password is too weak, type the password again two more times to confirm.

```
bt~# passwd ben
```

```
bt ~ # passwd ben
Changing password for ben
Enter the new password (minimum of 5, maximum of 127 characters)
Please use a combination of upper and lower case letters and numbers.
New password: ******
Bad password: too simple.
Warning: weak password (enter it again to use it anyway).
New password: ******
Re-enter new password: ******
Password changed.
```

6. Assign user *jerry* the password: **passj1**
7. Assign user *katy* the password: **passk1**
7. Leave the *terminal* window open to continue with the next task.

3.2 Using gawk With Logs

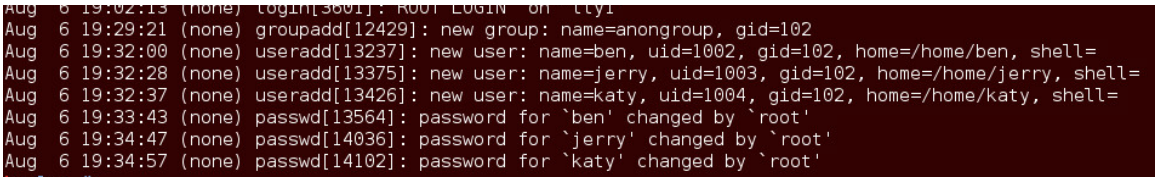
1. While *SSH'd* into the **DVL Server** from the **Kali** system, change to the **/var/log** directory.

```
bt~# cd /var/log
```

```
bt ~ # cd /var/log
bt log #
```


2. View the contents in the log file and scroll down to locate the relevant information about the new group and user creation.

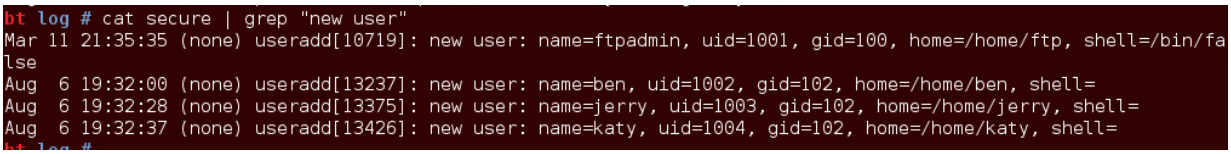
```
bt~# cat secure
```



Notice that at the bottom of the log, entries are shown where user accounts have been created, along with password creations. You will also notice information about incoming *SSH* connections.

3. To parse a search for new instances of new user created within the *secure log* file, enter the command below.

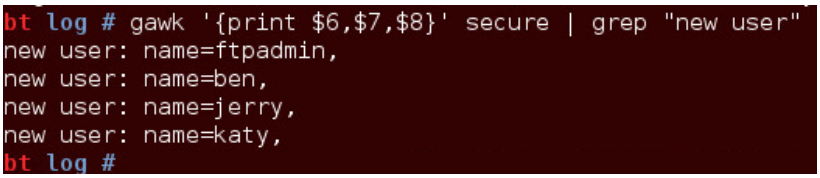
```
bt~# cat secure | grep “new user”
```



Notice the entire lines containing “*new user*” are displayed.

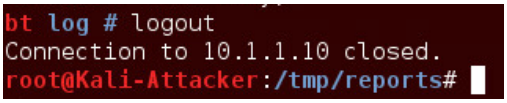
4. To determine the name of the new user created, we can use **grep** and **gawk** together. Enter the command below.

```
bt~# gawk ‘{print $6,$7,$8}’ secure | grep “new user”
```



5. Log out from the *SSH* session.

```
bt~# logout
```



6. Leave the *Kali* window open to continue with the next task.

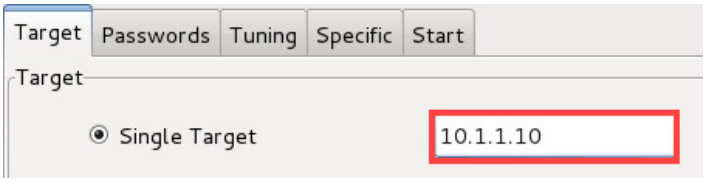
4 FTP Log Analysis

4.1 Password Cracking using Hydra

1. While on the *Kali* system, start up the **Hydra** password cracking application to perform a dictionary attack against a remote system. Type the command below in a **terminal** window.

```
root@Kali-Attacker:/tmp/reports# xhydra
```

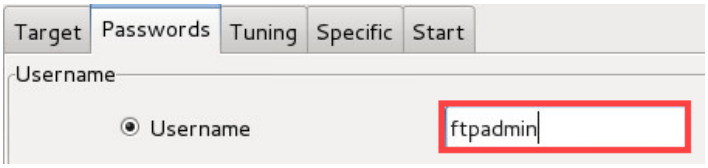
2. A *HydraGTK* graphical user interface will appear. On the *Target* tab, type **`10.1.1.10`** into the *Single Target* field.



3. Select **ftp** as the *Protocol*.

4. Click the **Passwords** tab.

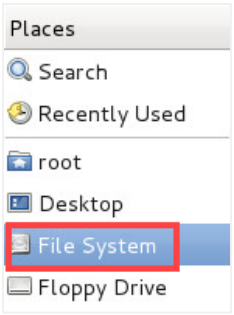
5. Type ***ftpadmin*** for the *Username*.



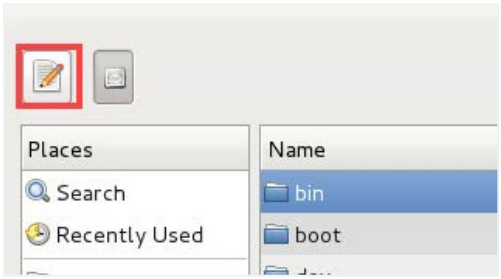
6. In the *Password* pane, select the radio button next to the **Password List** option.

7. Click on the **white space** to the right of *Password List*.

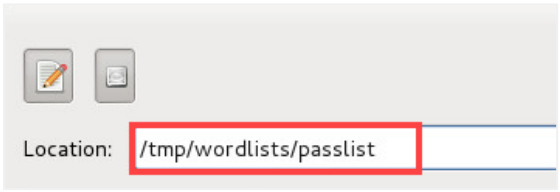
8. Notice a new *File Manager* window will appear. Click on the **File System** menu option.



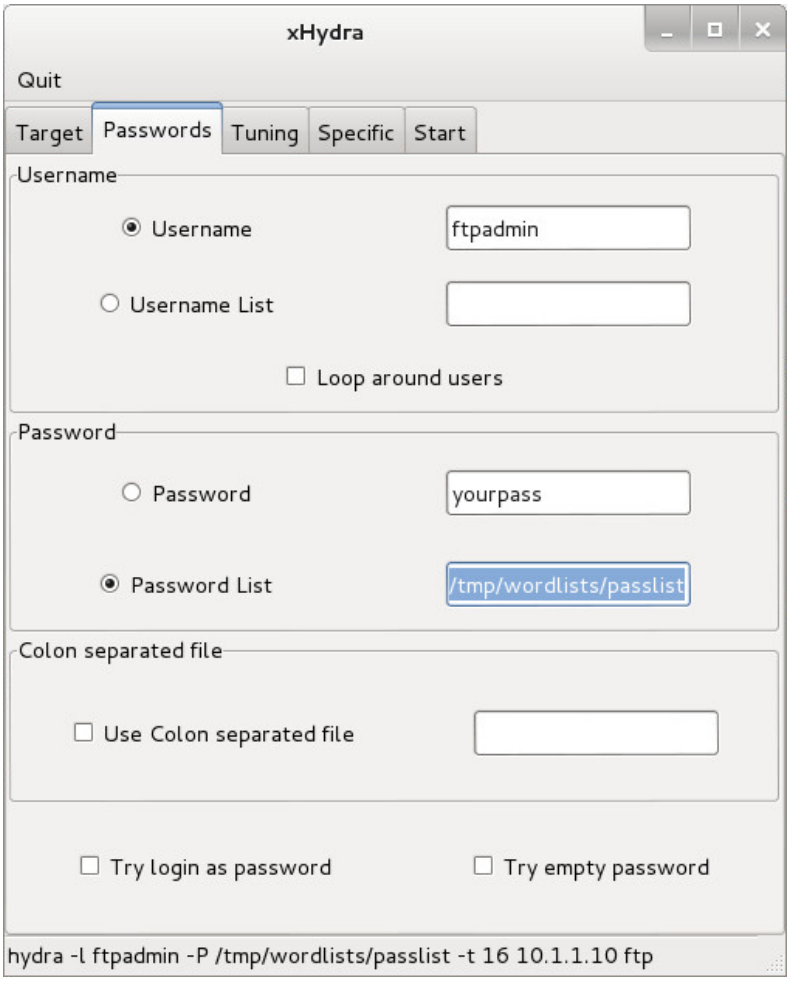
9. Click on the **Type a file name** button located in the top-left corner.



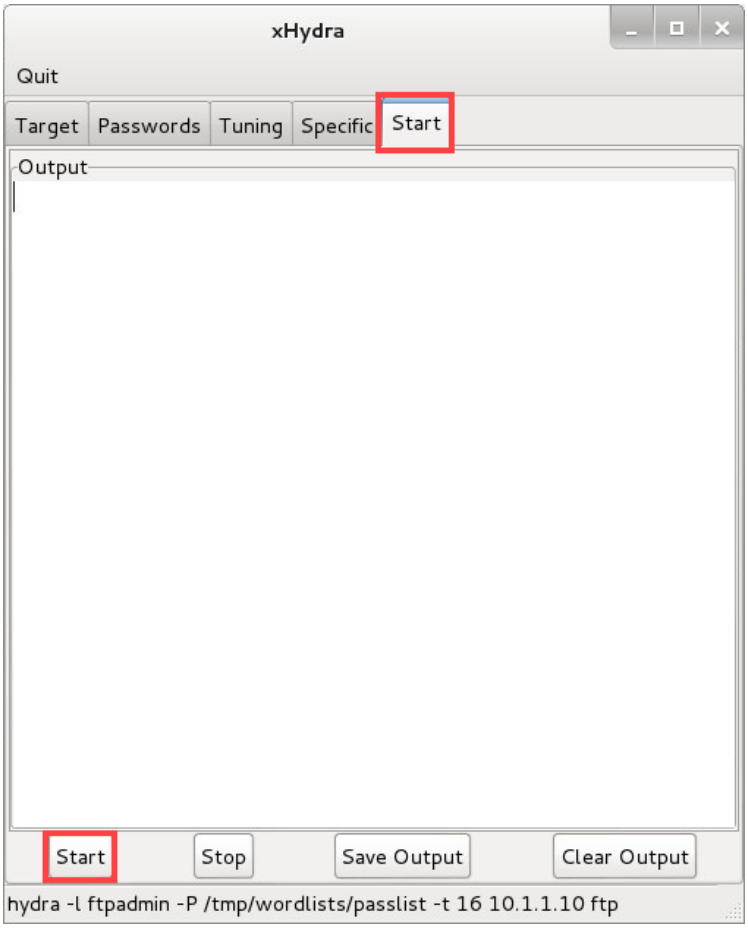
10. Type **`/tmp/wordlists/passlist`** in the *white space*. Press **Enter**.



11. Notice the *Password List* field is now populated. Verify your *HydraGTK* window displays the options as shown in the picture below.

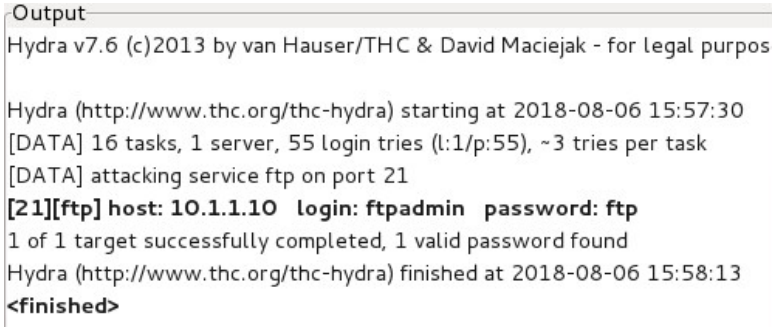


12. Click on the **Start** tab, followed by clicking on the **Start** button near the bottom of the *HydraGTK* window.



Let the scan run for about one minute.

13. Notice the program has cracked the password with a username as *ftpadmin* and a password of *ftp*.



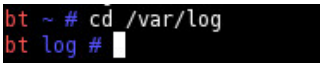
14. **Close** the program.

4.2 FTP Access Analysis

1. Change focus to the **DVL Server** and open new **terminal** window.

2. Navigate to the directory that holds the **proftpd.log** file.

```
bt~# cd /var/log
```



3. View the last 50 recorded items from the *FTP* service.

```
bt~# tail -50 proftpd.log
```

```
Aug 06 19:58:11 bt proftpd[21111] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:11 bt proftpd[21111] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 47129 usecs
Aug 06 19:58:11 bt proftpd[21111] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session closed.
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): error setting IPV6_V6ONLY: Protocol not available
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session opened.
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 361 usecs
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session closed.
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): error setting IPV6_V6ONLY: Protocol not available
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session opened.
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 26 usecs
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 240 usecs
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session closed.
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): error setting IPV6_V6ONLY: Protocol not available
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session opened.
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 19 usecs
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): error setting IPV6_V6ONLY: Protocol not available
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session opened.
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 413 usecs
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 44 usecs
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): mod_delay/0.5: delaying for 452 usecs
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session closed.
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): FTP session closed.
```

Notice the multiple failed login attempts recorded towards the end of the log file.

4. View the total amount of failed login attempt by issuing the command below (case sensitive).

```
bt~# cat proftpd.log | grep “Incorrect”
```

```
Aug 06 19:58:11 bt proftpd[21111] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21122] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21126] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21127] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
Aug 06 19:58:13 bt proftpd[21128] localhost (::ffff:203.0.113.2[::ffff:203.0.113.2]): USER ftpadmin (Login failed): Incorrect password.
```

5. The lab is now complete; you may end the reservation.

