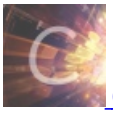
ctf中pwn题目总结

原创

博闻善行 已于 2022-02-16 20:28:55 修改 1239 收藏 11
分类专栏： 安全相关 CTF 文章标签： 安全 linux web安全
于 2020-11-16 23:25:58 首次发布



pwntools工具使用

1.安装

```
pip install pwntools （python2）
pip3 install pwntools （python3）
```

2.使用
Context设置
context是pwntools用来设置环境的功能。在很多时候，由于二进制文件的情况不同，我们可能需要进行一些环境设置才能够正常运行exp，比如有一些需要进行汇编，但是32的汇编和64的汇编不同，如果不设置context会导致一些问题。一般来说我们设置context只需要简单的一句话:

```
context(os='linux', arch='amd64', log_level='debug')
```

(1)os设置系统为linux系统，在完成ctf题目的时候，大多数pwn题目的系统都是linux
(2)**arch设置架构为amd64，可以简单的认为设置为64位的模式，对应的32位模式是’i386’**
(3) log_level设置日志输出的等级为debug，这句话在调试的时候pwntools会将完整的io过程都打印下来，使得调试更加方便。

3.数据打包
数据打包,即将整数值转换为32位或者64位地址一样的表示方式,比如0x400010表示为\x10\x00\x40一样,这使得构造payload变得很方便

用法:

- p32/p64: 打包一个整数,分别打包为32或64位
- u32/u64: 解包一个字符串,得到整数

4.接收远端传回的数据

```
recv(numb=字节大小, timeout=default) : 接收指定字节数。

recvall() : 一直接收直到达到文件EOF。

recvline(keepends=True) : 接收一行，keepends为是否保留行尾的\n。

recvuntil(delims, drop=False) : 一直读到delims的pattern出现为止。

recvrepeat(timeout=default) : 持续接收直到EOF或timeout。
```

5.向远端发送数据

```
send(data) : 发送数据。

sendline(data) : 发送一行数据，相当于在数据末尾加\n。
```

pwn题目调试

1. `checksec filename`
2.

```
context.log_level="debug"
context.terminal = ['tmux', 'splitw', '-h' ]
```

3.

```
context(arch='i386/amd64', os='linux', log_level='debug')
```

4.

```
ROPgadget --binary level3 |grep "push ebp"
ROPgadget --binary level3  --string="/bin/sh"
```

IDA使用说明

ctrl+x 查看交叉引用
ctrl+s 查看分段
ctrl+alt+k keypatch的快捷键
字母c可以将十六进制和汇编之间进行转换
右键可以将十六进制显示为十进制或者对应字符串
字符串搜索的时候要在汇编页面进行搜索

kali中文乱码

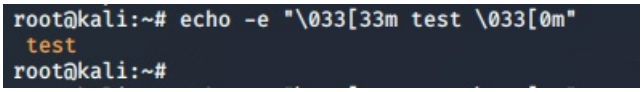
确定locales已经安装，用”apt-get install locales”命令；之后可用”locale -a”查看当前系统支持的字符集。

在命令行输入`”dpkg-reconfigure locales”`。进入图形化界面之后，（空格是选择，Tab是切换，*是选中），选中en_US.UTF-8和zh_CN.UTF-8，确定后，将`en_US.UTF-8`选为默认。
安装中文字体，`apt-get install xfonts-intl-chinese`和`apt-get install ttf-wqy-microhei`，这时发现中文乱码问题解决。

root@kali配色
未修改颜色前都是白色，容易看晕，修改方法：
vim .bashrc 进入bashrc文件修改环境变量，末尾插入

```
PS1='\[\e[31m\]\u@\h:\W#\[\e[m\] '
```

保存退出即可



libcSearch查找libc中函数地址

```
libc = LibcSearcher('write', write_addr)
libc_base=write_addr（通过函数泄露出的地址） - libc.dump('write')
binsh_addr = libc_base + libc.dump('str_bin_sh')
system_addr = libc_base + libc.dump('system')
```

Dynelf查找libc中函数地址

```
def leak(address):
    payload = 'A' * 112 + p32(writePLT) + p32(vulnAddress) + p32(1) + p32(address) + p32(4)
    p.send(payload)
    data =p.recv(4)
    return data

dynelf = DynELF(leak,elf=binary)
system_addr = dynelf.lookup("__libc_system",'libc')
```

基本做题技巧

做题

> 1.搜索字符串flag 或者/bin/sh
> 2.ctrl+x查看交叉引用
> 3.审题找洞

```
from pwn import *
context.log_level = 'debug'
context.arch = 'amd64' # 'i386'
```

将文件夹update压缩为update.tar.gz

```
tar -czvf update.tar.gz update/
```

调试命令：

```
context(os='linux', arch='amd64/i386', log_level='debug')
context.terminal = ['tmux', 'splitw', '-h']
pause()
```

gdb调试
查看变量的值print或者display

system和_system,ROP利用的时候选择_system

其他：

```
elf=ELF("./pwn-200")
write_plt=elf.plt["write"]
write_got=elf.got["write"]
write_sys=elf.sym["write"]
```

其中write_plt和write_sys是等价的

pip指定源安装

```
pip install pwntools -i  https://pypi.tuna.tsinghua.edu.cn/simple
```