【学习笔记】CTF PWN选手的养成（二）

原创

prettyX 于 2019-11-25 13:21:24 发布 1275 收藏 7
分类专栏： PWN 文章标签： PWN
版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：https://blog.csdn.net/prettyX/article/details/103230576
版权


PWN 专栏收录该内容
34 篇文章 10 订阅
订阅专栏

第三章 漏洞利用技术

课时1 栈溢出-这些都是套路

0X01 基础知识

1、寄存器：rsp/esp、rbp/ebp等

2、栈：

3、函数调用：call、ret

4、调用约定：

- __stdcall、__cdecl、__fastcall、__thiscall、__nakedcall、__pascal

5、参数传递：取决于调用约定，默认如下

- X86从右向左入栈
- X64优先寄存器，参数过多时才入栈

6、函数调用相关指令解读：

- call
- leave
- ret

0X02 栈溢出的保护机制

1、栈上的数据无法被当成指令来执行

- 数据执行保护（NX/DEP）
- 绕过方法：ROP

2、让攻击者难以找到shellcode地址

- 地址空间布局随机化（ASLR）
- 绕过方法：infoleak、ret2diresolve、ROP

3、检测Stack Overflow

- Stack Canary/Cookie
- 绕过方法：infoleak

4、现在NX+Stack Canary+ASLR基本是标配

0X03 栈溢出的利用方法

1、现代栈溢出利用技术基础：ROP

- 一种代码复用技术，通过控制栈调用来劫持控制流
- Google 关键字：Ret2libc、ROP
- CTF中ROP常规套路：（1）第一次触发漏洞，通过ROP泄露libc的address（如puts_got），计算system地址，然后返回到一个可以重现触发漏洞的位置（如main），再次触发漏洞，通过ROP调用system("/bin/sh")。（2）直接execve("/bin/sh",["/bin/sh"],NULL)，通常在静态链接时比较常用
- 习题：

1. Defcon 2015 Qualifier：R0pbaby（强烈推荐）
2. AliCTF 2016：vss
3. PlaidCTF 2013：ropasaurusrex

- 作业：

1. 根据r0pbaby的writeup重写EXP
2. 尝试做一下vss和ropasaurusrex

2、利用signal机制的ROP技术：SROP

- SROP：Sigreturn Oriented Programming
- 系统Signal Dispatch之前会将所有寄存器压入栈，然后调用signal handler,signal handler返回时会将栈的内容还原到寄存器。
- 如果事先填充栈，然后直接调用signal return，那在返回的时候就可以控制寄存器的值。
- 用的不是特别多，但是有时候很好用，推荐资料

1. http://angelboy.logdown.com/posts/283221-srop
2. http://www.2cto.com/article/201512/452080.html

- 例题：

1. Defcon 2015 Qualifier fuckup（这题比较难）
2. 建议自己写一个Demo自己测试

3、没有binary怎么办：BROP

- BROP：Blind Return Oriented Programming，盲打
- 目标：在拿不到目标binary的条件下进行ROP
- 条件：必须先存在一个已知的stack overflow的漏洞，而且攻击者知道如何触发这个漏洞；服务器进程在crash之后会重新复活，并且复活的进程不会被re-rand
- 用的不是特别多，但是在CTF中出现过
- 推荐资料：

1. http://ytliu.info/blog/2014/05/31/blind-return-oriented-programming-brop-attack-yi/
2. http://ytliu.info/blog/2014/06/01/blind-return-oriented-programming-brop-attack-er/

- 例题：HCTF 2016 出题人跑路了（pwn50）

4、劫持栈指针：stack pivot（很重要）

- 将栈劫持到其他攻击者控制的缓冲区

1. 向目标缓冲区填入栈数据（如ROP Chains），然后劫持esp到目标缓冲区。劫持esp的方法有很多，最常用的就是ROP

时利用可以直接改写esp的gadget，如pop esp，ret；

2. 是一种相对常用的利用技术，不仅用于栈溢出，也可以用在其他可以劫持控制流的漏洞。

- Stack Piovt的动机

1. 溢出字节数有限，无法完成ROP
2. 栈地址未知且无法泄露，但是某些利用技术却要求知道栈地址（ret2 dlresolve）
3. 劫持esp到攻击者控制的区域，也就变相的控制了栈中的数据，从而可以使非栈溢出的控制流劫持攻击也可以做ROP

- Stack Pivot的利用条件：

1. 存在地址已知且内容可控的buffer：（1）bss段，由于bss段尾端通常具有很大的空余空间（pagesize-usedsize），所以bss段段尾端也往往是stack pivot的目标；（2）堆块，如果堆地址已泄且堆上的数据可被控制，那堆也可以作为stack pivot的目标

- 控制流可劫持
- 存在劫持栈指针的gadgets:如pop esp，ret，除此之外还有很多，要具体binary具体分析
- 作业：

1. EKOPARTY CTF 2016 fuckzing-exploit-200（基于栈溢出的stack pivot，必做作业）
2. HACKIM CTF 2015-Exploitation 5（基于堆溢出的stack pivot，选做作业）

5、利用动态链接绕过ASLR：ret2dl resolve、fake linkmap

- 动态链接的过程就是从函数名到函数地址转换的过程，所以我们可以通过动态链接器来解析任何函数，且无需任何leak
- 前置技能：了解动态链接的过程

1. http://blog.chinaunix.net/uid-24774106-id-3053007.html
2. 《程序员的自我修养》

- 伪造动态链接的相关数据结构如linkmap、relplt，详见以下内容：

1. http://rk700.github.io/2015/08/09/return-to-dl-resolve
2. http://angelboy.logdown.com/posts/283218-return-to-dl-resolve
3. http://www.inforsec.org/wp/?p=389

- 理论上任何可以stack pivot且FULLRELRO未开的题目都可以利用这种技术，所以可以试试用这种技术去做一些之前的习题。
- 习题：

1. Codegate CTF Finals 2015 yocto(fake relplt) Http://o0xmuhe.me/2016/10/25/yocto-writeup/
2. HITCON QUALS CTF2015 readable(fake linkmap)
3. Hack.lu's 2015 OREO http://wapiflapi.github.io/2014/11/17/hacklu-oreo-with-ret2dl-resolve/

6、利用地址低12bit绕过ASLR：Partial Overwrite

- 在PIE开启的情况下，一个32地址的高20bit会被随机化，但是低12bit是不变的。所以可以通过只改写低12bit来绕过PIE，不仅在栈溢出使用，在各种利用都经常使用。（基本带PIE的题，每道都要用到这个技术）
- 作业：了解Partiaal Overwrite http://ly0n.me/2015/07/30/bypass-aslr-with-partial-eip-overwrite
- 习题：HCTF 2016 fheap（基于堆溢出的parital overwrite）（非常推荐）

7、绕过stack canary：改写指针与局部变量、leak canary、overwrite canary

- 至此所讲的所有套路，一旦遇到Stack Canary均无法使用！！可以说是栈溢出的大杀器。
- 绕过思路：

1. 不覆盖Stack Canary，只覆盖Stack Canary前的局部变量、指针。（1）已经几乎不可行，因为编译器会根据占用内存大

小从小到大排列变量。（2）但是在某些情况下依然可用。

2. Leak Canary：可以通过printf泄露，Canary一般从00开始。主要是这种方法。
3. Overwrite Canary:Canary在TLS，TLS地址被随机化。这种方法用的不是特别多。

8、溢出位数不够怎么办：覆盖ebp、partial overwrite

- 习题：XMAN 2016 广外女生-pwn、Codegate CTF Finals 2015,chess