pwnable.kr collision - 3 pt [writeup]

原创

普通网友 于 2019-07-24 13:22:22 发布 706 收藏

分类专栏： 你们的pwn 文章标签： pwn pwntools pwnable collision CTF

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/youshaoduo/article/details/97122935

版权

你们的pwn 专栏收录该内容

3 篇文章 0 订阅

订阅专栏

```
Daddy told me about cool MD5 hash collision today.
I wanna do something like that too!

ssh col@pwnable.kr -p2222 (pw:guest)
```

常规操作，先到服务器上看一下这是个什么东西：

```
➜  ~ ssh col@pwnable.kr -p 2222
col@pwnable.kr's password:

 ____  __    __  ____    ____  ____   _        ___      __  _  ____
|    \|  |__|  ||    \  /    ||    \ | |      /  _]    |  |/ ]|    \
|  o  )  |  |  ||  _  ||  o  ||  o  )| |     /  [_     |  ' / |  D  )
|   _/|  |  |  ||  |  ||     ||     || |___ |    _]    |    \ |    /
|  |  |  `  '  ||  |  ||  _  ||  O  ||     ||   [_ __ |     \|    \
|  |   \      / |  |  ||  |  ||     ||     ||     ||  ||  .  ||  .  \
|__|    \_/\_/  |__|__||__|__||_____||_____||_____||__||__|\_||__|\_|

- Site admin : daehee87.kr@gmail.com
- IRC : irc.netgarage.org:6667 / #pwnable.kr
- Simply type "irssi" command to join IRC now
- files under /tmp can be erased anytime. make your directory under /tmp
- to use peda, issue `source /usr/share/peda/peda.py` in gdb terminal
Last login: Wed Jul 24 01:06:46 2019 from 116.233.190.55
col@prowl:~$ ls
col  col.c  flag
col@prowl:~$ cat col.c
#include <stdio.h>
#include <string.h>
unsigned long hashcode = 0x21DD09EC;
unsigned long check_password(const char* p){
 int* ip = (int*)p;
 int i;
 int res=0;
 for(i=0; i<5; i++){
  res += ip[i];
 }
 return res;
}

int main(int argc, char* argv[]){
 if(argc<2){
  printf("usage : %s [passcode]\n", argv[0]);
  return 0;
 }
 if(strlen(argv[1]) != 20){
  printf("passcode length should be 20 bytes\n");
  return 0;
 }

 if(hashcode == check_password( argv[1] )){
  system("/bin/cat flag");
  return 0;
 }
 else
  printf("wrong passcode.\n");
 return 0;
}
col@prowl:~$
```

分析代码，首先映入眼帘的是**hashcode = 0x21DD09EC**，他是个16进制的数字

往下看，发现这里的函数很有意思，只循环了5次。

```
unsigned long check_password(const char* p){
 int* ip = (int*)p;
 int i;
 int res=0;
 for(i=0; i<5; i++){
  res += ip[i];
 }
 return res;
}
```

继续往下看，会告诉你**passcode length should be 20 bytes**，这就不难理解为什么上面的函数只循环了5次。

因为20byte=5int，一个int占4byte。

这道题也就变成了，**输入20个char，如何让这20个char变成5个int，这5个int加起来等于0x21DD09EC**
这TM就是一道数学题啊。

那么思路很明显了。20个字符要用16进制编码。那么payload就要这么写：

```
#!/usr/bin/python

from pwn import *

# str1 = '\x00\x00\x00\x00'+'\x00\x00\x00\x00'+'\x00\x00\x00\x00'+'\x00\x00\x00\x00'+'\xEC\x09\xDD\x21'
str2 = '\x01\x01\x01\x01'+'\x01\x01\x01\x01'+'\x01\x01\x01\x01'+'\x01\x01\x01\x01'+'\xE8\x05\xD9\x1D'
str3 = p32(0x01010101)*4 + p32(0x1DD905E8) #str2的另一种写法，偷懒写法

s = ssh(host='pwnable.kr', port=2222, user='col', password='guest')
s.connected()
cn = s.process(argv=['col', str2], executable='./col')
print cn.recv()
```

一开始我想这么简单的算术题，直接4个0加0x21DD09EC不就好了，直接上str1（这里没考虑用除法的原因是hashcode这玩意一看就不能被5整除，多个了小数，麻烦得很）。然而真实的情况确是：

```
[ERROR] Inappropriate nulls in argv[1]: '\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xec\t\
xdd!'
Traceback (most recent call last):
  File "/Users/youssef/Desktop/pwn/collision.py", line 9, in <module>
    cn = s.process(argv=['col', str1], executable='./col')
  File "/usr/local/Cellar/pwntools/3.12.2/libexec/lib/python2.7/site-packages/pwnlib/tubes/ssh.py", line 818, in
 process
    self.error('Inappropriate nulls in argv[%i]: %r' % (i, arg))
  File "/usr/local/Cellar/pwntools/3.12.2/libexec/lib/python2.7/site-packages/pwnlib/log.py", line 417, in error
    raise PwnlibException(message % args)
pwnlib.exception.PwnlibException: Inappropriate nulls in argv[1]: '\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\xec\t\xdd!'
```

好吧，\x00这玩意是null的编码，那就前面+4s，后面-4s。于是就出来了str2。

至于后面的 **\xE8\x05\xD9\x1D** 其实是0x1DD905E8的小端写法。

一般这种不可能给你出大端题，所以直接上小端，不用多想了。当然如果你懒得算小端数是啥，pwntools里面自带了小端转换器，可以将数字转换成小端的字符串。也就是str3的写法，用p32函数。同样的，还有p16，p64这样的函数，p32 转换4bit. p64 和 p16 则分别转换 8 bit 和 2 bit 数字。

大端和小端啥区别可以自己查一下，反正就是一个正着看，一个反着看。iOS是小端。

最后的结果：

```
➜  pwn /usr/local/bin/python /Users/youssef/Desktop/pwn/collision.py
[+] Connecting to pwnable.kr on port 2222: Done
[*] fd@pwnable.kr:
    Distro    Ubuntu 16.04
    OS:       linux
    Arch:     amd64
    Version:  4.4.179
    ASLR:     Enabled
[+] Starting remote process './col' on pwnable.kr: pid 59553
daddy! I just managed to create a hash collision :)
```