Cookie 崩溃分析报告



THINCT
December 11, 2023

什么是 cookie 崩溃

分析方法

🏿 崩溃复现

```
=#include <stdio.h>
     #include <stdlib.h>
    □void func0(char buf[])
 6
          for (unsigned int i = 0; i < 300; i++)
              buf[i] = i % 256;
 9
12
    ∃void func1(int* pArr)
13
         char buf[32]{0};
14
15
         printf("%x\n", *pArr);
16
         func0(buf);
18
19
    □int main(int argc, char* argv[])
21
          int* pArr = new int(0x123);
22
          func1 (pArr);
23
2.4
          printf("%x\n", *pArr);
25
26
         system("pause");
2.7
         return 0;
28
```

Figure 1: 源码构造浮现过程

崩溃复现



Figure 2: 崩溃结果

什么是 cookie 崩溃

分析方法

Magazina cookie 校验

堆栈(Stack)和堆(Heap)的 Cookie 安全检查是一种用于防止缓冲区溢出攻击的技术。这种技术的目的是通过引入一种额外的随机值(通常称为 Cookie)来增加攻击者预测攻击的难度,从而提高系统的安全性。

堆栈 Cookie 安全检查: 在堆栈 Cookie 安全检查中,通常是在函数的栈帧中添加一个随机生成的值,称为堆栈 Cookie 或 Canary。这个 Cookie 位于函数局部变量和返回地址之间,当函数返回时,会检查这个 Cookie 是否被破坏。

攻击者如果试图通过溢出缓冲区来修改返回地址,他们也必须修改堆栈 Cookie。由于 Cookie 的值是不可预测的,攻击者需要事先知道这个 Cookie 的值才能成功执行攻击。因此,堆栈 Cookie 提供了一种防范缓冲区溢出攻击的机制。

堆 Cookie 安全检查: 堆 Cookie 安全检查与堆栈 Cookie 类似,但是是在动态内存分配的堆上进行的。在动态内存分配 时,为分配的内存块添加了一个额外的 Cookie,并将这个 Cookie 存储在内存块的头部或尾部。当释放内存时,系统 会检查 Cookie 是否被破坏,从而检测出潜在的堆溢出。

实现原理:添加 Cookie:

在函数调用或内存分配时,系统生成一个随机的 Cookie 值,并将其添加到相关数据结构中(堆栈帧或动态内存块)。 检查 Cookie:

在函数返回或内存释放时,系统会检查 Cookie 是否被破坏。如果 Cookie 被破坏,系统就会认为发生了缓冲区溢出,从而触发相应的安全措施。增加攻击者难度:

由于 Cookie 是随机生成的,攻击者必须在进行攻击时预测正确的 Cookie 值,增加了攻击的难度。

cookie 校验

```
void func1(int* pArr)
00261000
           push
                        ebp
 002610C1
           mov
                        ebp, esp
 002610C3 sub
                        esp,24h
 00261006
                        eax, dword ptr [ security cookie (0263004h)]
           mov
 002610CB xor
                        eax, ebp
002610CD
                        dword ptr [ebp-4],eax
           mov
     char buf[32]{0};
 002610D0
           xor
                寄存器
 002610D2
           mov
                EAX = B247866C EBX = 010EB000 ECX = 01829D68 EDX = 01829D68
 002610D5
           mov
                   ESI = 0182B458 EDI = 0182BD70 EIP = 002610CD
 002610D8
           mov
                  ESP = 012FF788 EBP = 012FF7AC EFL = 00000286
 002610DB
           mov
 002610DE
           mov
                0 \times 012 FF7A8 = 00000004
 00261001
```

Figure 3: cookie 起点

Magazina cookie 校验



Figure 4: cookie 终点

结论:

参与 cookie 相关的异或运算的两个值,ebp 没有变化,另外 eax 和 ecx 在正常情况下应该是相同的,如果不相等,则说明堆栈出现破坏了.

什么是 cookie 崩溃

分析方法

🏴 观察存放 cookie 抑或结果的内存

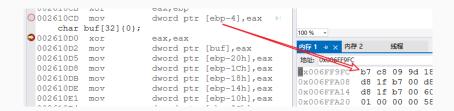


Figure 5: 记住 ebp-4 所指向的内存的值

观察存放 cookie 抑或结果的内存

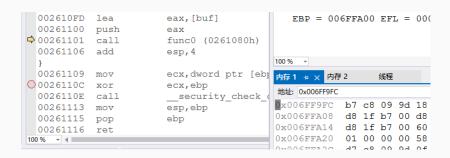


Figure 6: 继续记住 ebp-4 所指向的内存的值

🏴 观察存放 cookie 抑或结果的内存

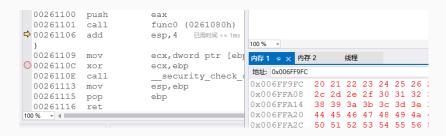


Figure 7: 直到发生变化

观察存放 cookie 抑或结果的内存

总结:

只要这个哨兵位置的内存发生变化, 就是发生了堆栈溢出.

什么是 cookie 崩溃

分析方法



调试类似堆栈破坏的难点在哪里呢?