CVE-2018-4338：在MACOS上通过BROADCOM AIRPORT KEXT进行信息泄露

一叶飘零 / 2018-10-25 10:37:00 / 浏览数 2845

原文地址

`https://www.zerodayinitiative.com/blog/2018/10/24/cve-2018-4338-triggering-an-information-disclosure-on-macos-through-a-broadc`

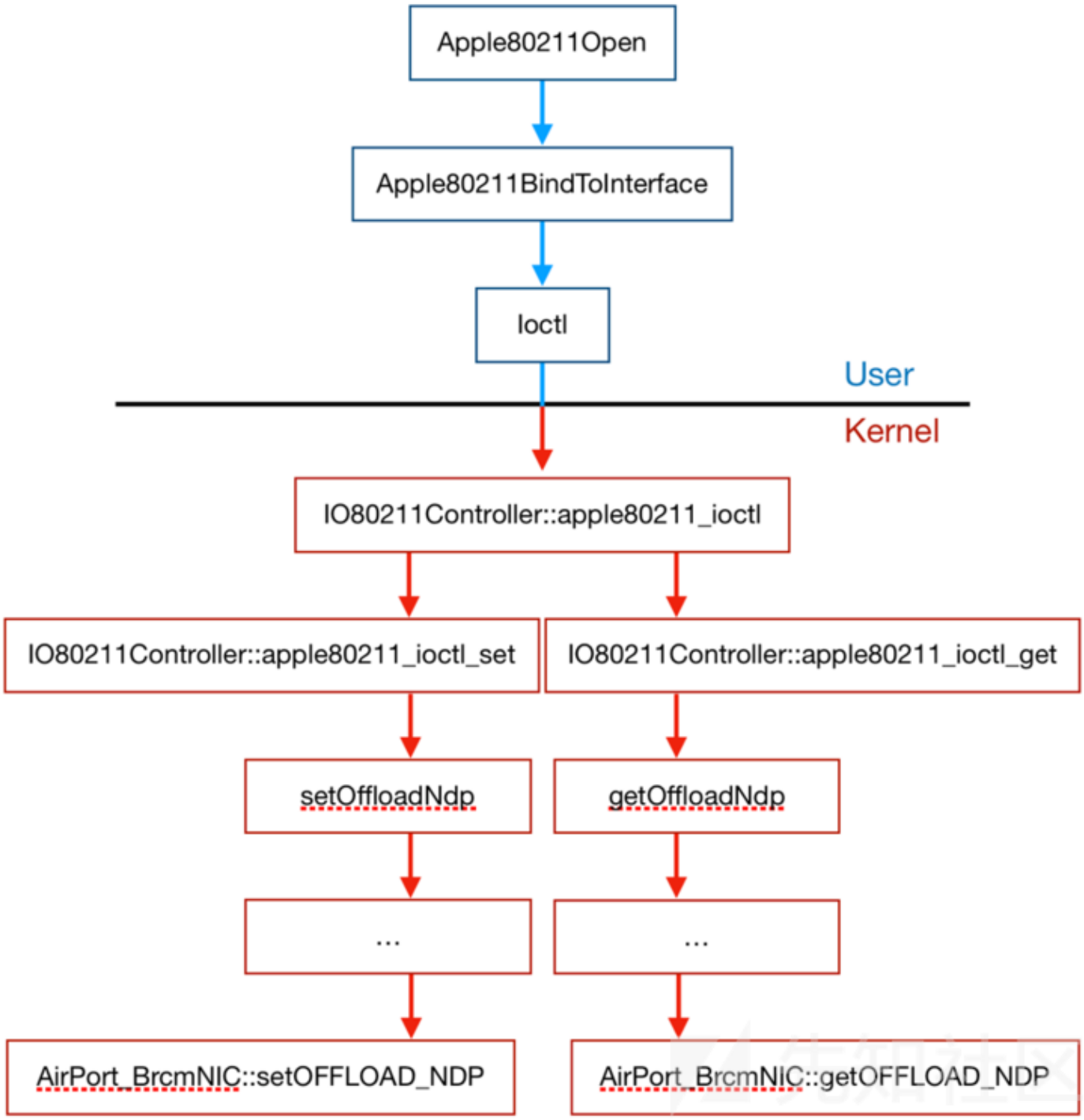
前言

在用户状态下执行提升权限的代码是利用漏洞的一个关键，这个原则也同样适用于Pwn2Own入门、提交TIP或者应对一些实时的恶意软件。为了逃逸最新的沙盒，攻击者和

最近，向ZDI程序提交的一项报告显示了macOS如何做到只有这种类型的信息泄露漏洞。延世大学的`Lee @ SECLAB`报道称，该漏洞存在于处理`Broadcom AirPort kext`的过程中，如果您不熟悉它们，那么可以这么说，kext文件（内核扩展的简称）就是macOS的驱动程序，它就类似于Windows中的DLL。既然有了补丁，Lee的写作和

安装程序

这个特定漏洞仅适用于开启了Wi-Fi状态的系统，虽然这种状态在通常情况下是无处不在的，但有时它也可能会被禁用。为了解决这个问题，可以使用脚本打开Wi-Fi。攻击者



漏洞

简而言之，此漏洞使得攻击者可以获取Apple OS
X局部环境中的提升权限的内核地址，它的根本原因在于AirPort.BrcmNIC.kext，它不会检查输入值，并且它会导致Out Of Bounds■OOB■。

发生这个错误是因为setOFFLOAD_NDP函数不检查输入值，这意味着OOB
Read的堆栈值存储在ol_nd_hostip变量中，攻击者可以使用getOFFLOAD_NDP函数读取ol_nd_hostip变量。

```
signed __int64 __fastcall AirPort_BrcmNIC::setOFFLOAD_NDP(__int64 a1, __int64
a2, __int64 a3)
{
…

    inp = a3;
    …
    if ( *(inp + 4) )
    {
        v7 = 0;
        do
            (*(*a1 + 0xDF0LL))(a1, "ol_nd_hostip", 0LL, 0LL, inp + 0x10LL * v7++ +
8, 16LL, 1LL, 0LL);
        while ( v7 < *(inp + 4) );
    }
    …
}
```

漏洞利用

既然我们知道`AirPort_BrcmNIC::setOFFLOAD_NDP`函数中发生了错误，我们只需要获取存储在`ol_nd_hostip`变量中的值。再看一下需要删除的函数：

```
signed __int64 __fastcall AirPort_BrcmNIC::setOFFLOAD_NDP(__int64 a1, __int64
a2, __int64 a3)
{
…

    inp = a3;
    …
    if ( *(inp + 4) )
    {
        v7 = 0;
        do
            (*(*a1 + 0xDF0LL))(a1, "ol_nd_hostip", 0LL, 0LL, inp + 0x10LL * v7++ +
8, 16LL, 1LL, 0LL);
        while ( v7 < *(inp + 4) );
    }
    …
}
```

从`setOffloadNdp`函数中可以看出，inp是该函数的局部变量。所述`old_nd_hostip`变量存储了数值高达`0x40`的字节，并且如果该值是0，它就可以被矫正。因此，由于`+ 0x68`，因此它可以通过运行循环七次被保存，然后，您就可以使用该`getOffloadNdp`函数泄漏堆栈值。

```
__int64 __fastcall setOffloadNdp(IO80211Controller *this, IO80211Interface *a2,
IO80211VirtualInterface
*a3, __int64 a4)
{
  unsigned __int64 user_var;          // rsi
  int inp;                            // [rsp+0h] [rbp-60h]

  …
  if ( *(a4 + 24) == 72 )
  {
    user_var = *(a4 + 32);
    if ( v7 )
    {
      result = IO80211Controller::copyIn(this, user_var, &inp, 0x48uLL);
      if ( !result )
      {
        if ( v4 )
          result = IO80211Controller::apple80211VirtualRequestIoctl(this,
0x802869C8, 192, v4, &inp);
        else
          result = IO80211Controller::apple80211RequestIoctl(this,
0x802869C8, 192, v5, &inp);
      }
    }
  }
return result;
```

执行时，概念证明（PoC）代码应产生以下输出：

```
[redacted]-mini:setOffload_NDP [redacted]$ uname -a
Darwin [redacted]-mini.local 17.4.0 Darwin Kernel Version 17.4.0: Sun Dec 17 09:19:54 PST 2017; r
oot:xnu-4570.41.2~1/RELEASE_X86_64 x86_64
[redacted]-mini:setOffload_NDP [redacted]$ ./test
[+]Prepare
ifname: en1
p[1] = 0x8
set Result : 0
get Result : 0
01 00 00 00 04 00 00 00 A1 53 79 9D 7F FF FF FF
00 80 00 B9 81 FF FF FF E8 BA 71 0A 92 FF FF FF
90 B9 71 0A 92 FF FF FF B6 7D 79 9D 7F FF FF FF
0C 7D 79 9D 01 00 00 00 00 80 00 B9 81 FF FF FF
C8 69 28 80 00 00 00 00
[+] io80211_base : 0xFFFFFF7F9D743000
[+] kslide : 0x1BBF3000
[redacted]-mini:setOffload_NDP [redacted]$
```

结论

如果您希望自己测试一下，PoC应该适用于macOS
10.13及之前的版本。虽然这不能用于自动执行代码，但攻击者可以利用它与其他漏洞一起在内核的中执行代码。此外，它还表明找到正确的内存位置是漏洞链的重要组成部

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板