title: PassiveFuzz框架试用与调试小记
date: 2016-11-21 22:22:12

categories: [OS X]

tags: [XNU]

0x00 摘要

前段时`TREND`的研究员在`POC`安全大会上介绍了一款`OS X`系统上面的`Fuzz`框架，并且开源了。`github`点这里。我记得看雪好像已经有翻译成中文版的PPT，找了一下没找到:-(。网上能找到的信息，这里就不复述了，有兴趣的读者可以自行阅读。

简单来说就是通过内核扩展，对一系列内核函数进`inline hook`，在这些被`hook`的函数被调用的时候，对参数进行一些随机的修改。所以叫做被动式Fuzz，而不是传统的主动调用内核函数传入不同的参数。

0x01 XNU内核函数符号

因为是通过编写`XNU`内核扩展的实现的`inline hook`构建的整套`Fuzz`的框架，内核的符号表是一个非常重要的事情，同时在编写一些内核漏洞的利用时，内核的符号表同样的重要。这里简单的介绍一下思路和流程，具体的思路网上已经有很详细的叙述，有兴趣的读者可以自行阅读相关的技术细节[1] [2] 。

- 读取内核文件的Mach-O头部
- 从`__TEXT`、`__LINKEDIT`的`segment`中获取相关信息
- 从`LC_SYMTAB`获取`nlist`数据结构
- 结合获取到的信息，就可以计算出需要的内核函数在运行时的地址了。

相关的源码也可以看到，不过和我们fuzz的核心逻辑关系不大，可以看到源码的`main.c`文件。

```
kern_return_t
the_flying_circus_start(kmod_info_t * ki, void *d)
{
    kern_return_t kr= KERN_SUCCESS;

#if DEBUG
    //moony_modify//printf("[DEBUG] Starting the circus
the_flying_circus_start...\n");
#endif
    // read kernel from filesystem and initialize a kernel_info structure
    if(init_kernel_info(&g_kernel_info)) return KERN_FAILURE; <--这里做的事情就
是解决内核符号

    // if zombie mode is activated, all the rootkit features will be activated
    // inside the zombie thread and we just return failure here
#if ZOMBIE_MODE > 0
    // disable logging features to avoid error messages from kext failure to
load
    //disable_oskextlog();
    //disable_kextd_syslog();
    // activate the zombie
    //unleash_the_zombie();
    //return KERN_FAILURE;
#endif

    /*...*/ //rootkit相关的功能，与我们的fuzz逻辑无关，不需要关注

    //@flyic moony_li@trendmicro.com 2015-06-17
    kr |= init_inline_hook();                <-- 初始化inline hook的信息
    kr |= install_inline_hook();              <-- inline hook 具体实施
    //set_kernel_panic_hook();//This way for monior panic dump does not work!
    //moony_modify//printf("[DEBUG] the_flying_circus_start done...\n");
    //init_collect_log();

    return KERN_SUCCESS;
}
```

0x02 inline hook的实现

这里简单的阅读一下源码，看看`inline hook`的具体实现。最核心的函数如下：

```
kern_return_t
install_trampoline_any(mach_vm_address_t patch_addr, mach_vm_address_t dest_address, void *orig_bytes)
{
    char trampoline[12] = "\x48\xB8\x00\x00\x00\x00\x00\x00\x00\x00" // mov rax, address
    "\xFF\xE0"; // jmp rax

    //mach_vm_address_t patch_addr = solve_kernel_symbol(&g_kernel_info, symbol);
    if (patch_addr == 0)
    {
#if DEBUG
        //moony_modify//printf("[ERROR] patch_addr[0x%x] is not valid [%s]\n", patch_addr, __FUNCTION__);
#endif
        return KERN_FAILURE;
    }
    // store the original bytes in user provided buffer
    memcpy(orig_bytes, (void*)patch_addr, sizeof(trampoline));
    // XOR the original bytes
    //for (int i = 0; i < TRAMPOLINE_SIZE; i++)
    //    ((char*)orig_bytes)[i] ^= XOR_KEY;
    // set the target address
    memcpy(trampoline+2, &dest_address, sizeof(mach_vm_address_t));
    // patch the target address with the trampoline
    disable_interrupts();
    disable_wp();
    memcpy((void*)patch_addr, trampoline, sizeof(trampoline));
    enable_wp();
    enable_interrupts();
    return KERN_SUCCESS;
}
```

实现逻辑十分简单：

- 解决函数符号之后，获取目标函数的地址
- 将目标地址处的代码替换成跳转的到我们自己实现的函数的代码

在`install_trampoline_any`处下断点。

```
* thread #1: tid = 0x0001, 0xffffff7f8ba5ff02
pasive_kernel_fuzz`install_trampoline_any(patch_addr=18446743524117703248,
dest_address=18446743522001688720, orig_bytes=0xffffff800e9aba80) + 34 at
hijacking_utils.c:271, stop reason = breakpoint 1.1
    frame #0: 0xffffff7f8ba5ff02
pasive_kernel_fuzz`install_trampoline_any(patch_addr=18446743524117703248,
dest_address=18446743522001688720, orig_bytes=0xffffff800e9aba80) + 34 at
hijacking_utils.c:271
   268  kern_return_t
   269  install_trampoline_any(mach_vm_address_t patch_addr, mach_vm_address_t
dest_address, void *orig_bytes)
   270  {
-> 271      char trampoline[12] = "\x48\xB8\x00\x00\x00\x00\x00\x00\x00\x00" //
mov rax, address
   272      "\xFF\xE0"; // jmp rax
   273
   274      //mach_vm_address_t patch_addr =
solve_kernel_symbol(&g_kernel_info, symbol);
```

查看`patch_addr`和`dest_address`

```
kernel.development`ipc_kmsg_get:
    0xffffff8009c5ea50 <+0>:  pushq  %rbp
    0xffffff8009c5ea51 <+1>:  movq   %rsp, %rbp
    0xffffff8009c5ea54 <+4>:  pushq  %r15
    0xffffff8009c5ea56 <+6>:  pushq  %r14
    0xffffff8009c5ea58 <+8>:  pushq  %r13
    0xffffff8009c5ea5a <+10>: pushq  %r12
    0xffffff8009c5ea5c <+12>: pushq  %rbx
    0xffffff8009c5ea5d <+13>: subq   $0x28, %rsp
    0xffffff8009c5ea61 <+17>: movq   %rdx, %r14
    0xffffff8009c5ea64 <+20>: movl   %esi, %r13d
    0xffffff8009c5ea67 <+23>: movq   %rdi, %r15
(lldb) dis -s dest_address
pasive_kernel_fuzz`trampline_ipc_kmsg_get:
    0xffffff7f8ba61890 <+0>:  pushq  %rbp
    0xffffff7f8ba61891 <+1>:  movq   %rsp, %rbp
    0xffffff7f8ba61894 <+4>:  subq   $0x80, %rsp
    0xffffff7f8ba6189b <+11>: movq   %rdi, -0x8(%rbp)
    0xffffff7f8ba6189f <+15>: movl   %esi, -0xc(%rbp)
    0xffffff7f8ba618a2 <+18>: movq   %rdx, -0x18(%rbp)
    0xffffff7f8ba618a6 <+22>: nop
    0xffffff7f8ba618a7 <+23>: nop
    0xffffff7f8ba618a8 <+24>: nop
    0xffffff7f8ba618a9 <+25>: nop
```

分别是我们的目标函数和`hook`函数。

执行到函数执行完之后。

```
* thread #1: tid = 0x0001, 0xffffff7f8ba5ff88 pasive_kernel_fuzz`install_trampoline_any(patch_addr=18446743524117703248, dest_address=18446743522001688720, orig_bytes=0xffffff800e9aba80) + 168 at hijacking_utils.c:295, stop reason = step over
    frame #0: 0xffffff7f8ba5ff88 pasive_kernel_fuzz`install_trampoline_any(patch_addr=18446743524117703248, dest_address=18446743522001688720, orig_bytes=0xffffff800e9aba80) + 168 at hijacking_utils.c:295
   292      memcpy((void*)patch_addr, trampoline, sizeof(trampoline));
   293      enable_wp();
   294      enable_interrupts();
-> 295      return KERN_SUCCESS;
   296  }
   297
   298
```

我们再次观察`patch_addr`和`dest_address`

```
kernel.development`ipc_kmsg_get:
    0xffffff8009c5ea50 <+0>:  movabsq $-0x807459e770, %rax      ; imm = 0xFFFFFF7F8BA61890
    0xffffff8009c5ea5a <+10>: jmpq   *%rax
    0xffffff8009c5ea5c <+12>: pushq  %rbx
    0xffffff8009c5ea5d <+13>: subq   $0x28, %rsp
    0xffffff8009c5ea61 <+17>: movq   %rdx, %r14
    0xffffff8009c5ea64 <+20>: movl   %esi, %r13d
    0xffffff8009c5ea67 <+23>: movq   %rdi, %r15
(lldb) dis -s dest_address
pasive_kernel_fuzz`trampline_ipc_kmsg_get:
    0xffffff7f8ba61890 <+0>:  pushq  %rbp
    0xffffff7f8ba61891 <+1>:  movq   %rsp, %rbp
    0xffffff7f8ba61894 <+4>:  subq   $0x80, %rsp
    0xffffff7f8ba6189b <+11>: movq   %rdi, -0x8(%rbp)
    0xffffff7f8ba6189f <+15>: movl   %esi, -0xc(%rbp)
    0xffffff7f8ba618a2 <+18>: movq   %rdx, -0x18(%rbp)
    0xffffff7f8ba618a6 <+22>: nop
    0xffffff7f8ba618a7 <+23>: nop
    0xffffff7f8ba618a8 <+24>: nop
    0xffffff7f8ba618a9 <+25>: nop
```

可以看到目标函数已经成功的被`inline hook`了。

0x03 执行pasive_kel_fuzz.kext

`github`中的源码克隆下来之后会有提前编译好的二进制文件，用了之后没啥反应也不知道为什么，我直接用了自己编译的内核扩展。

3.1 启动参数

因为我没有两台机器用的是虚拟机，所以不能使用文档中推荐的系统启动参数，我使用的启动参数是这样的

```
➜  Desktop sudo nvram boot-args
Password:
boot-args	debug=0x141 kext-dev-mode=1 kcsuffix=development pmuflags=1 -v
```

本来我使用的启动参数是

```
boot-args	debug=0xd66 _panicd_ip=xx.xx.xx.xx kext-dev-mode=1
kcsuffix=development pmuflags=1 -v
```

但是崩溃的`core`文件看不到调用栈，只能看到崩溃的一层栈，也不知道是为什么。所以使用在系统启动时使用`lldb` attach 上去，可以看到崩溃时的完整栈。

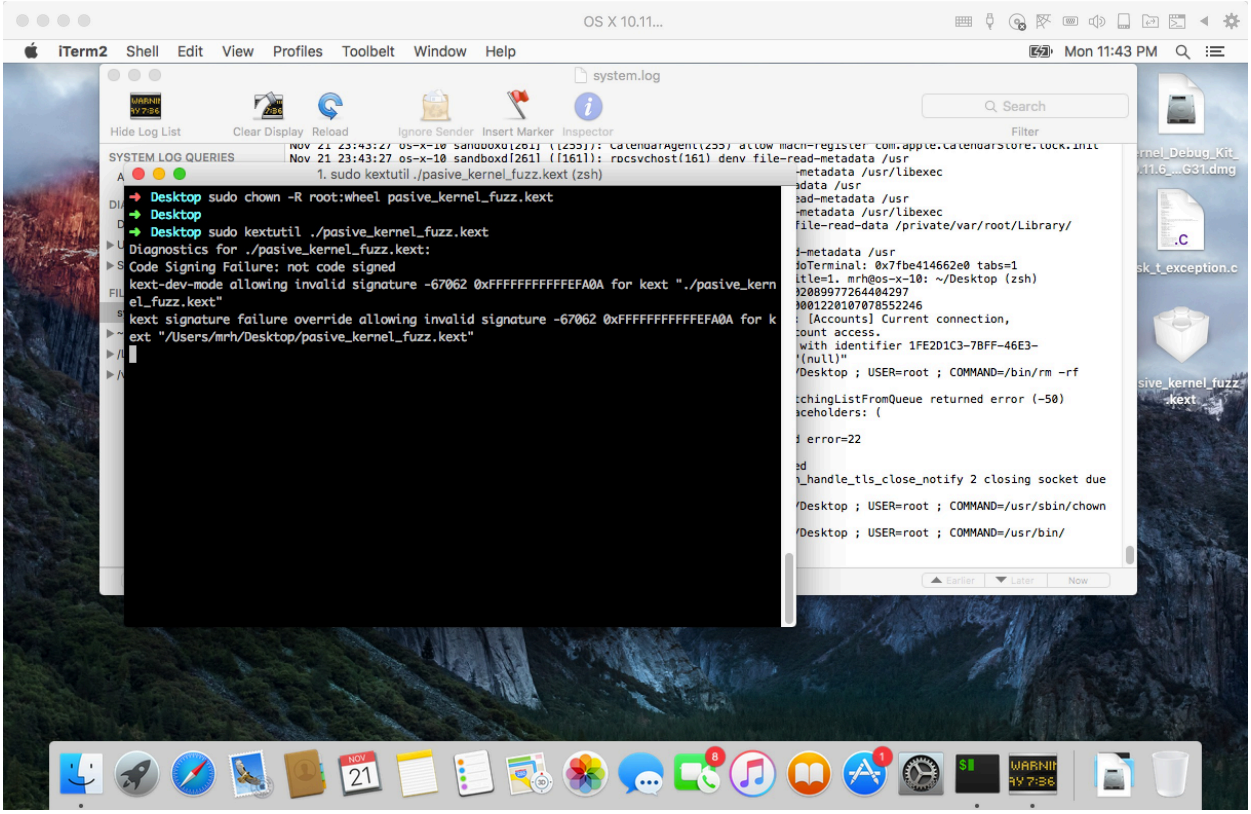
3.2 加载内核

没有签名是加载内核需要把`rootless`关掉，也就是`SIP`要关掉。

关闭之后就正常的按照文档中的介绍加载内核就可以了。

3.3 执行崩溃

加载内核之后，系统立刻就崩溃了。



调用栈如下

```
Process 1 stopped
* thread #1: tid = 0x0001, 0xffffff800d90e05b kernel.development`memcpy + 11, stop reason = EXC_BAD_ACCESS (code=1, address=0x163de000)
    frame #0: 0xffffff800d90e05b kernel.development`memcpy + 11
kernel.development`memcpy:
->  0xffffff800d90e05b <+11>: rep
    0xffffff800d90e05c <+12>: movsq  (%rsi), %es:(%rdi)
    0xffffff800d90e05e <+14>: movq   %rdx, %rcx
    0xffffff800d90e061 <+17>: andq   $0x7, %rcx
(lldb) bt
* thread #1: tid = 0x0001, 0xffffff800d90e05b kernel.development`memcpy + 11, stop reason = EXC_BAD_ACCESS (code=1, address=0x163de000)
  * frame #0: 0xffffff800d90e05b kernel.development`memcpy + 11
    frame #1: 0xffffff7f8e7485e8
    frame #2: 0xffffff7f8e746630
    frame #3: 0xffffff800e0b8507 kernel.development`::shim_io_connect_method_scalarI_scalarO(method=<unavailable>, object=<unavailable>, input=<unavailable>, inputCount=<unavailable>, output=<unavailable>, outputCount=<unavailable>) + 647 at IOUserClient.cpp:4086 [opt]
    frame #4: 0xffffff800e0bacd8 kernel.development`IOUserClient::externalMethod(this=<unavailable>, selector=<unavailable>, args=<unavailable>, dispatch=<unavailable>, target=<unavailable>, reference=<unavailable>) + 632 at IOUserClient.cpp:5295 [opt]
```

```
    frame #5: 0xffffff800e0b7e17
kernel.development`::is_io_connect_method(connection=0xffffff886e39ba50,
selector=10, scalar_input=<unavailable>, scalar_inputCnt=<unavailable>,
inband_input=<unavailable>, inband_inputCnt=0, ool_input=<unavailable>,
ool_input_size=<unavailable>, inband_output=<unavailable>, inband_outputCnt=
<unavailable>, scalar_output=<unavailable>, scalar_outputCnt=<unavailable>,
ool_output=<unavailable>, ool_output_size=<unavailable>) + 487 at
IOUserClient.cpp:3900 [opt]
    frame #6: 0xffffff7f8f84dd03
pasive_kernel_fuzz`trampline_is_io_connect_method(connection=0xffffff8016acc20
0, selector=10, scalar_input=0xffffff8016659330, scalar_inputCnt=2,
inband_input="", inband_inputCnt=0, ool_input=0, ool_input_size=0,
inband_output="", inband_outputCnt=0xffffff8018e155fc,
scalar_output=0xffffff886e39bd30, scalar_outputCnt=0xffffff886e39bd2c,
ool_output=0, ool_output_size=0xffffff8016659364) + 1587 at
is_io_connect_method_trampline.c:663
    frame #7: 0xffffff800db58750
kernel.development`_Xio_connect_method(InHeadP=<unavailable>,
OutHeadP=0xffffff8018e155d0) + 384 at device_server.c:8255 [opt]
    frame #8: 0xffffff800da83443
kernel.development`ipc_kobject_server(request=0xffffff80166592a0) + 259 at
ipc_kobject.c:340 [opt]
    frame #9: 0xffffff800da5ef03 kernel.development`ipc_kmsg_send(kmsg=
<unavailable>, option=<unavailable>, send_timeout=0) + 211 at ipc_kmsg.c:1443
[opt]
    frame #10: 0xffffff800da75985
kernel.development`mach_msg_overwrite_trap(args=<unavailable>) + 197 at
mach_msg.c:474 [opt]
    frame #11: 0xffffff7f8f85df47
pasive_kernel_fuzz`trampline_mach_msg_overwrite_trap(args=0xffffff886e39bf28)
+ 311 at mach_msg_overwrite_trap_trampline.c:131
    frame #12: 0xffffff800db7f000
kernel.development`mach_call_munger64(state=0xffffff8015920720) + 480 at
bsd_i386.c:560 [opt]
    frame #13: 0xffffff800dbb4de6 kernel.development`hndl_mach_scall64 + 22
```

可以看到frame #1 #2 #3的调用栈没有具体的函数符号，跳转到frame #6 通过变量获取一些信息。

```
(lldb) f 6
frame #6: 0xffffff7f8f84dd03
pasive_kernel_fuzz`trampline_is_io_connect_method(connection=0xffffff8016acc200, selector=10, scalar_input=0xffffff8016659330, scalar_inputCnt=2, inband_input="", inband_inputCnt=0, ool_input=0, ool_input_size=0, inband_output="", inband_outputCnt=0xffffff8018e155fc, scalar_output=0xffffff886e39bd30, scalar_outputCnt=0xffffff886e39bd2c, ool_output=0, ool_output_size=0xffffff8016659364) + 1587 at is_io_connect_method_trampline.c:663
   660          g_fuzz_sample_info_index--;
   661      }
   662      lck_mtx_unlock(g_fuzz_sample_info_mutext);
-> 663      kr = ((fn_is_io_connect_method_t )inlined_part_is_io_connect_method)(
   664                                                  //io_connect_t
   665                                                  connection,
   666                                              //uint32_t

(lldb) p (char*)g_fuzz_sample_info[g_fuzz_sample_info_index].env.szClassName
(char *) $3 = 0xffffff7f8faa9038 "prl_video_ac_surface"
(lldb) p (char*)g_fuzz_sample_info[g_fuzz_sample_info_index].env.szProcName
(char *) $4 = 0xffffff7f8faa9849 "WindowServer"
(lldb) p (char*)g_fuzz_sample_info[g_fuzz_sample_info_index].env.szServiceClassName
(char *) $5 = 0xffffff7f8faa9448 <no value available>
```

但是看不到`ServiceClassName`，调用内核调试的脚本showallkmods。

```
(lldb) showallkmods
UUID                                 kmod_info            address
   size                  id  refs              version name
B247A93A-1FEF-3A25-AECB-93F84593FDF6 0xffffff7f8f8fc660   0xffffff7f8f849000
  0xfc1000             110     0                   1 put.as.pasive-kernel-fuzz
5208EAB1-6B60-317E-83AF-FCE74FDD9B7F 0xffffff7f8e6a7a68   0xffffff7f8e64d000
  0x5e000              109     0                3.0.1 com.apple.filesystems.smbfs
2461725B-E5F1-3947-8AD8-8781308FA614 0xffffff7f8f20e508   0xffffff7f8f206000
  0x9000               108     0                  3.0 com.apple.filesystems.autofs
5A796890-4ED5-3BA9-8638-84EBBBDD2D53 0xffffff7f8e64b020   0xffffff7f8e648000
  0x5000               107     2                  1.0 com.apple.kext.triggers
47657EC5-2536-3174-97A7-6C19BA1067A4 0xffffff7f8f3854cc   0xffffff7f8f37e000
  0x9000               106     0              1.0.2d2 com.apple.driver.AppleTyMCEDriver
```

```
D9EF7435-0F3C-37BD-AA34-F1B7353C8D4F 0xffffff7f8f2ca528   0xffffff7f8f2c6000
  0x5000                105     0                 1.70
com.apple.driver.AudioAUUC
1AF6EE80-2420-C5D8-7F83-9E2446C1320C 0xffffff7f8f6db900   0xffffff7f8f6d7000
  0x5000                104     0                 1.1.0
com.parallels.driver.AppleIntelAC97Audio
D0B2999C-6EB8-3E88-8CA5-60E05B6AD462 0xffffff7f8f5964a0   0xffffff7f8f593000
  0x4000                103     0                   1
com.apple.driver.AppleOSXWatchdog
B418B8D4-27F9-373C-8BEC-802554F3F794 0xffffff7f8e776d10   0xffffff7f8e76c000
  0xb000                102     0                   1
com.apple.driver.pmtelemetry
B2989894-92E1-3A9F-8F9A-924960162ABB 0xffffff7f8e8a1e20   0xffffff7f8e89c000
  0x6000                101     0                 1.0.1
com.apple.iokit.IOUserEthernet
354FC780-7EA4-3C3F-A9E1-2658F63663A9 0xffffff7f8ed75b80   0xffffff7f8ed69000
  0x13000               100     0               108.2.3
com.apple.iokit.IOSurface
57A03351-E0D2-348B-A442-C4F5B6C4310E 0xffffff7f8ef41978   0xffffff7f8ef38000
  0xa000                99      0               4.4.6f1
com.apple.iokit.IOBluetoothSerialManager
854221E2-E1EB-361A-98C6-DC6B7E1EB0F4 0xffffff7f8ed964e0   0xffffff7f8ed8d000
  0xe000                98      1                   11
com.apple.iokit.IOSerialFamily
9B3C8089-B7BD-32F6-AC84-18501E3A5B1C 0xffffff7f8efdacb0   0xffffff7f8ef4c000
  0xc3000               97      0               4.4.6f1
com.apple.iokit.IOBluetoothFamily
3164D09B-101E-38E5-9399-BDE428C1E877 0xffffff7f8f0cb9a0   0xffffff7f8f0c8000
  0x5000                96      0                 7.0.0
com.apple.Dont_Steal_Mac_OS_X
D04FEA18-A347-31F1-A04A-ED762DB9F0DE 0xffffff7f8f1f6668   0xffffff7f8f1f3000
  0x7000                95      0                   1
com.apple.driver.CoreCaptureResponder
0B572B9E-F1D5-31B0-AB6B-E19630F5126C 0xffffff7f8f1e1c68   0xffffff7f8f1c9000
  0x22000               94      1                 1.0.4
com.apple.driver.corecapture
D8BF62A5-B66D-3495-9FBF-0B6D89CB2455 0xffffff7f8f6ef040   0xffffff7f8f6e6000
  0xa000                93      0                   1
com.apple.driver.AppleHV
A5A73220-5E26-3283-B29B-FEA298629620 0xffffff7f8f63f9d0   0xffffff7f8f63e000
  0x2000                92      0                 4.0.0
com.apple.driver.AppleIntelSlowAdaptiveClocking
514292C4-55BD-3550-9DEB-1431BC04A629 0xffffff7f8ed84c78   0xffffff7f8ed82000
  0x5000                91      1                 1.0.0
com.apple.iokit.IOSlowAdaptiveClockingFamily
864DACA4-8F2F-3442-B389-A42434D0F878 0xffffff7f8f77c9d8   0xffffff7f8f77a000
  0x3000                90      0                 4.1.0
com.apple.driver.AppleFIVRDriver
```

```
EB8B7A35-4C31-A25F-87FC-AB5AAE190585 0xffffff7f8e7664c0   0xffffff7f8e762000
  0x5000                89     0         12.1.0 41489 com.parallels.kext.tg
6E57BC33-C4AF-3611-97B7-31CA2A2C88DD 0xffffff7f8f379520   0xffffff7f8f375000
  0x5000                88     0                3.6.1
com.apple.driver.AppleUpstreamUserClient
FE49EB19-A41C-3E7B-89CE-5411C85593D4 0xffffff7f8f5be6f8   0xffffff7f8f5b1000
  0xe000                87     0               1.2.13
com.apple.driver.AppleMCCSControl
546B348A-B5A7-3C41-9DDE-02A0A7B640B5 0xffffff7f8f5a96f0   0xffffff7f8f5a0000
  0xe000                86     1             1.0.14d1
com.apple.driver.AppleSMBusController
4EB2843C-C821-3AD0-B333-575FD6ED6FB1 0xffffff7f8eea1c58   0xffffff7f8ee98000
  0x10000               85     0                2.4.1
com.apple.iokit.IONDRVSupport
5820F1CC-8084-3447-AA45-ABB32D56EF25 0xffffff7f8ee42550   0xffffff7f8ee32000
  0x11000               84     0                1.0.0
com.apple.driver.ACPI_SMC_PlatformPlugin
7444F81D-08A8-36CE-A017-247C66701E4D 0xffffff7f8ee0e2c0   0xffffff7f8ee03000
  0x12000               83     1                1.0.0
com.apple.driver.IOPlatformPluginLegacy
4BEF649C-7CFD-31CA-8D84-1F0DB25BF60B 0xffffff7f8edfedf8   0xffffff7f8edf9000
  0xa000                82     3              6.0.0d7
com.apple.driver.IOPlatformPluginFamily
0C1376F2-15F7-30AD-BEEB-4CFC4AC4240A 0xffffff7f8f50b9a0   0xffffff7f8f50a000
  0x3000                81     0             1.0.14d1
com.apple.driver.AppleSMBusPCI
5F94D8E3-B1E5-35D7-AB7A-6419C3AAC3B5 0xffffff7f8f764f40   0xffffff7f8f750000
  0x1e000               80     0                274.12
com.apple.driver.AppleHDAController
C6423C28-4CFB-32A8-BDD1-2D149DE52F74 0xffffff7f8f748608   0xffffff7f8f741000
  0xc000                79     1                274.12
com.apple.iokit.IOHDAFamily
16D694E8-A341-3DAC-A710-57BC95EF7758 0xffffff7f8f2ab810   0xffffff7f8f28d000
  0x31000               78     3                204.4
com.apple.iokit.IOAudioFamily
C334E229-C366-3862-8A15-2399723870C6 0xffffff7f8f28b220   0xffffff7f8f211000
  0x7c000               77     1                1.2.0 com.apple.vecLib.kext
90BC49A2-E1CC-4A22-FD55-BEF0A13A15DA 0xffffff7f8f6ccfa0   0xffffff7f8f6c8000
  0xf000                76     1                1.1.0
com.parallels.driver.AppleIntelAC97Controller
C1523713-8957-31FE-AA11-62E1787969B1 0xffffff7f8ee2bb98   0xffffff7f8ee17000
  0x19000               75     3                3.1.9
com.apple.driver.AppleSMC
5256D8E2-78AD-245C-06FB-3C14D329A8BF 0xffffff7f8e750600   0xffffff7f8e745000
  0x1a000               74     0         12.1.0 41489
com.parallels.kext.video
A360453D-2050-3C49-A549-AC0DD5E87917 0xffffff7f8e728b78   0xffffff7f8e6fe000
  0x3b000               73     6                2.4.1
com.apple.iokit.IOGraphicsFamily
```

```
41 E4D6A0C3-5C8D-3652-B673-3FAFAA027E2E 0xffffff7f8f3e1ac8   0xffffff7f8f3dd000
     0x9000                72     0                  181
   com.apple.driver.AppleHIDKeyboard
42 AC74012B-CBBC-323B-B760-0E31B664B7B5 0xffffff7f8eab1cb0   0xffffff7f8eaaa000
     0xa000                71     0                1.0.1
   com.apple.driver.usb.IOUSBHostHIDDevice
43 69D22F66-81DA-3CCF-8451-A7EBC991A601 0xffffff7f8ebe7dd0   0xffffff7f8ebc1000
     0x2f000               70     0                1.0.1
   com.apple.driver.usb.AppleUSBHub
44 204EA973-076B-3517-93BF-9057AD3DDBD6 0xffffff7f8f31d420   0xffffff7f8f31a000
     0x9000                69     0                5.0.0
   com.apple.driver.usb.cdc
45 70876950-E9CC-313C-A239-5C1E5CA936BF 0xffffff7f8f3147e8   0xffffff7f8f312000
     0x8000                68     1                5.0.0
   com.apple.driver.usb.networking
46 DC814FD4-7773-3054-8D2C-AFA8E6CA034A 0xffffff7f8ebfc008   0xffffff7f8ebf8000
     0x8000                67     1                1.0.1
   com.apple.driver.usb.AppleUSBHostCompositeDevice
47 AB75EB2D-F0DF-34C8-8CDC-C3BD98F6CFAD 0xffffff7f8ede5980   0xffffff7f8edd4000
     0x1a000               66     0                3.7.7
   com.apple.iokit.IOSCSIMultimediaCommandsDevice
48 F5455A4B-6444-375B-B41A-9DD21ED76068 0xffffff7f8edcd350   0xffffff7f8edc8000
     0x9000                65     1                  1.8
   com.apple.iokit.IOBDStorageFamily
49 D13AB661-E2CF-3761-8D59-61A1AC99637E 0xffffff7f8edc03c0   0xffffff7f8edba000
     0xb000                64     2                  1.8
   com.apple.iokit.IODVDStorageFamily
50 293F10B5-7BF9-3C0C-976E-9E588489C303 0xffffff7f8edb1c30   0xffffff7f8eda9000
     0xe000                63     3                  1.8
   com.apple.iokit.IOCDStorageFamily
51 12F3B8F2-0104-362F-9B60-7751D73D4D80 0xffffff7f8eda4748   0xffffff7f8ed9e000
     0x7000                62     0                3.7.7
   com.apple.iokit.SCSITaskUserClient
52 945A757B-968E-34CE-9067-525B6D0F8550 0xffffff7f8f1c4a58   0xffffff7f8f1af000
     0x16000               61     0              517.50.1
   com.apple.driver.CoreStorageFsck
53 3B950A54-941B-380E-9C93-31E70591696E 0xffffff7f8f186e90   0xffffff7f8f0ce000
     0xdb000               60     1              517.50.1
   com.apple.driver.CoreStorage
54 642EEDB3-7CE7-354A-9030-C644EC3D9083 0xffffff7f8f2d2f10   0xffffff7f8f2ce000
     0xa000                59     0                  3
   com.apple.driver.AppleXsanScheme
55 C8E6B461-D83A-3660-8A8B-43EA262D4C11 0xffffff7f8f09a390   0xffffff7f8f092000
     0xd000                58     0                2.6.2
   com.apple.iokit.IOAHCISerialATAPI
56 07D953DC-7B94-3AE8-A379-2B02F5D538F9 0xffffff7f8e303eb0   0xffffff7f8e2e9000
     0x2b000               57     3                3.7.7
   com.apple.iokit.IOSCSIArchitectureModelFamily
```

```
03357B30-E9B5-32DC-819D-CACA2B37AE19 0xffffff7f8f0ba0d0   0xffffff7f8f09f000
  0x1c000               56     0                2.8.5
com.apple.iokit.IOAHCIBlockStorage
D2861F03-33FC-3B1D-9572-00E11EB6B530 0xffffff7f8f7809a8   0xffffff7f8f77f000
  0x2000                55     0                3.0.1
com.apple.driver.AppleFileSystemDriver
8A48FC7E-CD9D-39E4-A243-59BBEB5D65BE 0xffffff7f8f778a40   0xffffff7f8f776000
  0x3000                54     0              1.0.0d1
com.apple.AppleFSCompression.AppleFSCompressionTypeDataless
8A37264E-9D9A-3B95-B0A1-EB1947CF70DA 0xffffff7f8f773460   0xffffff7f8f76e000
  0x6000                53     0                1.0.0
com.apple.AppleFSCompression.AppleFSCompressionTypeZlib
C1EA21DC-CEC4-34EF-8172-8D217927D3EC 0xffffff7f8f202a18   0xffffff7f8f1fa000
  0xa000                52     0                  38 com.apple.BootCache
C94AA870-B315-3B68-8942-94FD5095B016 0xffffff7f8ec0a548   0xffffff7f8ec00000
  0xb000                51     0                1.0.1
com.apple.driver.usb.AppleUSBEHCIPCI
D08C7EEE-CA01-3053-8D47-2587BBCF9C95 0xffffff7f8eb1c438   0xffffff7f8eb03000
  0x20000               50     0                1.0.1
com.apple.driver.usb.AppleUSBXHCIPCI
931A3E67-1954-320C-9619-C41C0A516804 0xffffff7f8eaf15e0   0xffffff7f8eabd000
  0x40000               49     1                1.0.1
com.apple.driver.usb.AppleUSBXHCI
5BF1DFC0-0647-3A29-8555-8C0341DFB8A3 0xffffff7f8eb93cb8   0xffffff7f8eb90000
  0x4000                48     0                1.0.1
com.apple.driver.usb.AppleUSBUHCIPCI
2F1227C1-A3A9-3709-B7A9-93506A65AB00 0xffffff7f8eb83e38   0xffffff7f8eb6e000
  0x1f000               47     1                1.0.1
com.apple.driver.usb.AppleUSBUHCI
0D241802-E006-3A72-B3C5-09ABDE37DAE0 0xffffff7f8eb5dfd8   0xffffff7f8eb2b000
  0x3f000               46     3                1.0.1
com.apple.driver.usb.AppleUSBEHCI
B9AFCAD5-5FFE-3E65-A186-CF4EA4571BCC 0xffffff7f8f7b8c98   0xffffff7f8f79a000
  0x1f000               45     0                3.1.8
com.apple.driver.AppleAHCIPort
58B77CC0-5211-342E-8935-8D05E0B96867 0xffffff7f8f087520   0xffffff7f8f074000
  0x1b000               44     3                2.8.1
com.apple.iokit.IOAHCIFamily
A6DFC31E-C32B-3FD9-8FC9-27791F49BBB5 0xffffff7f8f06b668   0xffffff7f8f062000
  0xa000                43     0                2.5.1
com.apple.driver.AppleIntelPIIXATA
148B6371-28AE-30E6-B469-00A9587C7EFF 0xffffff7f8f030668   0xffffff7f8f022000
  0x19000               42     1                2.5.3
com.apple.iokit.IOATAFamily
F74DD10C-690C-3B74-98F5-F50E093A1E68 0xffffff7f8ee90ce0   0xffffff7f8ee80000
  0x11000               41     0              3.1.4b1
com.apple.driver.AppleIntel8254XEthernet
```

```
848B398F-4D96-3024-8092-6CD3534D2CCA 0xffffff7f8e883f50   0xffffff7f8e868000
  0x2d000               40     2                  3.2
com.apple.iokit.IONetworkingFamily
FB980EB5-1F73-348E-9554-6F9F6FE481FD 0xffffff7f8e9c95d0   0xffffff7f8e953000
  0x9a000               39     0              900.4.1
com.apple.iokit.IOUSBFamily
79D250A3-843A-3750-BE64-A252CF17A148 0xffffff7f8e8fadb4   0xffffff7f8e8ac000
  0x6a000               38    12                1.0.1
com.apple.iokit.IOUSBHostFamily
401B7165-36E8-3EE5-849D-71DAEB3E46E4 0xffffff7f8e8a99a8   0xffffff7f8e8a8000
  0x4000                37     2                1.0.1
com.apple.driver.AppleUSBHostMergeProperties
0A3B2F6D-7FC8-34C1-B4B2-B4A53DC85DE5 0xffffff7f8f519ed0   0xffffff7f8f513000
  0x8000                36     0              161.0.0
com.apple.driver.AppleSmartBatteryManager
DFA558FE-59F9-32AA-8C3A-82BD65ECC094 0xffffff7f8ee550e0   0xffffff7f8ee4f000
  0xa000                35     0                  2.0
com.apple.driver.AppleEFINVRAM
CFC72657-568A-33B3-B84A-CF659674E655 0xffffff7f8ee4ca10   0xffffff7f8ee4a000
  0x5000                34     1                  2.0
com.apple.driver.AppleEFIRuntime
456C28F7-4F2B-3F00-97C9-BF6023DADD7C 0xffffff7f8f83bcb0   0xffffff7f8f838000
  0x4000                33     0                  4.0
com.apple.driver.AppleACPIButtons
9A3F90D7-1A6A-3FCD-9689-0BF6A66A29A2 0xffffff7f8ea75c58   0xffffff7f8ea1e000
  0x78000               32     3                2.0.0
com.apple.iokit.IOHIDFamily
801E20D9-1D7A-353F-A638-05430128D61D 0xffffff7f8f72c9c8   0xffffff7f8f72a000
  0x3000                31     0                  1.8
com.apple.driver.AppleHPET
87E6B264-9FE7-354F-A83B-2AF966681A50 0xffffff7f8f834540   0xffffff7f8f82e000
  0x7000                30     0                  4.0
com.apple.driver.AppleACPIEC
EA577FC5-B1EE-38B4-9B62-2938C01C2CB2 0xffffff7f8ed7fc58   0xffffff7f8ed7e000
  0x4000                29     3                  1.1
com.apple.iokit.IOSMBusFamily
6409E881-1F83-380E-8F03-F21DCFC4BF53 0xffffff7f8f5424f0   0xffffff7f8f53b000
  0x8000                28     0                  2.0
com.apple.driver.AppleRTC
BEB6C00A-8353-3DE6-A438-3D8AE2F9A5F0 0xffffff7f8f510ab8   0xffffff7f8f50d000
  0x4000                27     0                  2.1
com.apple.driver.AppleSMBIOS
04696395-E633-3657-89BD-9908A5C60F56 0xffffff7f8f797a28   0xffffff7f8f795000
  0x3000                26     0                  1.7
com.apple.driver.AppleAPIC
113F310F-1904-3F41-A206-1D275BF7A397 0xffffff7f8f846280   0xffffff7f8f83f000
  0x8000                25     0                  163
com.apple.nke.applicationfirewall
```

```
89  0ECF10A0-C16A-3013-B6B6-CD9F3DA76B01 0xffffff7f8e6efc00   0xffffff7f8e6e7000
      0x9000                 24     0                    3
    com.apple.security.quarantine
90  E32DE435-FAD0-3222-807A-32711BCB979B 0xffffff7f8e6e15c8   0xffffff7f8e6c9000
      0x1e000                23     1                300.0
    com.apple.security.sandbox
91  F2211BA2-E656-3187-B06E-CF9D6A3A3B5A 0xffffff7f8e6c7008   0xffffff7f8e6c4000
      0x5000                 22     2              1.0.0d1
    com.apple.kext.AppleMatch
92  7F6B05B1-14AC-3634-B5CA-7F69452730B4 0xffffff7f8e5b1b08   0xffffff7f8e5b0000
      0x2000                 21     0                    8
    com.apple.security.TMSafetyNet
93  8CF8BDE4-6E36-3163-9EA1-DB09980ED7B2 0xffffff7f8f62e60c   0xffffff7f8f60c000
      0x2b000                20     0                    2
    com.apple.driver.AppleKeyStore
94  D330951F-27FC-3A94-94A6-2AD7850E9C3C 0xffffff7f8e6ba808   0xffffff7f8e6b2000
      0x12000                19     2                1.0.5
    com.apple.driver.AppleMobileFileIntegrity
95  22552717-92AB-3B19-98B5-5C067A104219 0xffffff7f8f606580   0xffffff7f8f5ee000
      0x1e000                18     1                  1.0
    com.apple.driver.AppleCredentialManager
96  CDCF4D3F-89CC-3CDD-AB89-B6FFD304F26B 0xffffff7f8eeb9930   0xffffff7f8eea8000
      0x19000                17     0                417.4
    com.apple.driver.DiskImages
97  DC1AAB7C-F417-3238-BB3F-2A5B84D67B90 0xffffff7f8e265550   0xffffff7f8e24c000
      0x27000                16    12                  2.1
    com.apple.iokit.IOStorageFamily
98  C89107EE-2DF2-3BC3-9F6D-3133D43ED7EF 0xffffff7f8edf54d8   0xffffff7f8edf2000
      0x7000                 15     2                   31
    com.apple.iokit.IOReportFamily
99  C31A19C9-8174-3E35-B2CD-3B1B237C0220 0xffffff7f8ea1a4e0   0xffffff7f8ea13000
      0xb000                 14     1                28.30
    com.apple.driver.AppleFDEKeyStore
100 A29C7512-D3A8-3AED-9721-3A5FF1A32EB2 0xffffff7f8f8149f8   0xffffff7f8f7c8000
      0x60000                13     2                  4.0
    com.apple.driver.AppleACPIPlatform
101 F51AA3D6-EC2F-3AD3-A043-06DB79027AA2 0xffffff7f8e351c48   0xffffff7f8e32c000
      0x30000                12    22                  2.9
    com.apple.iokit.IOPCIFamily
102 5D7574C3-8E90-3873-BAEB-D979FC215A7D 0xffffff7f8eab7a00   0xffffff7f8eab4000
      0x9000                 11    21                  1.4
    com.apple.iokit.IOACPIFamily
103 9DDD9196-3824-3DCA-BAAA-7F383BC13C37 0xffffff7f8e784000   0xffffff7f8e77d000
      0x9000                 10     1                    1 com.apple.kec.Libm
104 39D0B4EB-B7F4-3891-96C2-F8B886656C8A 0xffffff7f8e6fb6b8   0xffffff7f8e6f1000
      0xd000                  9     0                    1 com.apple.kec.pthread
105 ABDB0534-113E-3A88-8B89-52345E3AFDF7 0xffffff7f8e612ea0   0xffffff7f8e5b3000
      0x95000                 8     5                  1.0
    com.apple.kec.corecrypto
```

```
106  ........-....-....-....-............ 0xffffff80142c1e00   0x0
        0x0                    7    60              15.6.0
     com.apple.kpi.unsupported
107  ........-....-....-....-............ 0xffffff80142c1d00   0x0
        0x0                    6    45              15.6.0 com.apple.kpi.private
108  ........-....-....-....-............ 0xffffff80142c1100   0x0
        0x0                    5    88              15.6.0 com.apple.kpi.mach
109  ........-....-....-....-............ 0xffffff80142c1200   0x0
        0x0                    4   103              15.6.0 com.apple.kpi.libkern
110  ........-....-....-....-............ 0xffffff80142c1300   0x0
        0x0                    3    97              15.6.0 com.apple.kpi.iokit
111  ........-....-....-....-............ 0xffffff80142c1c00   0x0
        0x0                    2     8              15.6.0 com.apple.kpi.dsep
112  ........-....-....-....-............ 0xffffff80142c1b00   0x0
        0x0                    1    76              15.6.0 com.apple.kpi.bsd
```

根据内核地址的比较

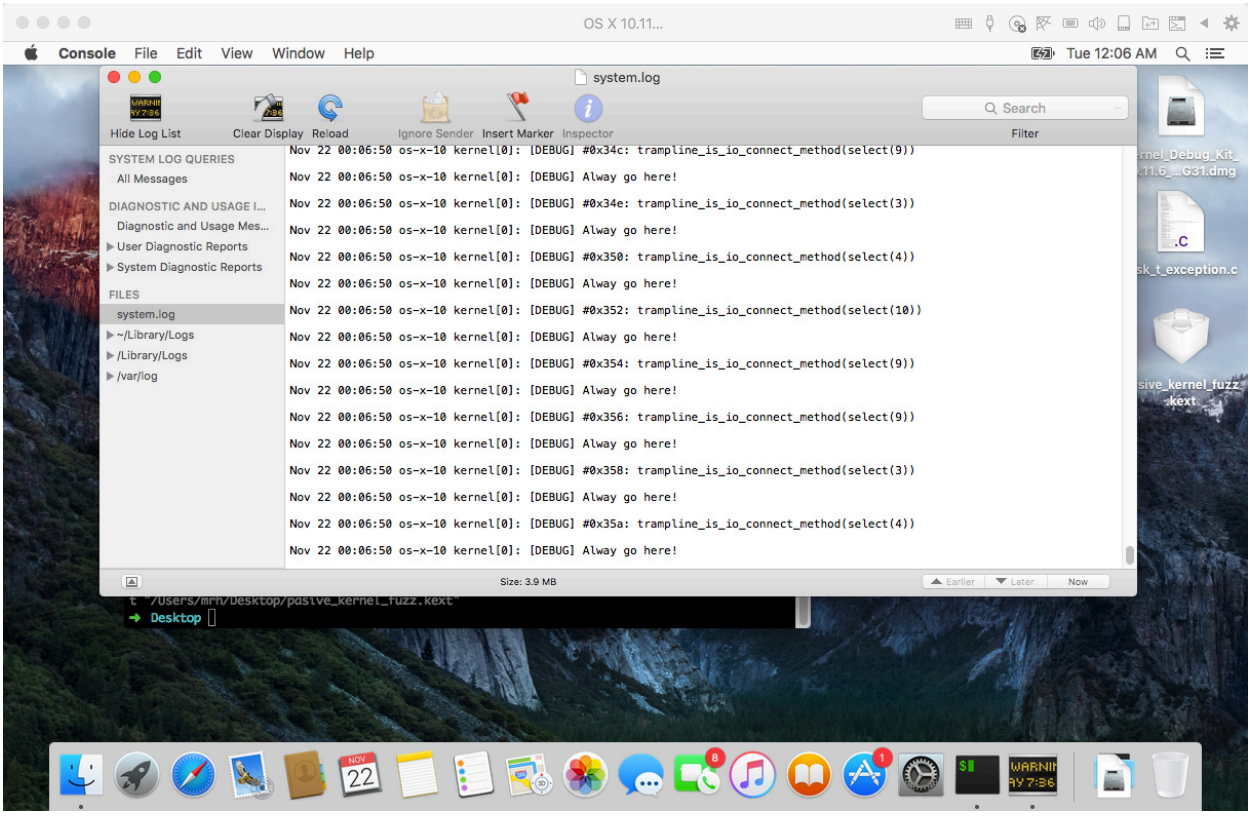
```
5256D8E2-78AD-245C-06FB-3C14D329A8BF 0xffffff7f8e750600   0xffffff7f8e745000
0x1a000               74     0         12.1.0 41489 com.parallels.kext.video
----
 frame #0: 0xffffff800d90e05b kernel.development`memcpy + 11
 frame #1: 0xffffff7f8e7485e8
 frame #2: 0xffffff7f8e746630
```

可以看到崩溃在`com.parallels.kext.video`这个虚拟机的内核扩展中，所以安装了`KDK`也没有函数符号。这个模块不是我们感兴趣的，所以添加一个白名单，剔除掉对这个调用的fuzz。

```
    "*",PROCESS_UID_ANY_INTEGER,"*","IGAccelSharedUserClient",0,//tocheck-43
    //By experience:
    //"profil",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
    //"notify",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
    //"watchdog",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
    //"vmware",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
    //"*","PROCESS_UID_ANY_INTEGER,*","vmware",ANY_MATCH_INTEGER,


//"*",PROCESS_UID_ANY_INTEGER,"*","AppleOSXWatchdogClient",ANY_MATCH_INTEGER,
    //"*",PROCESS_UID_ANY_INTEGER,"*","AppleSMCClient",ANY_MATCH_INTEGER,

    //Testing:
    //"SystemUIServer",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
    "WindowServer",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER, <--
windowserver 所有的相关的调用都不进行fuzz


    //"dock",PROCESS_UID_ANY_INTEGER,"*","*",ANY_MATCH_INTEGER,
```

这样一来，导致我们崩溃的这一条就不会进行`fuzz`从而剔除了这个干扰。编译之后再次加载

3.4 再次加载内核扩展



我把他们注释掉的调试信息都放开了，如果不把注释删了也看不到这些日志。

0x04 小结

初步的使用这个`fuzzing`框架的过程就是这样，这个框架还会对其他几个核心函数进行测试，在`ppt`中也有所描述。同时这里也有相对应的测试开关。

```
//bFuzzing flags
    g_inline_hook_entry[INLINE_ENUM_CREATE_MAPPING_IN_TASK].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_IPC_KMSG_GET].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_COPY_IO].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_IS_IO_CONNECT_METHOD].bFuzzing = true;

    g_inline_hook_entry[INLINE_ENUM_IPC_KMSG_SEND].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_MACH_MSG_OVERWRITE_TRAP].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_IS_IO_CONNECT_ASYNC_METHOD].bFuzzing =
false;
    g_inline_hook_entry[INLINE_ENUM_KDP_PANIC_DUMP].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_IOKIT_USER_CLIENT_TRAP].bFuzzing = false;
    g_inline_hook_entry[INLINE_ENUM_IS_IO_SERVICE_OPEN_EXTENDED].bFuzzing =
false;

```

本次实验中只是修改了白名单，框架中还提供了黑名单，简单阅读源码之后也很容易理解它的功能，白名单和黑名单的实现，可以让fuzz更加的稳定。

引用

[1]Revisiting Mac OS X Kernel Rootkits

http://phrack.org/issues/69/7.html#article

[2]Resolving kernel symbols

http://ho.ax/posts/2012/02/resolving-kernel-symbols/

[3]PassiveFuzzFrameworkOSX

https://github.com/SilverMoonSecurity/PassiveFuzzFrameworkOSX