Fresh Apples: Researching New Attack Interfaces on iOS and MacOS

Lilang Wu, Moony Li
- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interfaces
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interfaces
- Automatic Fuzzing solution
- 0Day vulnerabilities found
About us

- Lilang Wu
  - 4 years security
  - Mobile advance threat research
  - MacOS/iOS Vulnerability/Malware
  - @Lilang_Wu

- Moony Li
  - 9 years security
  - MacOS/Android/iOS vulnerability hunt and exploit
  - Sandbox/Emulator Development
  - @Flyic
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
Solution Overview

1. Identify attack interfaces by automatically reverse engineering

   - P-Diff (Diff Operation)
   - Kext Binary
   - UserClients diff
   - External Methods diff
   - XNU Syscall/Sysctl Part Source Code
   - Updated functions' entry list

2. Dynamic analysis for kernel attack interface

   - User
     - User Clients and openType Generator
     - External Methods Generator
     - External Methods diff
     - User Clients and openType Generator
   - Kernel
     - Kemon.kext
     - Kernel
     - Passive kernel fuzzer
     - Passive kernel fuzzer

3. Kernel fuzzing

   - Fuzzing system
     - Kernel Sanitizer
     - Graphic Driver
     - Touchbar Driver
     - HID driver
     - Updated functions' entry list
     - Updated functions' entry list
     - Updated functions' entry list
     - Updated functions' entry list

Dynamic analysis

- Mach_Msg
- IOConnectMethod
- Syscall
- Frida Hook
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interfaces
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interfaces
- Automatic Fuzzing solution
- 0Day vulnerabilities found
Attack Surfaces

Factory

Military Base

Big City
Think about Apple System

**Key Components**

- **XPC Services**
  - Interprocess communication mechanism

- **MIG system**
  - Generates RPC code for communication between a client and a server process

- **Kernel**
  - Core which controls over everything in the system

- **Kernel Extensions**
  - Operates or controls a particular type of device
New Attack Interfaces Generator

P-Diff (Diff Operation)
- Kexts Binary
- External Methods diff
- XNU Syscall/Sysctl Part Source Code
- XNU Trap/MIG Part Source Code

Interfaces Optimizer
- updated Userclients
- Kexts updated Functions' entry list
- XUN updated Functions' entry list
- XUN updated Functions' entry list

UserClients and openType Generator
- Class/Method String refine

Kexts External Methods Generator
- Kexts External Methods Generator

Syscall/Sysctl Generator
- Syscall/Sysctl Generator

Trap/MIG Generator
- Trap/MIG Generator

New Attack Interfaces
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
KEXTs Interfaces Analysis Flow

1. Classes inheritance relationship
2. Class/Method names refine
3. Connection Type - User Clients
4. User Client External Methods
Classes inheritance relationship

OSMetaClass::OSMetaClass

- manages run-time type information for Libkern and I/O Kit C++ classes
- rdi/x0: instance of register Meta class
- rsi/x1: Meta class name
- rdx/x2: instance of parent Meta class
- rcx/w3: size of register Meta class instance

```assembly
GLOBAL _sub_1_10AccelMemory_cpp proc near
push    rbp
mov     rbp, rsp
lea     rdi, _ZN13OSMetaClass10gMetaClassE ; O00AccelMemory::gMetaClass
lea     rsi, aloaccelmemory ; "10AccelMemory"
mov     rdx, cs:ZN4OSObject10gMetaClassE_0 ; O00Object::gMetaClass
mov     ecx, 0AHh ;
call   _ZN10OSMetaClassC2EPKcPKS_j ; O00MetaClass::OSMetaClass(char const*,OSMetaClass const*,uint)
lea     rax, off_59550
mov     cs:ZN13OSMetaClass10gMetaClassE, rax ; O00AccelMemory::gMetaClass
pop     rbp
ret
GLOBAL _sub_1_10AccelMemory_cpp endp```

X0, Hunk_FFFFF007500C8@PAGE
ADD X0, X0, Hunk_FFFFF007500C8@PAGEOFF
ARDP X1, H10EthernetController@PAGE ; "10EthernetController"
ADD X1, X1, H10EthernetController@PAGEOFF
ARDP X2, Hunk_FFFFF00750298@PAGE
ADD X2, X2, Hunk_FFFFF00750298@PAGEOFF
MOV W3, #0x118
GLOBAL _ZN10OSMetaClassC2EPKcPKS_j I0NetworkingFamily_0_bridge
```

---

rdi/x0: instance of register Meta class
rsi/x1: Meta class name
rdx/x2: instance of parent Meta class
rcx/w3: size of register Meta class instance
Class/Method names re-fine

Re-fine two method table

- instance method table
- meta method table
Connection Type - User Clients

IOService::newUserClient function

- creates an IOUserClient-based connection for communication with a non-kernel client
- invokes this function by calling the IOServiceOpen

```
switch (type) {
    case 0:
        LDWORD(w16) - {int (__fastcall *)(IOGraphicsAccelerator2 *)this->vtable->member 326)(this);
        w5 = -536870918;
        if (w19) {
            w20 = (IOcelIDisplayPipeUserClient2 *)w19;
            w10 = NULL;
            w21 = IOAccelerSurface2::init(w19, mL, w7);
            goto LABEL_28;
        }
        break;
    default:
        LDWORD(w16) - {int (__fastcall *)(IOGraphicsAccelerator2 *, LWORD) this->vt�le - > _2M2100gra
            this, 
            type);
        w5 = -536870918;
        if (w22) {
            w23 = (IOUserClient *)w22;
            if (unsigned _int8)IOAccelerContext2::init(w22, mL, w7)
        goto LABEL_36;
            *(void (__fastcall **))(IOUserClient *, LWORD)(*w23 + mL)[w23, mL];
            w5 = -536870918;
        }
        break;
    case 2u:
        LDWORD(w16) - {int (__fastcall *)(IOGraphicsAccelerator2 *)this->vt�le - > member 327}(this);
```
IOUserClient::externalMethod

- Can be call by is_io_connect_method from user-space
- execute related external method according to selector

SubUserClient::externalMethod() or SubUserClient::getTargetAndMethodForIndex() or SubUserClient::getAsyncTargetAndMethodForIndex()

External methods
How Automation?

**CONNECTION TYPE - USER CLIENTS**

- Find the `<connection type – user client>` tuple, like

```c
enum {
    // connection types for IOSTateServer
    kIOFSERVERConnectType = 6,
    kIOFSERVERSharedConnectType = 1,
    kIOFSDiagnoseConnectType = 2,
};
```

**USER CLIENT EXTERNAL METHODS**

- Find or construct a external methods dispatch table, like

<table>
<thead>
<tr>
<th>connection type</th>
<th>selector</th>
<th>scalarInputCount</th>
<th>structureInputSize</th>
<th>scalarOutputCount</th>
<th>structureOutputSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0x1f0000</td>
<td>0</td>
<td>0x70</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>connection type</th>
<th>selector</th>
<th>scalarInputCount</th>
<th>structureInputSize</th>
<th>scalarOutputCount</th>
<th>structureOutputSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0x1f0000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0x1f0000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0xc</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0x400</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Connection Type - User Clients

Automation Methodology

- Locate the `newUserClient` function address in the driver
- Analyze the ASM instructions to enumerate the connection types
- Analyze the ASM instructions to get the corresponding user client for each connection type

```
do {
  if (type == kIOHIDParamConnectType) {
    if (eventsOpen) {
      newConnect = new IOHIDParamUserClient;
    } else {
      err = kIOKReturnNotOpen;
      break;
    }
  } else if (type == kIOHIDServerConnectType) {
    newConnect = new IOHIDUserClient;
  } else if (type == kIOHIDStackShotConnectType) {
    newConnect = new IOHIDStackShotUserClient;
  } else if (type == kIOHIDEventSystemConnectType) {
    newConnect = new IOHIDEventSystemUserClient;
  } else {
    err = kIOKReturnUnsupported;
  }
```
User Client External Methods – Graceful

Defined as Global or Local Constant Array
- `externalMethod`
- OR
- `getMethodAndTargetForIndex`
- `getAsyncTargetAndMethodForIndex`
User Client External Methods – Graceful

Automation Methodology

- Locate the starting address for each constant array in the symbol table
- Parse the contents according to the IOExternalMethodDispatch or IOExternalMethod structure from starting address

```c
struct IOExternalMethod {
    IOService * object;
    IOMethod   func;
    IOOptionBits flags;
    IOByteCount count0;
    IOByteCount count1;
};

struct IOExternalAsyncMethod {
    IOService * object;
    IOAsyncMethod func;
    IOOptionBits flags;
    IOByteCount count0;
    IOByteCount count1;
};

struct IOExternalMethodDispatch {
    IOExternalMethodAction function;
    uint32_t    checkScalarInputCount;
    uint32_t    checkStructureInputSize;
    uint32_t    checkScalarOutputCount;
    uint32_t    checkStructureOutputSize;
};
```
User Client External Methods - Ugly

Automation Methodology

- Locate the address of override externalMethod/getTarget.../getAsycTarget... Function
- Analyze the ASM instructions to get selector and external methods
Automation Implementation

- Locate the Address of Key Const Array and Parse their Content
  - For global const external method dispatch
  - For local const external method dispatch

- Analyze the ASM Instructions
  - For newUserClient method
  - For externalMethod/getTar.../getAsynTar... method
Parse the External Method Dispatch Array

Global/Local Const Array

- Parse "Symbol Table" section
- Search Constant Array name, shown as "String Table Index"
  - Start with "__ZN" or "__ZN"
- Locate the address, shown as "value"
Analyze the ASM Instructions

Methods Compare

**Angr/Miasm**
- Support binary to ASM
- Support ASM CG/CFG
- Support Emulating using JIT

**Start From Scratch**
- Can only Simulate part of Instruction Operation
- Can only care about certain registers’ data flow

- Need to construct a kext analysis engine
- Need disassembling, CG/CFG
- Need to vm address operation

...
Analyze the ASM Instructions

What we want?

If

else if

else if

else if

else if

...

Switch

case

External method 1

case

External method 2

case

External method 3

case

External method 4

...

...
Analyze the ASM Instructions

- Combination Method

- Custom KEXT Analysis Engine
- CG/CFG based on Miasm
- Custom Instructions Emulator
- Attack Surfaces
Custom KEXT Analysis Engine

- Init a MachO structure handler
- Parse key sections and segments
- Vm addr and file addr operation
- Vm memory operation based on vm and file addr
Custom KEXT Analysis Engine
CG/CFG BASED ON MIA SM

- Init a disassembling engine, like capstone
- Init Miasm machine
- Generate CFG local information

https://github.com/cea-sec/miasm
Generate CFG local information

AppleHDAEngine::newUserClient

loc_2C1F1:
    mov    r12d, 0E00000C2h
    cmp    r14d, 3
    jz     short loc_2C235
    cmp    r14d, 2
    jz     loc_2C284
    test   r14d, r14d
    jnz    loc_2C354
    mov    rax, es:off_920E0h
    xor    ecx, ecx
    mov    rdi, r13
    mov    rsi, r15
    mov    rdx, [rbp-40h]
    mov    r8, [rbp-30h]
    call   qword ptr [rax+788h]
    mov    r12d, eax
    jmp    loc_2C354

; C CODE XREF: AppleHDA

loc_2C235:
    lea    rax, __ZN24AppleHDAEngineUserClient9
    mov    rdi, [rax]
    mov    rax, [rdi]
    call   qword ptr [rax+88h]
Analyze key paths based on CFG

Key Paths based on Key registers
- RCX register in “newUserClient” function
- RSI register in “externalMethod” function
- Tracking data flow between registers, as shown below, RCX move to R14D register

AppleHDAEngine::newUserClient

```
loc_87
MOV    R12D, 0xE088002C2
CMP    R14D, 0x3
JZ     loc_cb
->    c_next:loc_93  c_to:loc_cb

loc_cb
LEA    RAX, QWORD PTR [RIP + 0x6FE54]
MOV    RDI, QWORD PTR [RAX]
MOV    RSI, R15
CALL   loc_155

loc_14a
CMP    R14D, 0x2
JZ     loc_14a
->    c_next:loc_9d  c_to:loc_14a

loc_155
MOV    RDI, R13
XOR    ECX, ECX
MOV    RDI, R13
MOV    RSI, R15
MOV    RDX, QWORD PTR [RBP + 0xFFFFFFFFFC0]
MOV    RS, QWORD PTR [RBP + 0xFFFFFFFFFD8]
CALL   QWORD PTR [RAX + 0x788]
```
Automation Practice

CUSTOM INSTRUCTIONS EMULATOR

- Analysis key functions based on CFG
- Emulate key instructions operation
- Output User Client or external method information
Emulate key instructions operation

- **ARM Emulator**
  - adrp/adr, add, mov/movz, orr, ldr, bl...

- **X86_64 Emulator**
  - lea, mov, call, cmp, jz, je...

```cpp
if not cmp(mnemonic, "blt"):
    reg_num = insn.op_count(IS_OP_REG)
    if reg_num == 1:
        continue
    f_reg = get_first_reg(insn)
    if f_reg == ARM64_CONST.ARM64_REG_X2R or f_reg == ARM64_CONST.ARM64_REG_D0 or:
        f_reg = arm64_const.ARM64_REG_X2R:
        continue
    s_reg = get_second_reg(insn)
    if s_reg:
        s_reg_v = get_actual_value_by_reg(s_reg)
        if not (s_reg_v and s_reg_v == meta_class.class_self_addr):
            continue
    else:
        continue
    f_reg_v_vm = get_actual_value_by_reg(f_reg)
    if iskext:
        f_reg_v_f = k_header.get_prelink_from_vm(f_reg_v_vm)
    else:
        f_reg_v_f = k_header.get_f_from_vm(each_mif_f, each_mif_vm, f_reg_v_vm)
    parse_func(k_header, meta_class, f_reg_v_vm, f_reg_v_f, iskext)
```
Emulate register operation

```python
def get_single_IMM(insn):
    seg_num = insn.op_count(CS_OP_IMM)
    if seg_num > 1:
        print "Extract: too much imm reg!"
    if seg_num != 1:
        print "Extract: no imm reg found!"
    return to_x(insn.op_find(CS_OP_IMM, 1).value.imm)

def get_mem_op_offset(insn):
    mem_num = insn.op_count(CS_OP_MEM)
    if mem_num >= 1:
        offset = insn.op_find(CS_OP_MEM, 1).mem.disp
        return offset

def get_mem_op_reg(insn):
    mem_num = insn.op_count(CS_OP_MEM)
    if mem_num >= 1:
        offset = insn.op_find(CS_OP_MEM, 1).mem.base
        return offset

def get_first_reg(insn):
    return insn.op_find(CS_OP_REG, 1).value.reg

def get_second_reg(insn):
    return insn.op_find(CS_OP_REG, 2).value.reg
```

```python
class x_reg_manager(object):
    def __init__(self):
        self.x = [1]*256
        for i in range(256):
            self.x[i] = 0

    def get_actual_value_by_regN(self, reg):
        # global x0
        return self.x[reg]

    def set_actual_value_by_regN(self, reg, reg_val):
        self.x[reg] = reg_val
```

```python
from capstone import x86_const
```
Output User Client or external method information

AppleHDAEngine::newUserClient

<table>
<thead>
<tr>
<th>index</th>
<th>CanOpen</th>
<th>TOpenType</th>
<th>ServiceName</th>
<th>extends</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>True</td>
<td>8</td>
<td>AppleHDAEngineOutput</td>
<td>IOAudioEngine::gMetaClass-&gt;AppleHDAEngine-&gt;AppleHDAEngineOutput</td>
</tr>
<tr>
<td>86</td>
<td>True</td>
<td>0</td>
<td>AppleHDAEngine</td>
<td>IOAudioEngine::gMetaClass-&gt;AppleHDAEngine</td>
</tr>
</tbody>
</table>

ServiceName | OpenType | UserClient
---|----------|---------------------
AppleHDAEngine | 0x3 | AppleHDAEngineUserClient::metaClass
AppleHDAEngine | 0x2 | DspFuncUserClient::Create(IOAudioEngine*, task*)

AppleHDAEngineUserClient::externalMethod

| selector | cSIC | cSIS  | cSOC  | cSOS  | func_name
|----------|------|-------|-------|-------|---------------------
| 0        | 2    | 0     | 0     | 4895  | AppleWebKitUserClient::getState
| 1        | 2    | 4895  | 0     | 0     | AppleWebKitUserClient::setState
| 2        | 0    | 0     | 0     | 0     | AppleWebKitUserClient::resetDSPToPropertyList
| 3        | 1    | 0     | 1     | 0     | AppleWebKitUserClient::isPortPresent
| 4        | 0    | 0     | 6     | 0     | AppleWebKitUserClient::getHardwareVolume
| 5        | 1    | 0     | 0     | 16    | AppleWebKitUserClient::getActiveSpatialChannels
| 6        | 0    | 0     | 3     | 0     | AppleWebKitUserClient::getAudioSnoopEnabled
| 7        | 3    | 0     | 0     | 0     | AppleWebKitUserClient::setAudioSnoopEnabled
| 8        | 2    | 0     | 0     | 0     | WebViewUserClient::setSpatialChannelMute

Process finished with exit code 0
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
Kernel Interfaces

Implementation files in XNU source code

```
BUILD
├── OBJ
│   ├── DEBUG_X86_64
│       ├── bsd
│       │   └── DEBUG
│       │       └── init_sysent.c
│   └── EXPORT_HDRS
│       ├── bsd
│       │   └── sys
│       │       ├── syscall.h
│       │       └── sysproto.h
│       └── systm.h
├── -- osfmk
│   ├── -- device
│       │   └── device.defs //mig
│   ├── -- kern
│       │   └── syscall_sw.c //traps
│       └── -- mach
│           └── mach_traps.h //traps
```
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
Kernel Diff Methodology

Step 1: diff above files in coarse-grain for getting the newly interfaces

Step 2: diff XNU files in and tag those changed or newly added files

Step 3: Tag those changed or newly added functions

Step 4: construct CG (calling graph) and get the entry function

Step 5: save those entry functions that in above interface vector
Kernel Diff Analysis Practice (1/2)

Updated Function "getvolattrlist ()" in vfs_attrlist.c

- Diff kernel source code files and find the difference functions
- Meld: A open source visual diff and merge tool
- CVE-2018-4243 Patch

https://github.com/GNOME/meld
Kernel Diff Analysis Practice (2/2)

getvolattrlist() for example

- P-diff: A tool Implemented by IDA pro script
  - Construct CG for getvolattrlist function
  - List and report calling sequence that entry function in attack vector
  - CodeRefsTo(ea, flow) and CodeRefsFrom(ea, flow)

```c
/* P-diff calling sequence 0 */: _getvolattrlist() < _getattrlist_internal() < _getattrliststat_internal() < _getattrliststat()
/* P-diff calling sequence 1 */: _getvolattrlist() < _getattrlist_internal() < _getattrlist()
/* P-diff calling sequence 2 */: _getvolattrlist() < _getattrlist_internal() < readdirattr() < _getattrlistbulk()
/* P-diff calling sequence 3 */: _getvolattrlist() < _getattrlist_internal() < _getattrliststat_internal() < _getattrlist()
```

```
220 AUE_GETATTRLIST ALL { int getattrlist(const char *path, struct attrlist *alist, void *attributeBuffer, size_t bufferSize, u_long options) NO_SYSCALL_STUB; }
461 AUE_GETATTRLISTBULK ALL { int getattrlistbulk(int dirfd, struct attrlist *alist, void *attributeBuffer, size_t bufferSize, uint64_t options); }
228 AUE_FGETATTRLIST ALL { int fgetattrlist(int fd, struct attrlist *alist, void *attributeBuffer, size_t bufferSize, u_long options); }
476 AUE_GETATTRLISTAT ALL { int getattrlistat(int fd, const char *path, struct attrlist *alist, void *attributeBuffer, size_t bufferSize, u_long options); }
```
KEXTs Diff Analysis

Difference with Kernel and KEXTs

- KEXTs are closed source
- Using IDA pro script with plugin "Bindiff"

```
P-Diff: entry functions
AppleHDAM::createVolumeAndmuteControlsForActivePathSet()
AppleHDAM::handlePowerStateChange()
AppleHDAM::resetDSPFileList()
AppleHDAM::input::init(NSArray, NSDictionary, NSDictionary, NSArray)
AppleHDAM::input::init_audioStream()
AppleHDAM::input::performFormatChange(IOAudioStream, IOAudioStreamFormat, IOAudioSampleRate)
AppleHDAM::output::protectedChangePathSet(IOAudioControl)
AppleHDAM::output::init(NSArray, NSDictionary, NSDictionary, NSArray)
AppleHDAM::output::protectedChangePathSet(IOAudioControl)

AppleHDAM::userClient::getStateAction(UserClientData)
AppleHDAM::userClient::setHardwareVolume()
AppleHDAM::userClient::setStateAction(UserClientData)
```

```
0.67  0.69  G------  0000000000000000  AppleHDATM_CS42L33::getSampleRate()()
0.62  0.96  G-------  0000000000000000  AppleHDATM_CS42L33::getSampleRate()()
0.59  0.98  G-------  0000000000000000  AppleHDATM_CS42L33::setSampleRateForDeviceChannel()
0.57  0.67  G---EL  0000000000000000  AppleHDATM_MYMpTa5576L::createHandler()
0.56  0.75  GI-LE--  0000000000000000  AppleHDAPath::sWidgetAmplifierGainAdjustable()
0.53  0.73  --E---  0000000000000000  AppleHDAPath::sWidgetAmplifierGainAdjustable()
0.52  0.63  G---EL  0000000000000000  AppleHDA::createHandler()
```

Similarity
Disadvantages about KEXTs static analysis

- KEXTs are closed source, many method strings are stripped
- Function call usually use *(object_ptr + offset) type
  - difficult to construct CFG, which bring noise for KEXTs vector analysis
  - difficult to construct CG, which bring noise for KEXTs diff analysis

```c
u20 = (*(int *)__fastcall **)(IORegistryEntry *, __int64, AMDRadeonX4XXX_AMDAccelResource *, __DWORD, __DWORD, __DWORD)(*(__DWORD *)(this_ptr + 0x870LL))(  
  this_ptr,  
  u2,  
  accelResource_offset8,  
  0LL,  
  *(&__DWORD *)(this_ptr + 594),  
  0LL);  

  // AMDRadeonX4XXX_AMD3DGLOpenGLContext::bindResource(I0AccelCommandStreamInfo &, I0AccelResource2 &, bool, I0AccelChannel2 &)
```
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
## Comparison of dynamic trace

<table>
<thead>
<tr>
<th></th>
<th>User Trace</th>
<th>Kernel Trace</th>
<th>Embedded in OS</th>
<th>Any priviledge?</th>
<th>Support script?</th>
<th>Performance</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frida</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Root or Repack</td>
<td>Yes</td>
<td>Middle</td>
<td>iOS/Osx</td>
</tr>
<tr>
<td>Dtrace</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Root</td>
<td>Yes</td>
<td>High</td>
<td>Osx</td>
</tr>
<tr>
<td>lldb</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Root</td>
<td>Yes</td>
<td>Low</td>
<td>iOS/Osx</td>
</tr>
<tr>
<td>Kernel hook</td>
<td>---</td>
<td>Yes</td>
<td>No</td>
<td>Root</td>
<td>No</td>
<td>Middle</td>
<td>Osx</td>
</tr>
</tbody>
</table>
Frida Hook in User Mode

1. Set up frida trace
   - iOS/MacOS real machine
   - Frida-server-10.x.x-ios-armX
   - FridaGadget.dylib
   - Fria-Server
   - Mach_Msg
   - IOConnectMethod
   - Syscall
   - Kernel
   - [2] Behavior trace in iOS/MacOS
   - USB/Wifi/File/Pipe

2. Behavior trace in iOS/MacOS
   - Host (e.g. MacOS)
   - iokit!* .js
   - syscall!* .js
   - fs!* .js ...

3. Trace log collect
   - Frida Controller
   - *.py
   - LogStream
xpc_connection_send_message API context (e.g.)

{"time":"2017-09-18T10:38:32.807Z","txnType":"moony?","lib":"libxpc.dylib","method":"xpc_connection_send_message","artifact":[
    {
      "name":"connection",
      "value":"0x1658d090",
      "argSeq":0,
    },
    {
      "name":"connectioninfo",
      "value":"\tconnection=0x1658d090\tconnectionName=\tconnectionPid=2312\tconnectionProcName=Preferences",
      "argSeq":0,
    },
    {
      "name":"retval","value":374477440,"argSeq":-1
    }]
}
Hunt more dynamic relation if you like

(e.g. profile install)
Dtrace introduction

Source: Solaris Dynamic Tracing Guide
The typical interface for fuzzing:

- system call,
- IOKit,
- mach msg,
- network,
- Disk,
- XPC

```
$ dtrace -l | more

<table>
<thead>
<tr>
<th>ID</th>
<th>PROVIDER</th>
<th>MODULE</th>
<th>FUNCTION</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>dtrace</td>
<td></td>
<td></td>
<td>BEGIN</td>
</tr>
<tr>
<td>2</td>
<td>dtrace</td>
<td></td>
<td></td>
<td>END</td>
</tr>
<tr>
<td>4</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock</td>
<td>adaptive-acquire</td>
</tr>
<tr>
<td>5</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock</td>
<td>adaptive-spin</td>
</tr>
<tr>
<td>6</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock</td>
<td>adaptive-block</td>
</tr>
<tr>
<td>7</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_try_lock</td>
<td>adaptive-acquire</td>
</tr>
<tr>
<td>8</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_try_lock</td>
<td>adaptive-spin</td>
</tr>
<tr>
<td>9</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock</td>
<td>adaptive-block</td>
</tr>
<tr>
<td>10</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_unlock</td>
<td>adaptive-release</td>
</tr>
<tr>
<td>11</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_ext_lock</td>
<td>adaptive-acquire</td>
</tr>
<tr>
<td>12</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_ext_lock</td>
<td>adaptive-spin</td>
</tr>
<tr>
<td>13</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_ext_lock</td>
<td>adaptive-block</td>
</tr>
<tr>
<td>14</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_ext_unlock</td>
<td>adaptive-release</td>
</tr>
<tr>
<td>15</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock.spin</td>
<td>adaptive-acquire</td>
</tr>
<tr>
<td>16</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_mtx_lock.shared</td>
<td>rw-acquire</td>
</tr>
<tr>
<td>17</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_lock.shared</td>
<td>rw-block</td>
</tr>
<tr>
<td>18</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_lock.exclusive</td>
<td>rw-acquire</td>
</tr>
<tr>
<td>19</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_lock.exclusive</td>
<td>rw-block</td>
</tr>
<tr>
<td>20</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_lock.exclusive</td>
<td>rw-spin</td>
</tr>
<tr>
<td>21</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_done</td>
<td>rw-release</td>
</tr>
<tr>
<td>22</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_try_lock.shared</td>
<td>rw-acquire</td>
</tr>
<tr>
<td>23</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_trlock.exclusive</td>
<td>rw-acquire</td>
</tr>
<tr>
<td>24</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_shared_to_exclusive</td>
<td>rw-upgrade</td>
</tr>
<tr>
<td>25</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_shared_to_exclusive</td>
<td>rw-spin</td>
</tr>
<tr>
<td>26</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_shared_to_exclusive</td>
<td>rw-block</td>
</tr>
<tr>
<td>27</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_rw_exclusive_to_shared</td>
<td>rw-downgrade</td>
</tr>
<tr>
<td>28</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_spin_lock</td>
<td>spin-acquire</td>
</tr>
<tr>
<td>29</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_spin_lock</td>
<td>spin-spin</td>
</tr>
<tr>
<td>30</td>
<td>lockstat</td>
<td>mach_kernel</td>
<td>lck_spin_unlock</td>
<td>spin-release</td>
</tr>
<tr>
<td>31</td>
<td>profile</td>
<td></td>
<td></td>
<td>profile-97</td>
</tr>
<tr>
<td>32</td>
<td>profile</td>
<td></td>
<td></td>
<td>profile-199</td>
</tr>
<tr>
<td>33</td>
<td>profile</td>
<td></td>
<td></td>
<td>profile-499</td>
</tr>
<tr>
<td>34</td>
<td>profile</td>
<td></td>
<td></td>
<td>profile-997</td>
</tr>
<tr>
<td>35</td>
<td>profile</td>
<td></td>
<td></td>
<td>profile-1999</td>
</tr>
</tbody>
</table>
```
Dtrace script (e.g. file probe)

```c
#include <dtrace.h>

dtrace::BEGIN
{
    printf("Tracing... Hit Ctrl-C to end.\n");
}

/* save time at start */
io::wait-start
{
    self->start = timestamp;
}

/* process event */
io::wait-done
/self->start/
{
    /*
    * wait-done is used as we are measuring wait times. It also
    * is triggered when the correct thread is on the CPU, obviating
    * the need to link process details to the start event.
    */
    this->elapsed = timestamp - self->start;
    @files[pid, execname, args[2]->fi_pathname] = sum(this->elapsed);
    self->start = 0;
}

/* print report */
dtrace::END
{
    normalize(@files, 1000);
    printf("%s %12s %8s %s\n", "PID", "CMD", "TIME", "FILE");
    printa("%6d %12.12s %08d %s\n", @files);
    
    /*
    * print summary:
    */
    printf("Summary\n");
    printf("%8s %8s %8s %8s\n", "PID", "CMD", "TIME", "FILE");
    printa("%6d %12.12s %08d %s\n", @files);
    
    /*
    * print details:
    */
    printf("Details\n");
    printf("%8s %8s %8s %8s\n", "PID", "CMD", "TIME", "FILE");
    printa("%6d %12.12s %08d %s\n", @files);
}``
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
Enhanced kernel fuzz

I. 3D online games (OpenGL, Unreal game engine,)
II. Peripheral devices operation (e.g., wifi, bluetooth)
III. IOKit services matching and other operation
IV. Font render
V. Other scenario you collected

[1] Call kernel interface API

User mode

Kernel mode

[2] Intercept the kernel counterpart API and fuzz data

[2] Catch first crash spot and analysis

make SDKROOT=macosx ARCH_CONFIGS=X86_64 KERNEL_CONFIGS="KASAN"
KASAN in iOS/OSX Kernel

make SDKROOT=macosx ARCH_CONFIGS=X86_64
KERNEL_CONFIGS="KASAN"

/System/Library/Kernels/kernel*
Future plan

- **Syzkaller-like Fuzzing in Kernel Mode**
  
  There would be existing many environment preparation or initialization (e.g. Open correct service, initialize the target devices and send the correct mach message id) before one special kernel API (e.g. IOConnectionCallMethod) could work properly in kernel.

  So why do not we intercept the kernel API at the proper time under proper state and fuzz it like AFL in kernel mode directly?

- **Porting KASAN/KMSAN for close-source driver**
  
  Modifying the memory API to asan_* API in import table in driver module, or patch the code with memory management in driver to support kernel sanitizer could be as further research.
Agenda

- About US
- Solution Overview
- Static Analysis for Kernel and KEXTs Attack Interface
  - KEXTs Interfaces Vector Automatic Generate
  - Kernel Interfaces Vector Automatic Generate
  - Kernel/KEXTs Interfaces Diff Analysis
- Dynamic Analysis for Kernel and UserMode Attack Interface
- Automatic Fuzzing solution
- 0Day vulnerabilities found
CVE-2018-4462

Integer overflow vulnerability in AMDFramebuffer driver
CVE-2018-4462 – Root Cause

Frame #0: 0xffffffff8d91e324 AMDFramebuffer'AMDFramebuffer::getPixelInformationFromTiming(AtiDetailedTimingInformation const&, IOPixelInformation*, int, int) + 388
AMDFramebuffer'AMDFramebuffer::getPixelInformationFromTiming:
- 0xffffffff8d91e324 <+388>: movq (%rcx,%rdi,8), %rcx
  0xffffffff8d91e328 <+392>: movq %rsi, %rdi
  0xffffffff8d91e32b <+395>: movq %rcx, %rsi
  0xffffffff8d91e32c <+398>: callq 0xffffffff8cbce0 ; Utilities::str_copy(char*, char const*, unsigned long)
(lldb) register read rcx
  rcx = 0xffffffff8d926030 AMDFramebuffer::getPixelInformationFromTiming(AtiDetailedTimingInformation const&, IOPixelInformation*, int, int)::PIXEL_ENCODING
(lldb) register read rdi
  rdi = 0xffffffff2000001

unsigned int v10; // [sp+00h] [bp-2bh]01
uint v11; // [sp+01h] [bp-2ah]01
void *v12; // [sp+02h] [bp-2ch]01
void *v13; // [sp+03h] [bp-2dh]01
__int64 v14; // [sp+04h] [bp-2eh]01
unsigned int int15; // [sp+05h] [bp-2fh]01

v14 = a1;
v13 = a2;
v12 = a3;
v11 = a4;
v10 = a5;
bzero(a3, ebx); if ((signed int)v10 <= 2) {

check

move the value(0xf2000001) of rbp+var_28 to rdi, and
extends it to 64 bits. then becomes 0xffffffff2000001

use

rdi*8 becomes so big , crash point
may be leak info if craft the value of rdi
however, if we craft the rdi value, the rdi*8 can be control by user
OOB read in AMDRadeonX4000 Extension

OOB Read Vulnerability Found in AMDRadeonX4000_AMDAccelResource Initialize Process

* thread #1, stop reason = signal SIGSTOP
* frame #0: 0xffff7fa0965d3 AMDRadeonX4000_AMDAccelResource::initialize(IOAccelNewResourceArgs*, unsigned long long) + 1525
  frame #1: 0xffff7f9fca546b IOAcceleratorFamily2::IOAcceleratedUserClient2::new_resource(IOAccelNewResourceArgs*, IOAccelNewResourceReturnData*, unsigned long long, unsigned int*) + 1893
  frame #2: 0xffff7f9fca4a41 IOAcceleratorFamily2::IOAcceleratedUserClient2::s_new_resource(IOAccelSharedUserClient*, void*, IOExternalMethodArguments*) + 151
  frame #3: 0xffff801d5cb8 kernel.development::IOUserClient::externalMethod(0000000000000000, selector=<unavailable>, args=0xffff83dd4b3b58, dispatch=0xffff7f9fee8260, target=0xffff8054df780, reference=0x0000000000000000) at IOUserClient.cpp:3538 [opt]
  frame #4: 0xffff7f9fca4d98 IOAcceleratorFamily2::IOAcceleratedUserClient2::externalMethod(unsigned int, IOExternalMethodArguments*, IOExternalMethodDispatch*, OSObject*, void*) + 120
  frame #5: 0xfffff880e0d6ef7f kernel.development::is_io_connect_method(connection=0xffff8054df780, selector=0, scalar_input=<unavailable>, scalar_output=<unavailable>, inband_input=<unavailable>, inband_output=0ffffff8066a03e0c, scalar_output=0xfffff83dd4b3ce0, scalar_outputCnt=0xfffff83dd4b3ce0, oo1_output=0, oo1_output_size=0xfffff80591d5c) at IOUserClient.cpp:3994 [opt]
  frame #6: 0xfffff801cfbb34 kernel.development::Xio_connect_method(InHeadP=<unavailable>, OutHeadP=0xfffff806ba03de0) at device_server.c:8379 [opt]
  frame #7: 0xfffff801c8d27d kernel.development::ipc_kobject_server(request=0xfffff805919000, option=<unavailable>) at ipc_kobject.c:359 [opt]
  frame #8: 0xfffff801c59465 kernel.development::ipc_kmsg_send(kmsg=0xfffff805919000, option=3, send_timeout=0) at ipc_kmsg.c:1832 [opt]
  frame #9: 0xfffff801ce78a75 kernel.development::mach_msg_overwrite_trap(args=<unavailable>) at mach_msg.c:549 [opt]
  frame #10: 0xffff801c63323 kernel.development::mach_call_munger64(state=0xfffff806ca0c480) at bsd_i386.c:573 [opt]
  frame #11: 0xfffff801ce23486 kernel.development::handle_mach_call64 + 22
OOB read – Root Cause

; CODE XREF: AMDBadEntry::initialize(IAccelNewResourceArgs *,ulong long)+58Dj
mov ecx, [r15+0F8h]
test rex, rex
jz short loc_E603
shl rcx, 3
lea rdi, [rcx+rcx*2]
call _IOMalloc
mov [r12+178h], rax --- rax== buffer address which create by IOMalloc
------ (a)
test rax, rax
jz short loc_E62A
or byte ptr [r12+186h], 8
mov ecx, [r15+0F8h] -------r15==structureInput, ecx=(uint32_t*) structureInput+62) -------(b)
mov [r12+180h], ecx
test rcx, rcx ------- test rcx, if zero, break -------(c)
jz short loc_E639
or edx, edx ------- index -------(d)
mov rsi, [r15+rdx+98h] --- mov structureInput+rdx+0x98 to rsi
mov [rax+rdx], rsi ----move rsi to rax+rdx, rax== buffer address which create by IOMalloc -------(g)
mov [rax+rdx+8], rsi
mov esi, [r15+rdx+0A0h]
mov [rax+rdx+10h], esi
add rdx, 18h -------(e)
dec rex -------decrease rcx value -------(f)
jnz short loc_E5D3
OverFlow in IOUSBFamily Extension

OverFlow bug due to No Boundary Check
OverFlow – Root Cause

frame #7: 0xfffff8004a149a kernel.development`I0BufferMemoryDescriptor::inTaskWithPhysicalMask(inTask=0xfffff80a10ebd88, options=65538, capacity=18446744073709551615, physicalMask=18446744073709547520) at I0BufferMemoryDescriptor.cpp:354 [opt]

18446744073709551615 = 0xffffffffffffffff

(lldb) f 8
I0USBFamily was compiled with optimization - stepping may behave oddly; variables may not be available.
frame #8: 0xfffff7f8529efac I0USBFamily`I0USBInterfaceUserClient::LowLatencyPrepareBuffer(this=0xfffff80a5dbd800, bufferData=0xffffffffa750b2bab0, addrOut=0xffffffffa750b2ba00) at I0USBInterfaceUserClient.cpp:2358 [opt]
(lldb) fr v
(I0USBInterfaceUserClientV3 *) this = 0xfffff80a5dbd800
(LowLatencyUserBufferInfoV3 *) bufferData = 0xffffffffa750b2bab0
(uint64_t *) addrOut = 0xffffffffa750b2ba00
(BOOL) preparedIOMD = false
(uint64_t) dataBuffer = 0
(IOReturn) ret = <variable not available>

(I0USBDevice *) device = <variable not available>
(I0OptionBits) optionBits = 0
(mach_vm_address_t) physicalMask = <variable not available>

(I0BufferMemoryDescriptor *) dataBufferDescriptor = <register rax is not available>
...
...
(lldb) memory read --size 8 --format x --count 12 0xffffffffa750b2bab0
0xffffffffa750b2bab0: 0x0e3834d99444cc1 0x0000000000000049
0xffffffffa750b2bac0: 0xffffffffffffffff 0x0000000000000000
0xffffffffa750b2bad0: 0x0000000000000000 0xfffffffff8004cc0590
0xffffffffa750b2bae0: 0x0000000000000000 0xffffffffa750b2bbac
0xffffffffa750b2baf0: 0xfffffffff8004cc080f 0xffffffff7f852ede68
0xffffffffa750b2bb0: 0xffffffffa750b2bb0 0x0000000000000000
(lldb)
NULL PAGE Reference in IntelAccelerator

NULL PAGE Reference bug found in IntelAccelerator
Pseudo Code snippet of IntelAccelerator::newGTT function

```c
v8 is not always returns normal address, may be null
v8 = (unsigned int *)IntelAccelSysMemory::lockForCPUAccess(
    *(IntelAccelSysMemory **)(v6 + 24),
    *(task **)kernel_task_0,
    t0);

if ( v5 )
    {  
        v9 = this->member547;
        if ( v9 )
            {  
                v10 = this->member546;
                v11 = v8;
                do
                    {  
                        *(unsigned int *)((char *)v8 + (unsigned int)v11) = *(DWORD *)(v10 + v11);
                        v11 = (unsigned int)(v11 + 4);
                    }  
                while ( v9 > v11 );
            }
    }
else
    {  
        use v8 as dst addr
        memcpy(v8, *(const void **)(this->member44 + 616), LOWORD(this->member551) >> 10);  
        IntelAccelerator::releaseGARTMemory(this, LOWORD(this->member552), LOWORD(this->member553), v4);
        IntelAccelerator::releaseGARTMemory(this, LOWORD(this->member554), LOWORD(this->member555), v4);
    }
```

---(a)
Divide Zero in AMDRadeonX4000 Extension

Divide Zero BUG found in IOAccelCommandQueue class

(lldb) bt
* thread #1, stop reason = signal SIGSTOP
  frame #0: 0xffff7f88b04941 AMDRadeonX4000`BltMgr::HwlOptimizeBufferBltRects(BltInfo*, unsigned int) + 879
  frame #1: 0xffff7f88b1e47 AMDRadeonX4000`SiBltMgr::Adjust3dBltInfo(BltInfo*) + 662
  frame #2: 0xffff7f88b1be2 AMDRadeonX4000`SiBltMgr::Execute3dBlt(BltInfo*) + 76
  frame #3: 0xffff7f88b04241 AMDRadeonX4000`BltMgr::Memset(BltDevice*, _UBM_MEMSETINFO*) + 753
  frame #4: 0xffff7f88a75506 AMDRadeonX4000`AMDRadeonX4000_AMDAtomicBlitManager::doMemset(_UBM_MEMSETINFO*, ABM_OPTIONS const*) + 28
  frame #5: 0xffff7f88a73d29 AMDRadeonX4000`AMDRadeonX4000_AMDAtomicBlitManager::doMemset(_UBM_MEMSETINFO*, ABM_OPTIONS const*) + 263
  frame #6: 0xffff7f88a7647 AMDRadeonX4000`AMDRadeonX4000_AMDAtomicBlitManager::doMemset(_UBM_MEMSETINFO*, ABM_OPTIONS const*) + 390
  frame #7: 0xffff7f88a769c9 AMDRadeonX4000`AMDRadeonX4000_AMDAtomicBlitManager::prepare() + 108
  frame #8: 0xffff7f889bb73c3e IOAcceleratorFamily2`IOAccelSegmentResourceList::prepare() + 48
  frame #9: 0xffff7f889bc994 IOAcceleratorFamily2`IOAccelCommandQueue::coalesceSegment(IOAccelCommandQueueSegment*, unsigned int*, IOAccelSegmentResourceList*, IOAccelKernelCommand const*, IOAccelKernelCommand const*) + 78
  frame #10: 0xffff7f889b3c1e IOAcceleratorFamily2`IOAccelCommandQueue::processCommandBuffer(unsigned int, unsigned int) + 666
  frame #11: 0xffff7f889b1e88 IOAcceleratorFamily2`IOAccelCommandQueue::process_command_buffer(unsigned int, unsigned int) + 924
  frame #12: 0xffff7f889b4e6 IOAcceleratorFamily2`IOAccelCommandQueue::submit_command_buffer(unsigned int, unsigned int, unsigned long, unsigned long) + 252
  frame #13: 0xffff7f889b4b90 IOAcceleratorFamily2`IOAccelCommandQueue::submit_command_buffers(IOAccelCommandQueue*, void*, IOExternalMethodArguments*) + 827
  frame #14: 0xffff7f889b42c9 IOAcceleratorFamily2`IOAccelCommandQueue::s Submit command buffers(IOAccelCommandQueue*, void*, IOExternalMethodArguments*) + 250
  frame #15: 0xffff8006224978 kernel.development`IOUserClient::externalMethod(this=<unavailable>, selector=<unavailable>, args=0xffffffa753d3b9b8, dispatch=0xffffff7f889fac68, target=<unavailable>, reference=<unavailable>) + 82 at IOUserClient.cpp:5689 [opt]
  frame #16: 0xffff800622da02 kernel.development`is_io_connect_method(connection=<unavailable>, selector=1, scalar_input=<unavailable>, scalar_inputCnt=<unavailable>, inband_input=<unavailable>, inband_inputCnt=32, ool_input=0, ool_input_size=0, inband_output=",", inband_outputCnt=0xffff80b137560c, scalar_output=0xfffffa753d3bce0, scalar_outputCnt=0xfffffa753d3bce0, ool_output=0, ool_output_size=0xffff80b13e3138) at IOUserClient.cpp:4304 [opt]
  frame #17: 0xffff8005b8c386 kernel.development`Xio_connect_method(InHeadP=<unavailable>, OutHeadP=0xfffffa753d3bce0) at device_server.c:8379 [opt]
  frame #18: 0xffff8005a49d4 kernel.development`ipc_kobjc t_server(request=0xffff80b13e050, option=3) at ipc_kobjct.c:361 [opt]
  frame #19: 0xffff8005a608e kernel.development`ipc_kmsg_send(kmsg=0xffff80b13e050, option=3, send_timeout=0) at ipc_kmsg.c:1868 [opt]
  frame #20: 0xffff8005a80e03 kernel.development`mach_msg_overwrite_trap(args=<unavailable>) at mach_msg.c:553 [opt]
  frame #21: 0xffff8005b7073b kernel.development`mach_msg_manger64(state=0xffff80a777100) at bsd_i386.c:580 [opt]
  frame #22: 0xffff8005a2a476 kernel.development`hdl_talent64 + 22
ASM Code snippet of AMDRadeonX4000\`BltMgr::HwlOptimizeBufferBltRects Function

```
__text:00000000000BB75B  div   esi
__text:00000000000BB75D  mov   r14d, 0
__text:00000000000BB763  mov   r12d, 0   -----init r12d with 0   --(a)
__text:00000000000BB769  test  edx, edx

-----omitted code-----

.....

__text:00000000000BB93C  loc_BB93C: ; CODE XREF:
BltMgr::HwlOptimizeBufferBltRects(BltInfo *,uint)+3E1j
__text:00000000000BB93C  xor   edx, edx
__text:00000000000BB93E  mov   eax, r13d
__text:00000000000BB941  div   r12d   -----r12d is not always nonzero   ---(b)
__text:00000000000BB944  cmp   eax, r14d
__text:00000000000BB947  jbe  short loc_BB95B
__text:00000000000BB949  mov   dword ptr [rsi+rbx-0Ch], 0
__text:00000000000BB951  mov   [rsi+rbx-4], r12d
__text:00000000000BB956  mov   eax, r14d
__text:00000000000BB959  jmp  short loc_BB97C
```
https://github.com/dongyangwu/p-joker

new version will release later