PanicXNU 3.0

Who Are We

- Juwei Lin (@panicall)
- Senior Staff Engineer at TrendMicro
- A few CVEs
- CodeBlue2018 & BHEU 2018

TREND MICRO | research

Who Are We

- Junzhi Lu(@pwn_orz)
- Engineer at TrendMicro
- Fewer CVEs

TREND MICRO | research

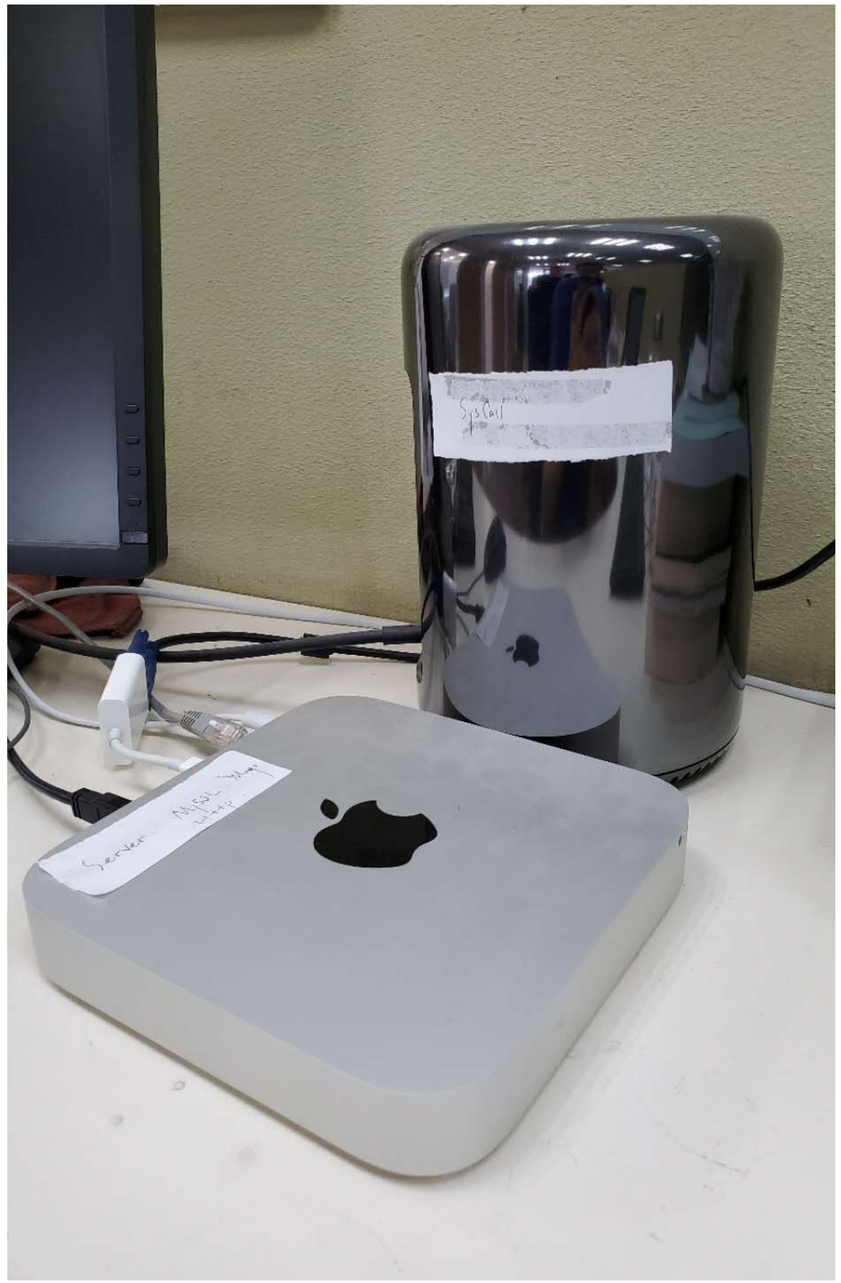
Agenda

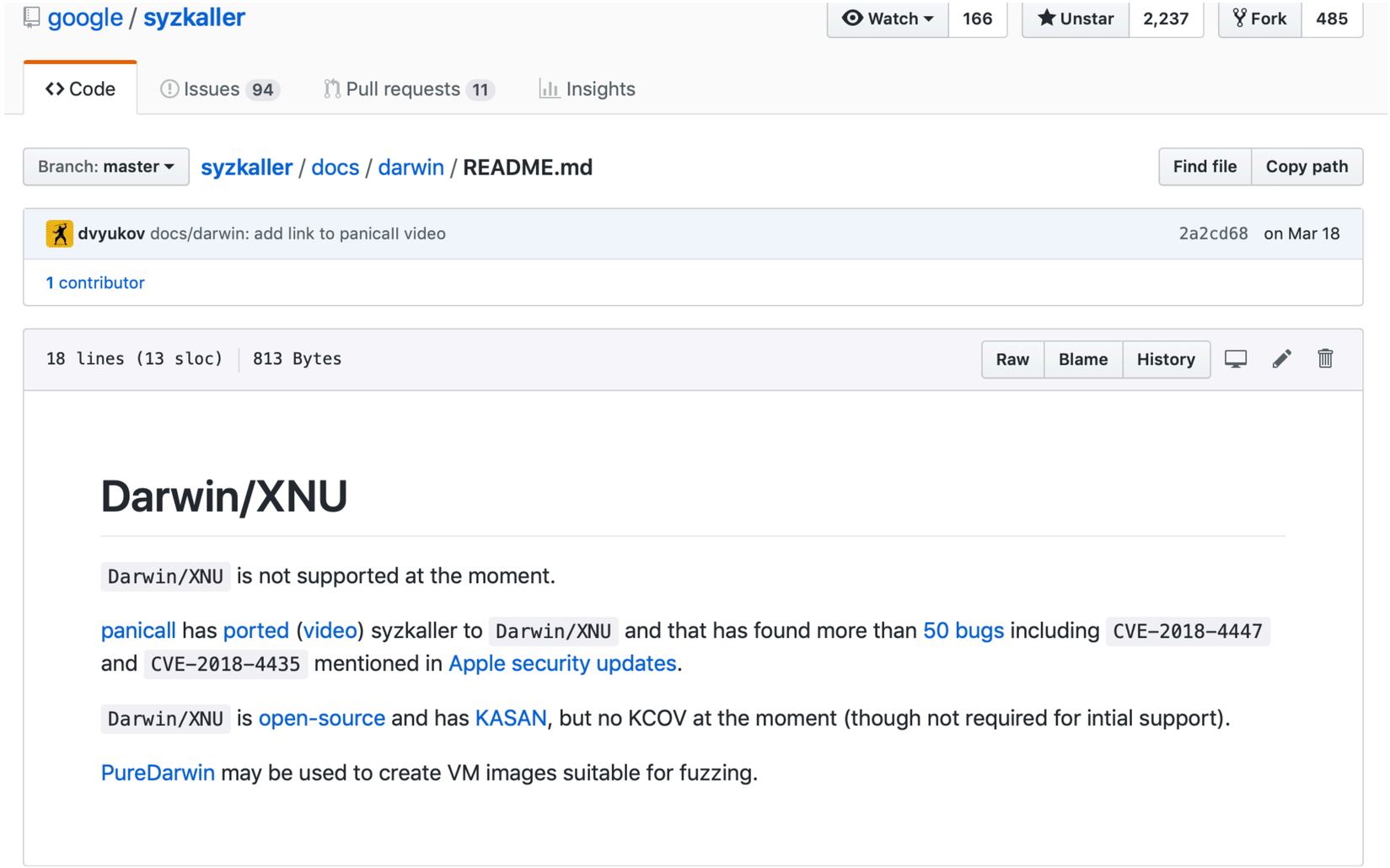
- What is PanicXNU
- PanicXNU 1.0 - 3.0
  - Syscall Fuzzing (PanicXNU 1.0)
  - IOKit Active Fuzzing (PanicXNU 2.0)
  - IOKit Passive Fuzzing (PanicXNU 3.0)
- Fuzzing Visualization and Daily Operation
- Case Study
- Acknowledgements

What is PanicXNU?

1) XNU Fuzzer(syscall, iokit, kext, traps…)

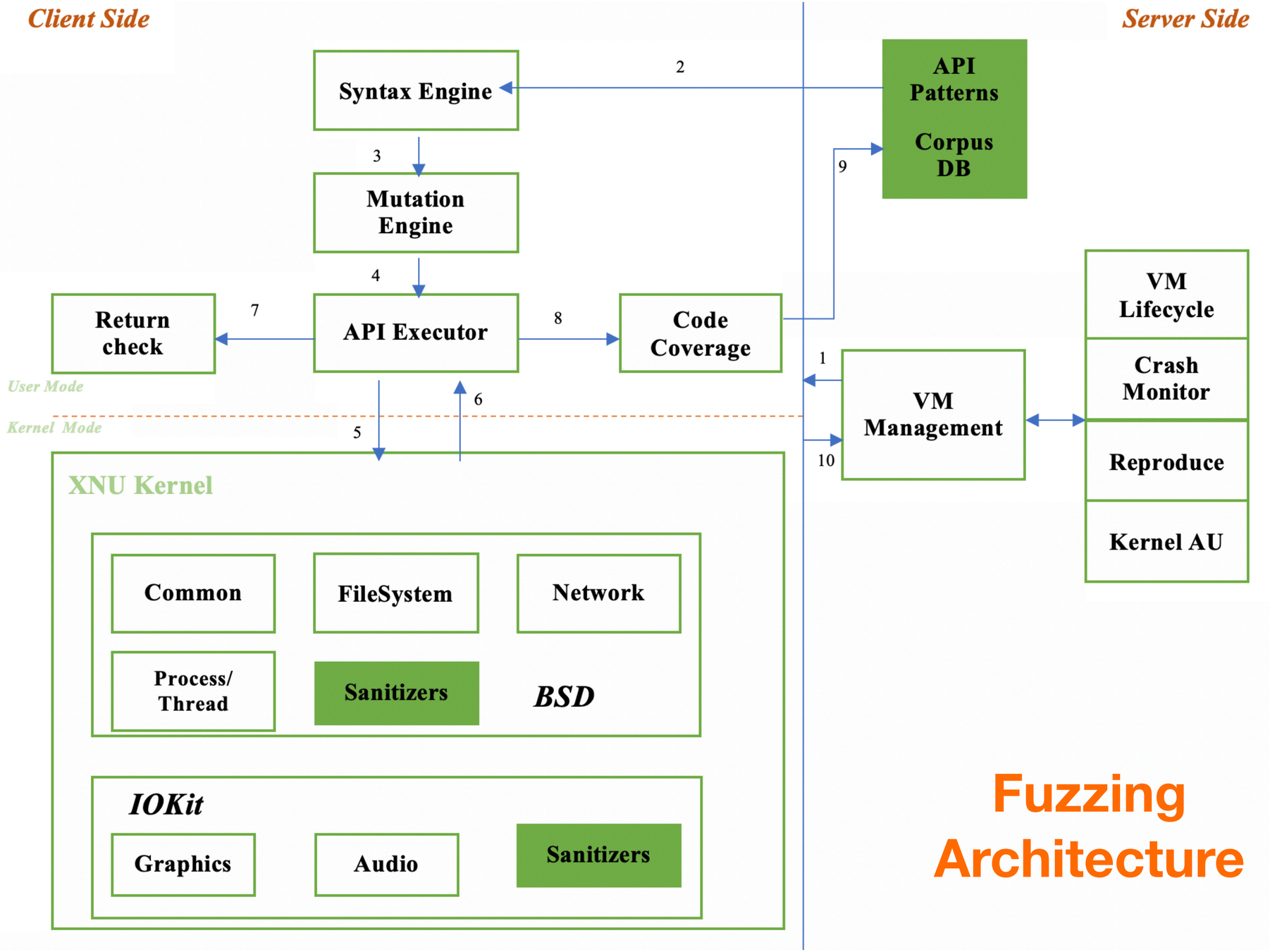
2) Based on **Syzkaller**





https://github.com/google/syzkaller/blob/master/docs/darwin/README.md

This page was created after my #BHEU2018 talk of PanicXNU 1.0;
Today I will talk about PanicXNU 3.0


Client Side
Server Side
Syntax Engine
2
API Patterns
Corpus DB
3
Mutation Engine
9
4
Return check
7
API Executor
8
Code Coverage
VM Lifecycle
Crash Monitor
Reproduce
Kernel AU
1
VM Management
10
User Mode
Kernel Mode
6
5
XNU Kernel
Common
FileSystem
Network
Process/ Thread
Sanitizers
BSD
IOKit
Graphics
Audio
Sanitizers
Fuzzing Architecture

Terminology

- Pattern: API syntax description, e.g. “int open(const char*, int)”
  - We use patterns to describe APIs which can guide generation/mutation fuzzing
  - Use patterns to generate corpus
- Corpus: parameters for the API, e.g. “open(‘/tmp/test’, 0)”
  - The real fuzzing data

Syscall Fuzzing(PanicXNU 1.0)

- Already shared at #BHEU2018 (https://www.blackhat.com/eu-18/briefings/schedule/#drill-apple-core-up-and-down---fuzz-apple-core-component-in-kernel-and-user-mode-for-fun-and-profit-12923)
- Key Idea: 530 API patterns + sanitizers
- Found more than 50 syscall crashes(most of them are not vulnerable) including 2 for Pwn2Own macOS 10.14

Supplements

- How to add support for BSD code coverage?
- How to enable KASAN?

Q1. BSD Code Coverage

- Add Kcov module **kcov.cpp** to /osfmk/kern
  - Implement your own **__sanitizer_cov_trace_pc**
  - Register as a **kext module** to provide coverage data for usermode query
- Update configuration for kcov module
  - Update /config/MASTER to add kcov device: pseudo-device kcov 1 init kcov_init
  - Update /osfmk/conf/files to add kcov: osfmk/kern/kcov.c optional kcov
  - Update /bsd/conf/makefile.template to add sanitizer: add -fsanitize-coverage=trace-pc into existing CFLAGS

Register as a kext module inside osfmk:

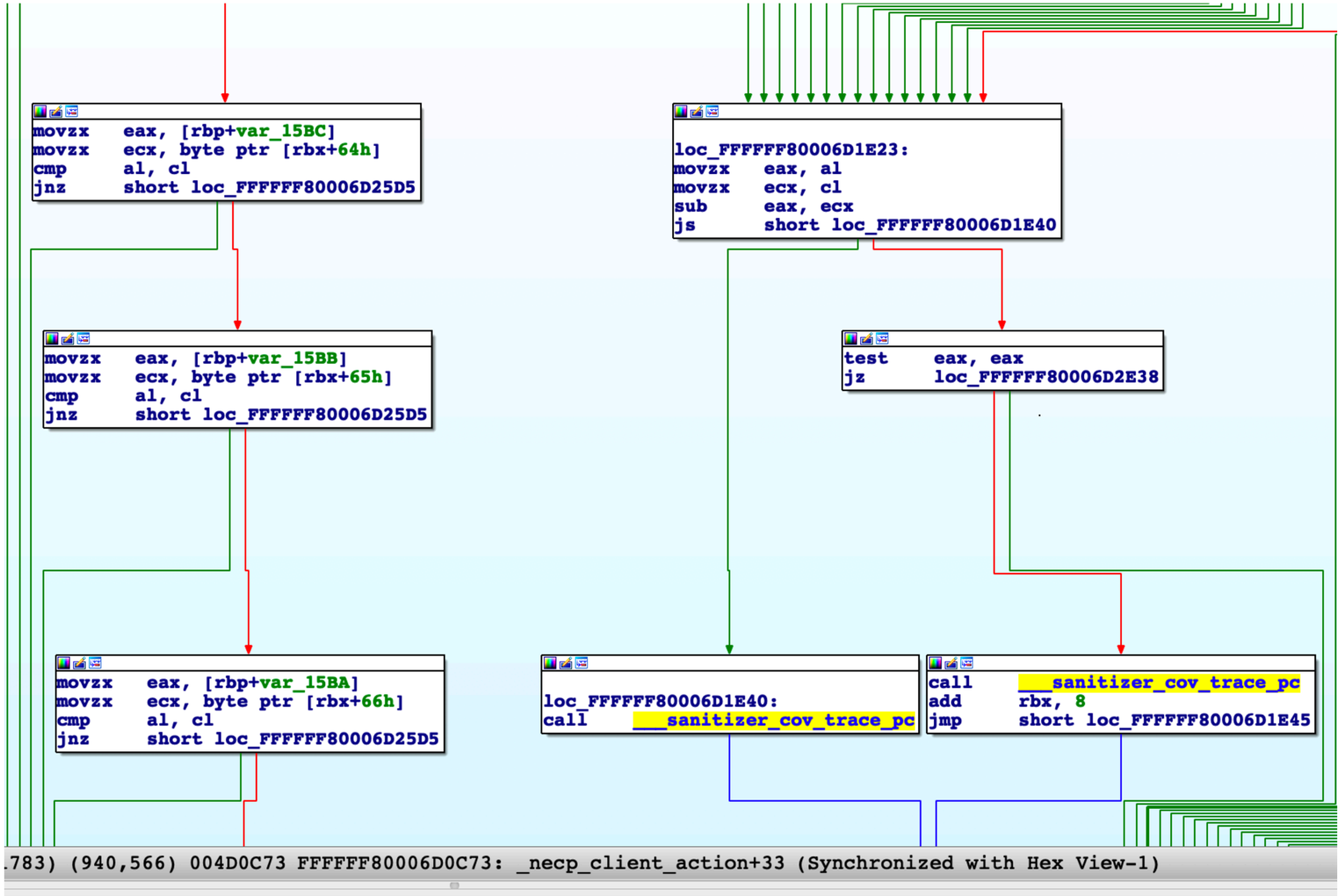
```
static struct cdevsw kcov_cdevsw =
{
    kcov_open,
    kcov_close,
    kcov_read,
    eno_rdwrt,
    kcov_ioctl,
    eno_stop,
    eno_reset,
    NULL,
    eno_select,
    eno_mmap,
    eno_strat,
    eno_getc,
    eno_putc,
    0
};
```

```
void
kcov_init( void )
{
    int ret = 0;

    KCovLockGroup = lck_grp_alloc_init("KCovLockGroup", LCK_GRP_ATTR_NULL);
    if (!KCovLockGroup) return;

    ret = cdevsw_add(-1, &kcov_cdevsw);
    if (ret < 0) {
        panic("kcov_init: failed to allocate a major number!");
    }

    //makedev in osfmk and bsd is different!
    //bsd: ((dev_t)(((x) << 24) | (y)))
    //osfmk: ((dev_t)(((x) << 8) | (y)))
    //devfs_make_node(makedev(ret, 0), DEVFS_CHAR,
    //                UID_ROOT, GID_WHEEL, 0666, "kcov", 0);
    devfs_make_node(ret<<24, DEVFS_CHAR,
                    UID_ROOT, GID_WHEEL, 0666, "kcov", 0);
}
```

Be careful makedev macro in osfmk and bsd is different!

Syscall Code Coverage Problem

- No Problem for Genetic Algorithm during Fuzzing
- But not friendly for researcher to do fuzzing review
  - BasicBlock compilation option is still not good
  - Maybe try SBI or IntelPT?



Q2. Enable KASAN

- Add the following functions into {xnu_root_folder}/san/kasan-blacklist-x86_64

```
fun:machine_trace_thread_get_kva
fun:PHYSMAP_PTOV_check
fun:is_ept_pmap
fun:pmap_pde
fun:pltrace
fun:iv_alloc
fun:iv_dealloc
fun:ivace_release
fun:unsafe_convert_port_to_voucher
fun:convert_port_to_voucher
fun:convert_port_name_to_voucher
fun:ipc_voucher_notify
fun:convert_voucher_to_port
fun:ivac_dealloc
fun:convert_port_to_voucher_attr_control
fun:ipc_voucher_attr_control_notify
fun:convert_voucher_attr_control_to_port
fun:iv_dedup
fun:ipc_register_well_known_mach_voucher_attr_manager
fun:mach_voucher_extract_attr_content
fun:ivace_lookup_values
fun:mach_voucher_extract_attr_recipe
fun:mach_voucher_extract_all_attr_recipes
fun:mach_voucher_debug_info
fun:mach_voucher_attr_command
fun:mach_voucher_attr_control_get_values
fun:ipc_get_pthpriority_from_kmsg_voucher
fun:ipc_voucher_send_preprocessing
fun:ipc_voucher_prepare_processing_recipe
fun:ipc_voucher_receive_postprocessing
fun:mach_generate_activity_id
fun:user_data_release_value
fun:user_data_get_value
fun:user_data_extract_content
fun:ivace_reference_by_index
fun:ipc_replace_voucher_value
fun:ivace_reference_by_value
fun:re_queue_tail
fun:host_request_notification
fun:OSMalloc_Tagalloc
fun:OSMalloc_Tagrele
fun:OSMalloc_Tagfree
fun:enqueue_tail
fun:remque
fun:enable_preemption_internal
fun:entry_queue_change_entry
fun:semaphore_create
fun:thread_exception_enqueue
fun:thread_call_func_cancel
fun:_pending_call_enqueue
fun:waitq_select_one_locked
fun:dequeue_head
fun:do_backtrace32
fun:ccdigest_final
fun:ccctr_setctr
fun:ccdrbg_init
fun:ccdrbg_generate
fun:cpu_datap
```

fun:ccdrbg_reseed
fun:cpu_shadowp
fun:PMAP_ZINFO_PALLOC
fun:pmap_pte
fun:pmap64_pml4
fun:pmap64_user_pml4
fun:PMAP_ZINFO_PFREE
fun:pmap_update_pte
fun:pmap64_pdpt
fun:pmap64_pde
fun:set_dirbase
fun:cpu_is_running
fun:pmap_pcid_validate_current
fun:PV_HASHED_ALLOC
fun:PV_HASHED_KERN_ALLOC
fun:pmap_pv_throttle
fun:pv_hash_add
fun:PV_HASHED_FREE_LIST
fun:PV_HASHED_KERN_FREE_LIST
fun:pv_hash_remove
fun:pmap_classify_pagetable_corruption
fun:DBLMAP_CHECK
fun:LDTALIAS_CHECK
fun:timer_queue_cpu
fun:enqueue_head
fun:bpf_tap_mbuf
fun:icmp_input
fun:flow_divert_kctl_send
fun:au_to_attr
fun:inflate_fast
fun:ivac_alloc
fun:ipc_create_mach_voucher
fun:ipc_voucher_attr_control_create_mach_voucher
fun:mach_voucher_attr_control_create_mach_voucher
fun:host_create_mach_voucher
fun:tlb_flush_global
fun:cpuid
fun:do_cpuid
fun:ipsec_start
fun:necp_assign_client_result
fun:pfr_route_kentry
fun:pfr_unroute_kentry
fun:in6_getsockaddr_s
fun:au_to_arg32
fun:au_to_arg
fun:au_to_attr32
fun:au_to_attr64
fun:au_to_exit
fun:au_to_groups
fun:au_to_newgroups
fun:au_to_in_addr_ex
fun:au_to_ipc
fun:au_to_ipc_perm
fun:au_to_file
fun:au_to_process32
fun:au_to_process64
fun:au_to_process
fun:au_to_process32_ex
fun:au_to_process64_ex
fun:au_to_return32
fun:au_to_return
fun:au_to_seq
fun:au_to_subject32
fun:au_to_subject64
fun:au_to_subject
fun:au_to_subject32_ex
fun:au_to_subject64_ex
fun:au_to_exec_strings
fun:au_to_header32_ex_tm
fun:au_to_header32_tm
fun:au_to_header64_tm
fun:au_to_trailer
fun:so_set_recv_anyif

IOKit Active Fuzzing

- Still based on **syzkaller**
- Code coverage driven
- API patterns based on reverse engineering

Use Syzkaller to fuzz IOKit

- Minimum Requirements:
  - IOKit API Patterns
  - IOKIT API Executor
  - Module Code Coverage

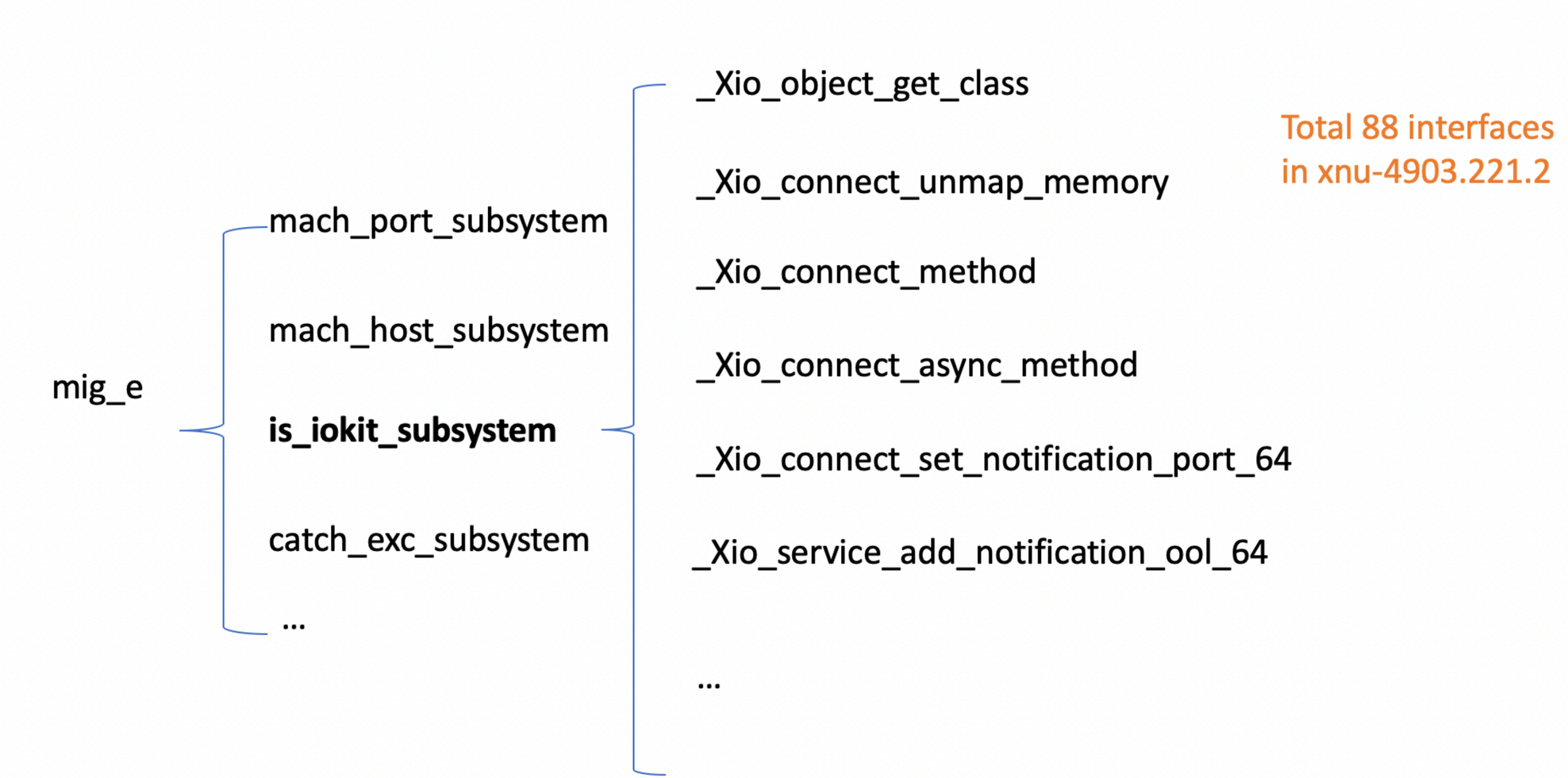
IOKit API Patterns

- For those open source IOKit modules, we extract the API patterns directly from source code

- For those **close source** IOKit modules, we use reverse engineering and other methods to help extract API patterns.

IOKit API Patterns

- IOKit Attack Surface



IOKit API Patterns

- Extract _Xio_connect_method patterns
  - Very famous interface, very big attack surface
  - Structural tables: sMethod, sDispatch
  - Unstructural: switch-case, if-else …

IOKit API Patterns

- Current Support
- Example: AppleFDEKeyStore

```
# base on macOS 10.13.6
# example for all 0 parameters:
# sli const[0], sli_len const[0], sti const[0], sti_size const[0], slo const[0], slo_len const[0], sto const[0], sto_size const[0]

resource conn_AppleFDEKeyStoreUserClient[int32]

syz_open_service$AppleFDEKeyStoreUserClient(service ptr[in, string["AppleFDEKeyStore"]], open_type const[0]) conn_AppleFDEKeyStoreUserClient

syz_call_method$userClientClose(io_conn conn_AppleFDEKeyStoreUserClient, sel const[1], sli const[0], sli_len const[0], sti const[0], sti_size const[0], slo con
syz_call_method$selfTest(io_conn conn_AppleFDEKeyStoreUserClient, sel const[2], sli const[0], sli_len const[0], sti const[0], sti_size const[0], slo const[0],
syz_call_method$setPassphraseWithUserID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[3], sli const[0], sli_len const[0], sti ptr[in,array[int64]], sti_si
syz_call_method$getPassphrase(io_conn conn_AppleFDEKeyStoreUserClient, sel const[4], sli const[0], sli_len const[0], sti const[0], sti_size const[0x10], slo co
syz_call_method$deletePassphrase(io_conn conn_AppleFDEKeyStoreUserClient, sel const[5], sli const[0], sli_len const[0], sti const[0], sti_size const[16], slo c
syz_call_method$getPassphraseNoCop(io_conn conn_AppleFDEKeyStoreUserClient, sel const[6], sli const[0], sli_len const[0], sti const[0], sti_size const[32], slo

syz_call_method$createKey(io_conn conn_AppleFDEKeyStoreUserClient, sel const[12], sli const[0], sli_len const[0], sti const[0], sti_size const[24], slo const[0
syz_call_method$setKeyWithUserID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[13], sli const[0], sli_len const[0], sti const[0], sti_size const[152], slo

syz_call_method$deleteKey(io_conn conn_AppleFDEKeyStoreUserClient, sel const[15], sli const[0], sli_len const[0], sti const[0], sti_size const[16], slo const[0
syz_call_method$unwrapVolumeKeyGetUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[16], sli const[0], sli_len const[0], sti const[0], sti_size const[272
syz_call_method$unwrapDiskKEKGetUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[17], sli const[0], sli_len const[0], sti const[0], sti_size const[300],
syz_call_method$wrapVolumeKey(io_conn conn_AppleFDEKeyStoreUserClient, sel const[18], sli const[0], sli_len const[0], sti const[0], sti_size const[32], slo con
syz_call_method$wrapDiskKEK(io_conn conn_AppleFDEKeyStoreUserClient, sel const[19], sli const[0], sli_len const[0], sti const[0], sti_size const[32], slo const
syz_call_method$unwrapDiskKEKToSMC(io_conn conn_AppleFDEKeyStoreUserClient, sel const[20], sli const[0], sli_len const[0], sti const[0], sti_size const[300], s
syz_call_method$setStashKey(io_conn conn_AppleFDEKeyStoreUserClient, sel const[21], sli const[0], sli_len const[0], sti const[0], sti_size const[20], slo const
syz_call_method$commitStash(io_conn conn_AppleFDEKeyStoreUserClient, sel const[22], sli const[0], sli_len const[0], sti const[0], sti_size const[0], slo const[
syz_call_method$getStashKey(io_conn conn_AppleFDEKeyStoreUserClient, sel const[23], sli const[0], sli_len const[0], sti const[0], sti_size const[4], slo const[
syz_call_method$setPBKDF(io_conn conn_AppleFDEKeyStoreUserClient, sel const[24], sli const[0], sli_len const[0], sti const[0], sti_size const[68], slo const[0]
syz_call_method$getPBKDF(io_conn conn_AppleFDEKeyStoreUserClient, sel const[25], sli const[0], sli_len const[0], sti const[0], sti_size const[68], slo const[0]
syz_call_method$setPassphraseGetUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[26], sli const[0], sli_len const[0], sti const[0], sti_size const[1028]
syz_call_method$createKeyGetUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[27], sli const[0], sli_len const[0], sti const[0], sti_size const[8], slo c
syz_call_method$setKeyGetUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[28], sli const[0], sli_len const[0], sti const[0], sti_size const[136], slo co
syz_call_method$userClientEnrypt(io_conn conn_AppleFDEKeyStoreUserClient, sel const[29], sli const[0], sli_len const[0], sti const[0], sti_size const[48], slo

syz_call_method$getStashKeyUUID(io_conn conn_AppleFDEKeyStoreUserClient, sel const[31], sli const[0], sli_len const[0], sti const[0], sti_size const[4], slo co



syz_close_service$AppleFDEKeyStore(io_conn conn_AppleFDEKeyStoreUserClient)
```

IOKit API Patterns

- Review if our R.E. is enough
  - With help of passive corpus
  - With help of fuzzing visualization

IOKit Executor

- Syzkaller has its own syscall executor which doesn’t support IOKit
  - It uses `syscall` API with syscall ID to execute the fuzzing call
  - It supports calling none-syscall functions which we can use to add supporting IOKit

```
long execute_syscall(call_t* c, long a0, long a1, long a2, long a3, long a4, long a5, long a6, long a7, long a8, l
{
    int ret = 0;
    int leak_check_count = 0;

_try_execute:
    if (c->call)
        ret = c->call(a0, a1, a2, a3, a4, a5, a6, a7, a8, a9);
    else
    {
        if (c->sys_nr > MAX_SYS_CALL_ID) {
            return -1;
        }
        ret = syscall(c->sys_nr, a0, a1, a2, a3, a4, a5, a6, a7, a8, a9);
    }
```

support IOKit call here

1. Call none syscall function during fuzzing

```
{Name: "syz_call_method$unmap_user_memory", CallName: "syz_call_method", Args: []Type{
    &ResourceType{TypeCommon: TypeCommon{TypeName: "conn_IGAccelGLContext", FldName: "io_conn", TypeSize: 4
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sel", TypeS
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sli", TypeS
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sli_len", T
    &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "sti", TypeSize: 8}, Type: &IntType{IntTypeC
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sti_size",
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "slo", TypeS
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "slo_len", T
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sto", TypeS
    &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "sto_size",
```

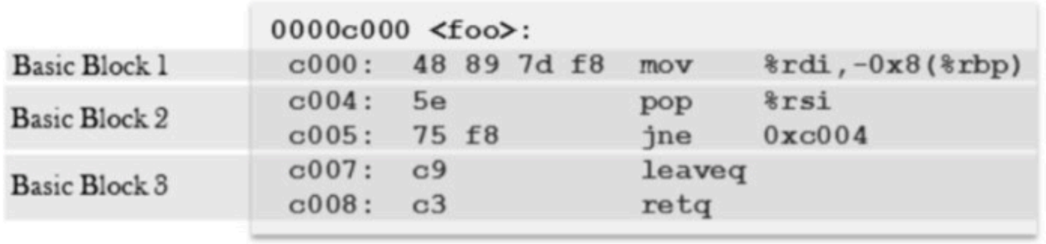
2. IOKit call name

```
void syz_call_method(io_connect_t conn, uint32_t sel,
                     uint64_t *sli, uint32_t sli_len,
                     char *sti, size_t sti_size,
                     uint64_t *slo, uint32_t slo_len,
                     char *sto, size_t sto_size)
{
    uint64_t input_scalar[8];
    char input_structure[4096];

    uint64_t output_scalar[8];
    uint32_t output_scalar_cnt = slo_len;

    char output_structure[4096];
    size_t output_structure_size = sto_size;

    if (sli == 0)
        sli = input_scalar;
    if (sti == 0)
        sti = input_structure;
    if (slo == 0)
        slo = output_scalar;
    if (sto == 0)
        sto = output_structure;

    IOConnectCallMethod(
                        conn, sel,
                        sli, sli_len,
                        sti, sti_size,
                        slo, &output_scalar_cnt,
                        sto, &output_structure_size);
}
```

3. syz_call_method

IOKit Code Coverage

- Get code coverage from close source IOKit modules
  - Possible methods: DBI, SBI, emulator, Intel PT
  - We use SBI



(a) An unmodified application function.

```
8007:  c9              leaveq
8008:  c3              retq

0000c000 <foo>:
 c000:  e9 de ad be ef  jmpq   0x8000
 c005:  90 90 90 90     [empty space]
```

(b) The application function after it has been relocated and the old function entry has been linked to it.

```
00008000 <_rel_foo>:
 8000:  48 89 7d f8        mov    %rdi,-0x8(%rbp)
 8004:  5e                 pop    %rsi
 8005:  0f 85 f8 ff ff ff  jne    0x8004
 800b:  c9                 leaveq
 800c:  c3                 retq

0000c000 <foo>:
 c000:  e9 de ad be ef  jmpq   0x8000
 c005:  90 90 90 90     [empty space]
```

(c) The application function after the branches have been converted to use 32-bit offsets.

```
00008000 <_rel_foo>:
 8000:  90 90 90 90 90     [empty space]
 8005:  48 89 7d f8        mov    %rdi,-0x8(%rbp)
 8009:  90 90 90 90 90     [empty space]
 800d:  5e                 pop    %rsi
 800e:  0f 85 f8 ff ff ff  jne    0x8009
 8014:  90 90 90 90 90     [empty space]
 8019:  c9                 leaveq
 801a:  c3                 retq

0000c000 <foo>:
 c000:  e9 de ad be ef  jmpq   0x8000
 c005:  90 90 90 90     [empty space]
```

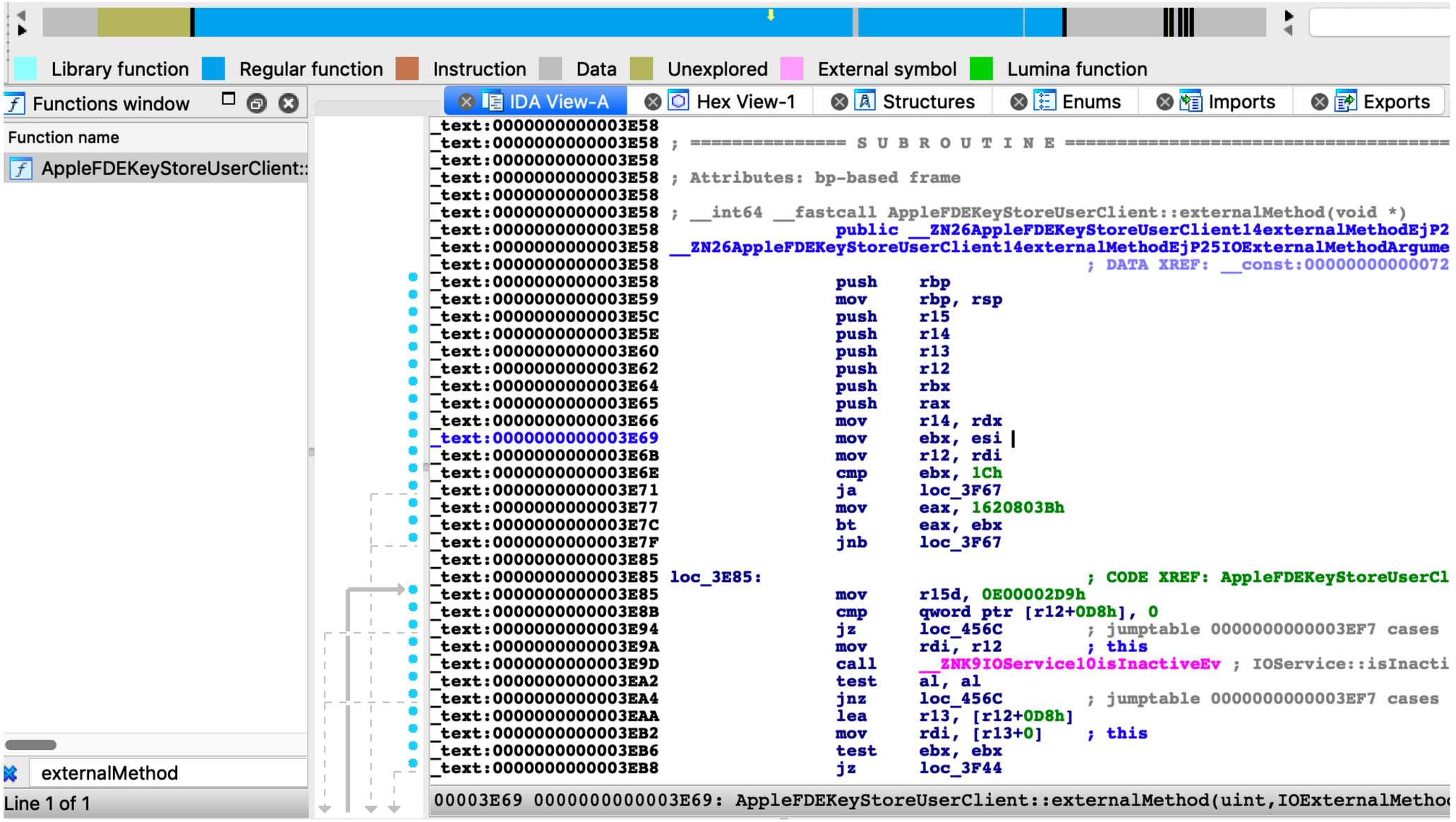
(d) The application function after it has been padded with 5 bytes at each instrumentation point.

```
00008000 <_rel_foo>:
 8000:  e9 fa de db ad     jmpq   0x9310
 8005:  48 89 7d f8        mov    %rdi,-0x8(%rbp)
 8009:  90 90 90 90 90     [empty space]
 800d:  5e                 pop    %rsi
 800e:  0f 85 f8 ff ff ff  jne    0x8009
 8014:  90 90 90 90 90     [empty space]
 8019:  c9                 leaveq
 801a:  c3                 retq
```

https://github.com/mlaurenzano/PEBIL

I got the SBI idea from project PEBIL.

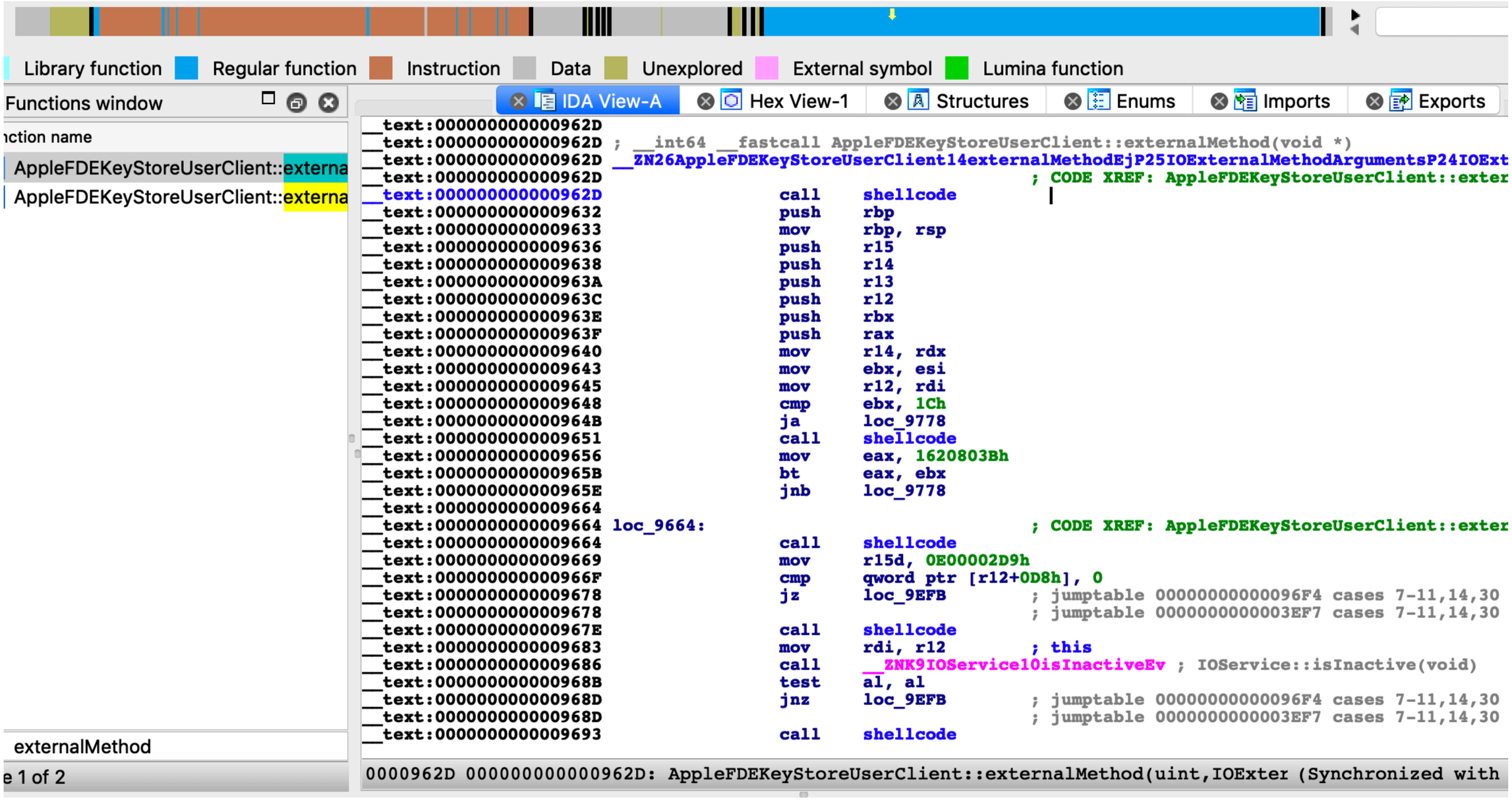
An example before my SBI



Before SBI(10.14.3 AppleFDEKeyStore)

An example after my SBI

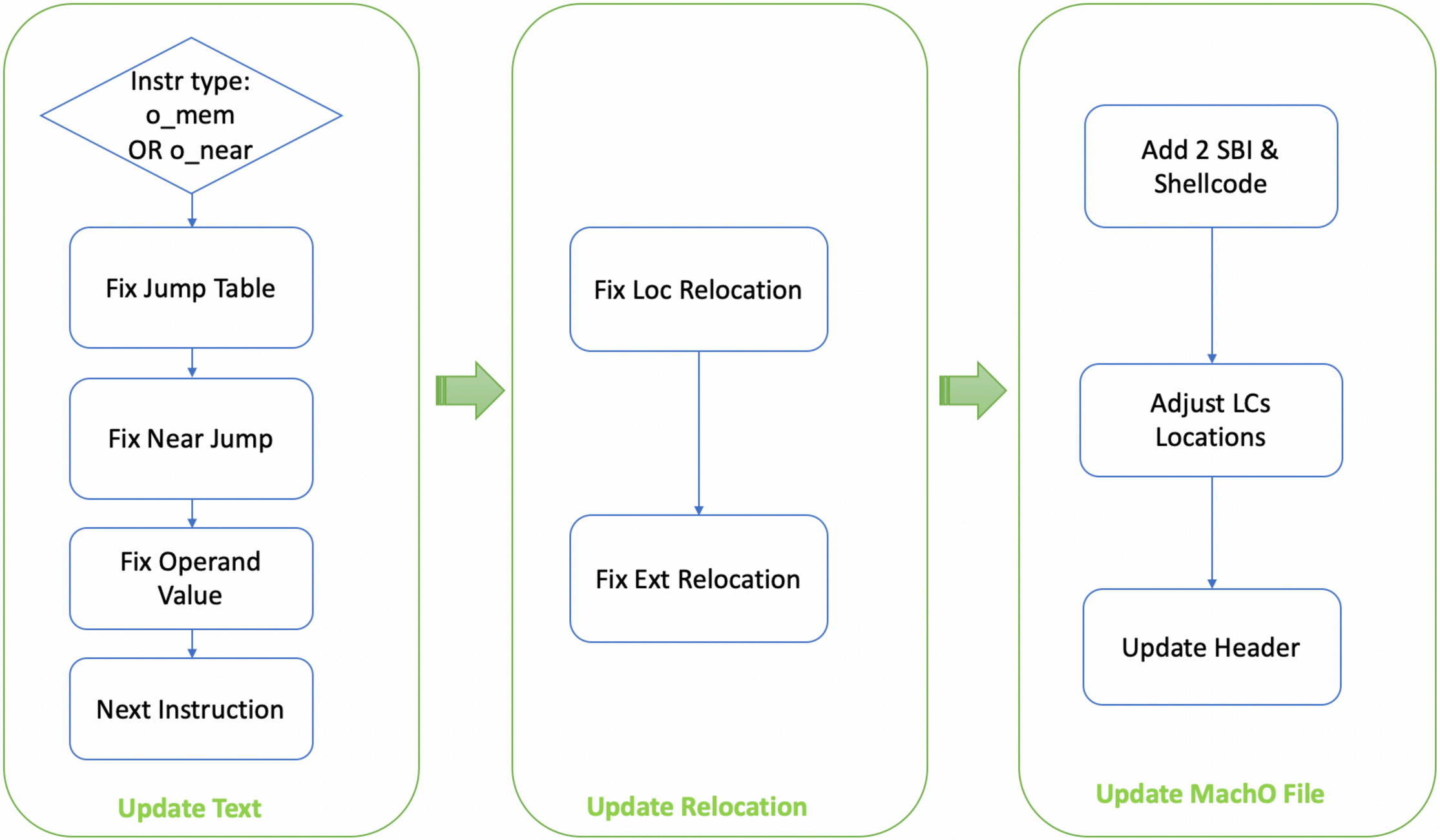
After SBI(10.14.3 AppleFDEKeyStore)



After SBI(10.14.3 AppleFDEKeyStore)

- The differences between before-SBI and after-SBI
  - 1 externalMethod function to 2 in IDA view
  - 1 text section to 2 text sections in IDA view
  - The first 5 bytes of original externalMethod function was patched to point to new function
  - New externalMethod function in new text section has each shellcode call at the beginning of each block

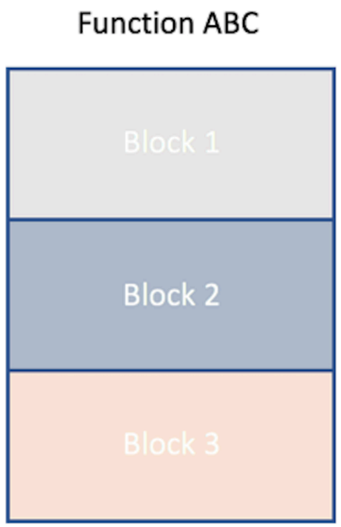
MachO SBI WorkFlow

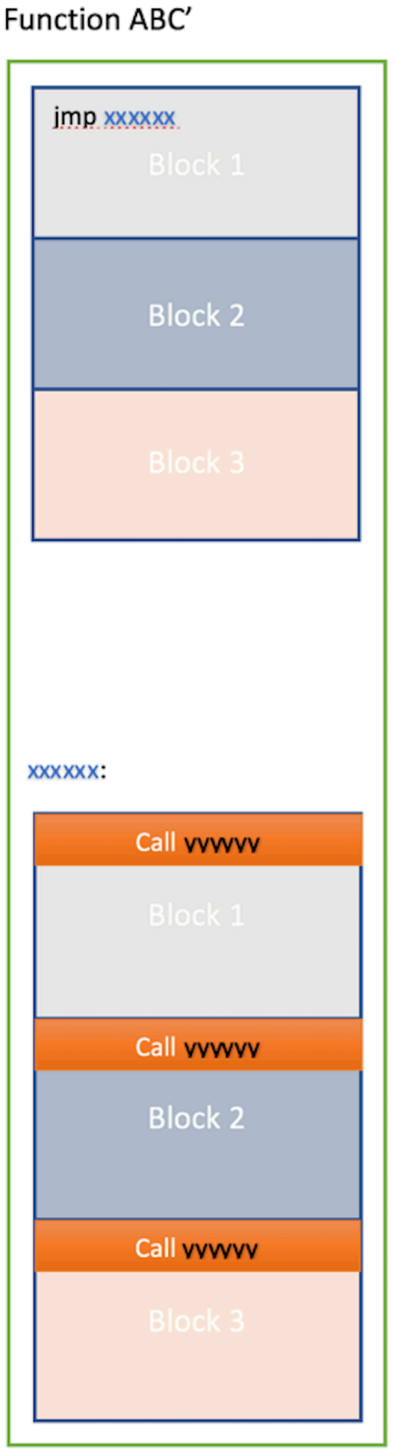


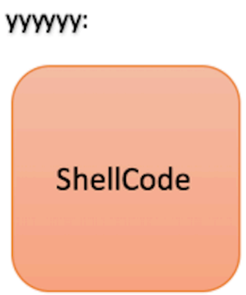
Update Text

Rebuild Text Segment:

1. Handle all functions, blocks, instructions
2. For each function ABC, we will rebuild it as ABC’
3. Anyone calls ABC’, it jumps to xxxxxx
4. xxxxxx calls shellcode yyyyyy at the start of each block
5. shellcode yyyyyy records current RIP into buffer
6. Return buffer as code coverage







Some facts here:

1. Location ABC is in original TEXT section
2. Location xxxxxx is in new added SBI TEXT section
3. Location yyyyyy is in new added SBI TEXT section

Update Text: o_mem, o_near

op.type	Description	Data field
o_void	No Operand	None
o_reg	General Register (al,ax,es,ds...)	Reg
o_mem	Direct Memory Reference (DATA)	Addr
o_phrase	Memory Ref [Base Reg + Index Reg]	Phrase
o_displ	Memory Reg [Base Reg + Index Reg + Displacement]	Phrase+Addr
o_imm	Immediate Value	Value
o_far	Immediate Far Address (CODE)	Addr
o_near	Immediate Near Address (CODE)	Addr

```
4C 8D BD 84 F4+lea     r15, [rbp+v

                loc_3818:               ; unsigned int
B9 00 04 00 00  mov     ecx, 400h
45 31 C9        xor     r9d, r9d        ; bool
4C 89 EF        mov     rdi, r13        ; this
4C 89 F6        mov     rsi, r14        ; unsigned __int8 *
4C 89 E2        mov     rdx, r12        ; void *
4D 89 F8        mov     r8, r15         ; unsigned int *
E8 CB E1 FF FF  call    __ZN16AppleFDEKeyStore13getPassphraseEPhPvjPjb ; AppleFDEKe
85 C0           test    eax, eax
0F 84 37 02 00+jz       loc_3A70

48 FF C3        inc     rbx
49 83 C6 10     add     r14, 10h
48 83 FB 03     cmp     rbx, 3
76 D2           jbe     short loc_3818
```

Before SBI(10.14.3 AppleFDEKeyStore)

1. Fix Near Jump
   76 —> 0F 86
2. Fix Operand
   D2 —> C4 FF FF FF

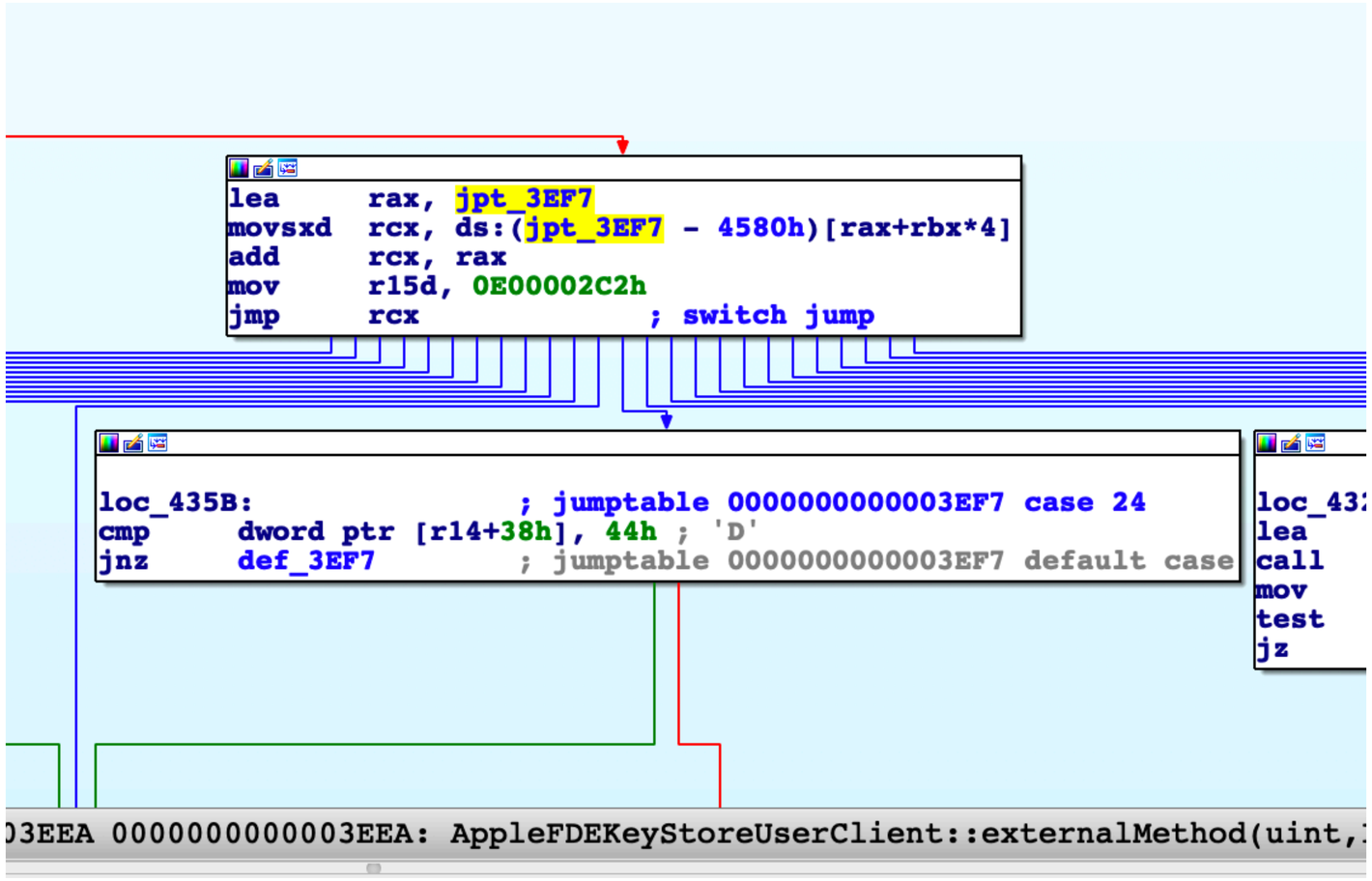
```
                loc_849F:
E8 9C 5A 00 00  call    shellcode
B9 00 04 00 00  mov     ecx, 400h       ; unsigned int
45 31 C9        xor     r9d, r9d        ; bool
4C 89 EF        mov     rdi, r13        ; this
4C 89 F6        mov     rsi, r14        ; unsigned __int8 *
4C 89 E2        mov     rdx, r12        ; void *
4D 89 F8        mov     r8, r15         ; unsigned int *
E8 33 59 00 00  call    __ZN16AppleFDEKeyStore13getPassphraseEPhPvjPjb_0 ;
85 C0           test    eax, eax
0F 84 9A 02 00+jz       loc_875F

E8 76 5A 00 00  call    shellcode
48 FF C3        inc     rbx
49 83 C6 10     add     r14, 10h
48 83 FB 03     cmp     rbx, 3
0F 86 C4 FF FF+jbe      loc_849F
```

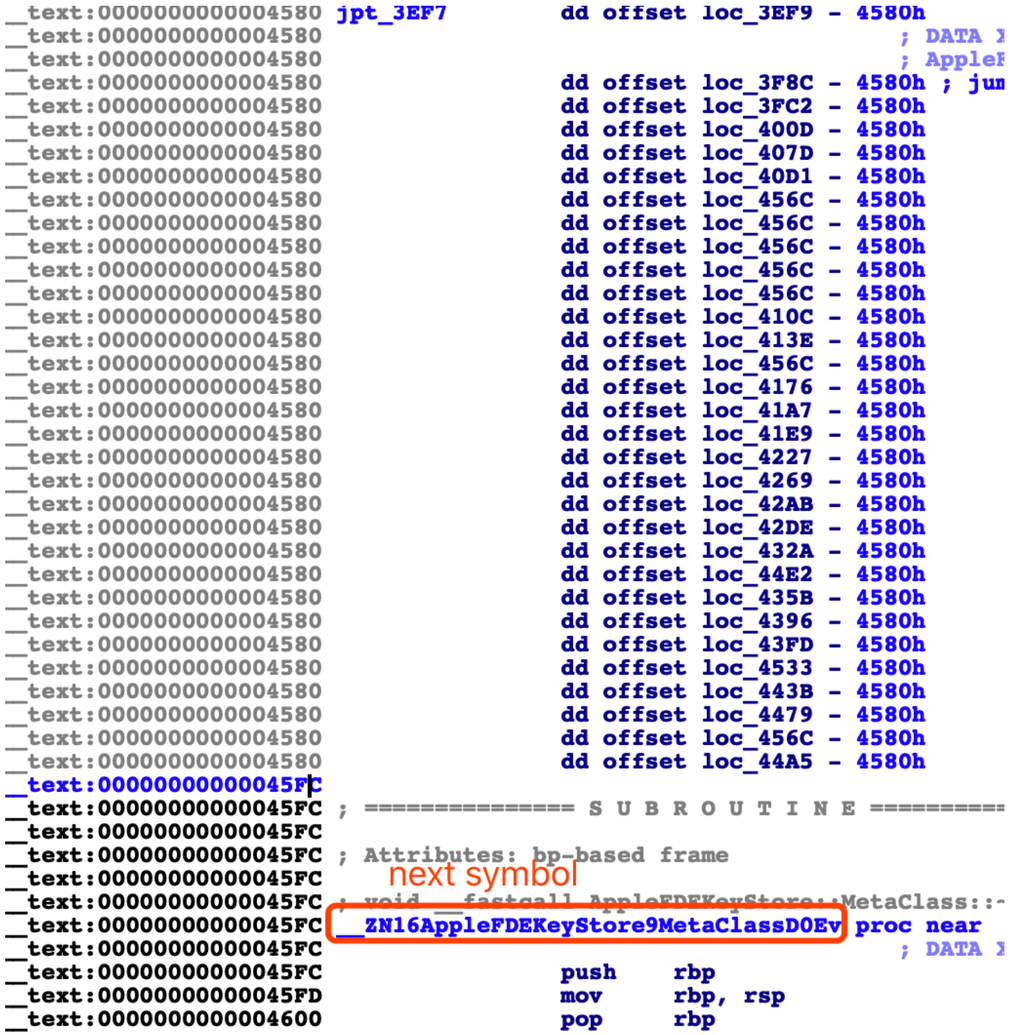
After SBI(10.14.3 AppleFDEKeyStore)

Full Near -> Long Jump List

Mnem	Long Jump Opcode
jo	\x0f\x80
jno	\x0f\x81
jb\jc\jnae	\x0f\x82
jae\jnb\jnc	\x0f\x83
jz\je	\x0f\x84
jnz\jne	\x0f\x85
jbe\jna	\x0f\x86
ja\jnbe	\x0f\x87
js	\x0f\x88
jns	\x0f\x89
jp\jpe	\x0f\x8a
jnp\jpo	\x0f\x8b
jl\jnge	\x0f\x8c
jge\jnl	\x0f\x8d
jle\jng	\x0f\x8e
jg\jnle	\x0f\x8f
jmp	\xe9

Jump Table



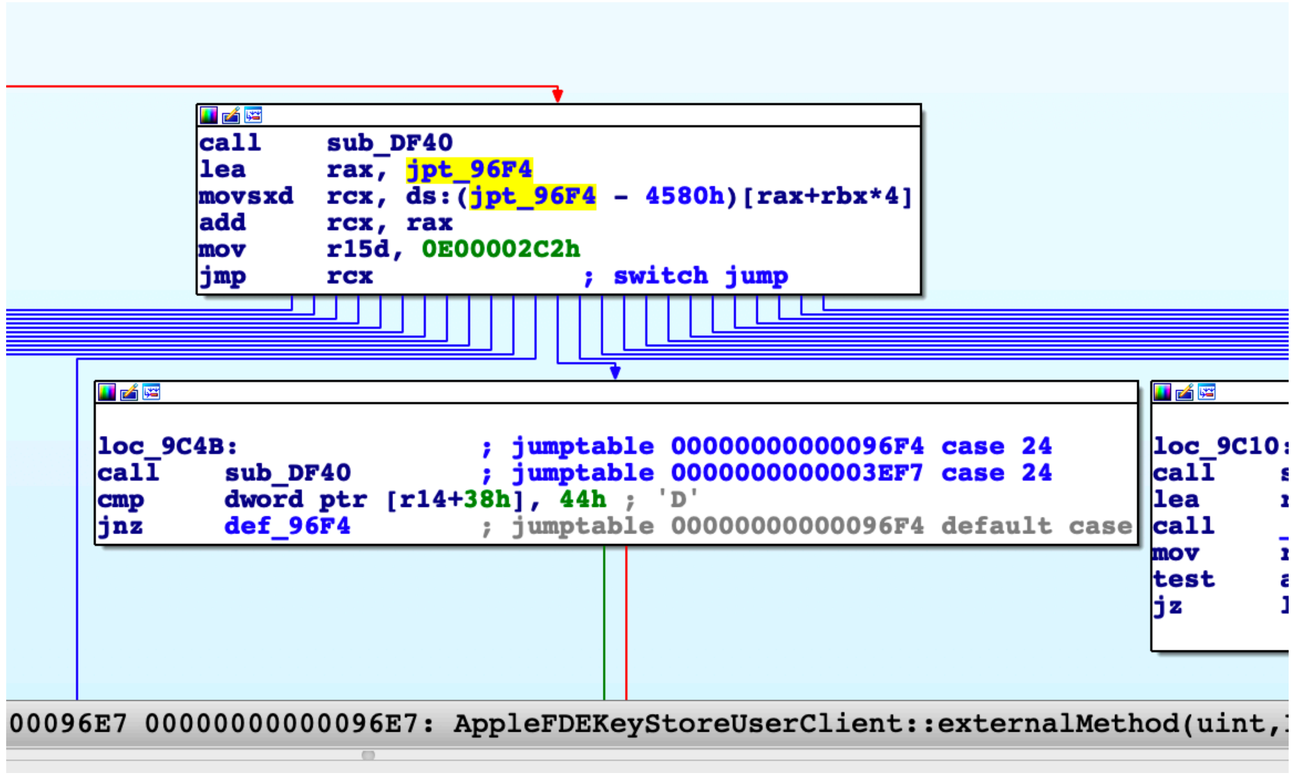


10.14.3 AppleFDEKeyStore

Jump table is a range of data in text code:

1) we must recognize it as data instead of code
2) We must fix it during SBI or the new function will jump back to original function which may panic

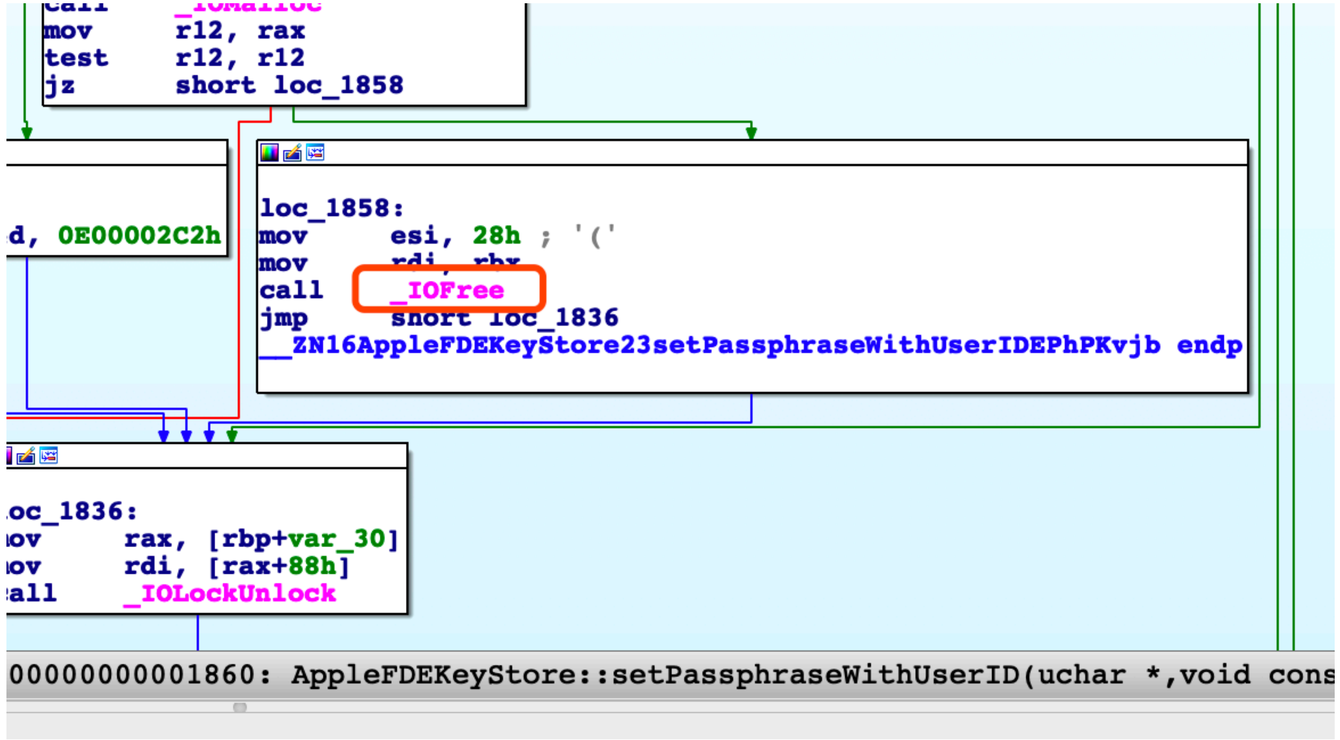
- How to recognize it?
  - Operand value of the instruction(type: o_mem or o_near) points to somewhere inside text section
  - But doesn’t point to any existing function block
  - All the jump table item must points to existing function block
  - Jump table size: find next symbol after the table start, the table end lies just before the found symbol
- How to fix it during SBI?
  - Re-use the jump table instead of copying the table into new SBI text
  - Fix every table entry making it points to new code branch in SBI text

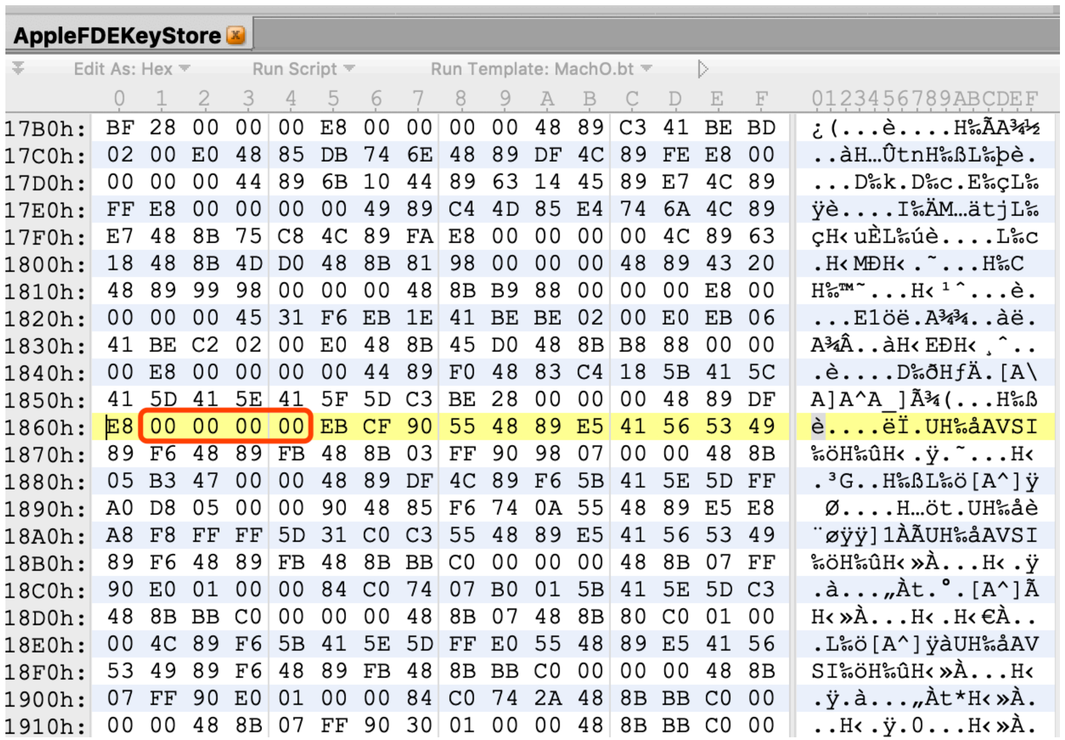


10.14.3 AppleFDEKeyStore after SBI

1. The above code is in new added SBI text, not in the original text section (address around 0x9c4b, not 0x435b)
2. The jump table is re-used at location 0x4580
3. But all table entries are updated to point to branches in new added SBI text

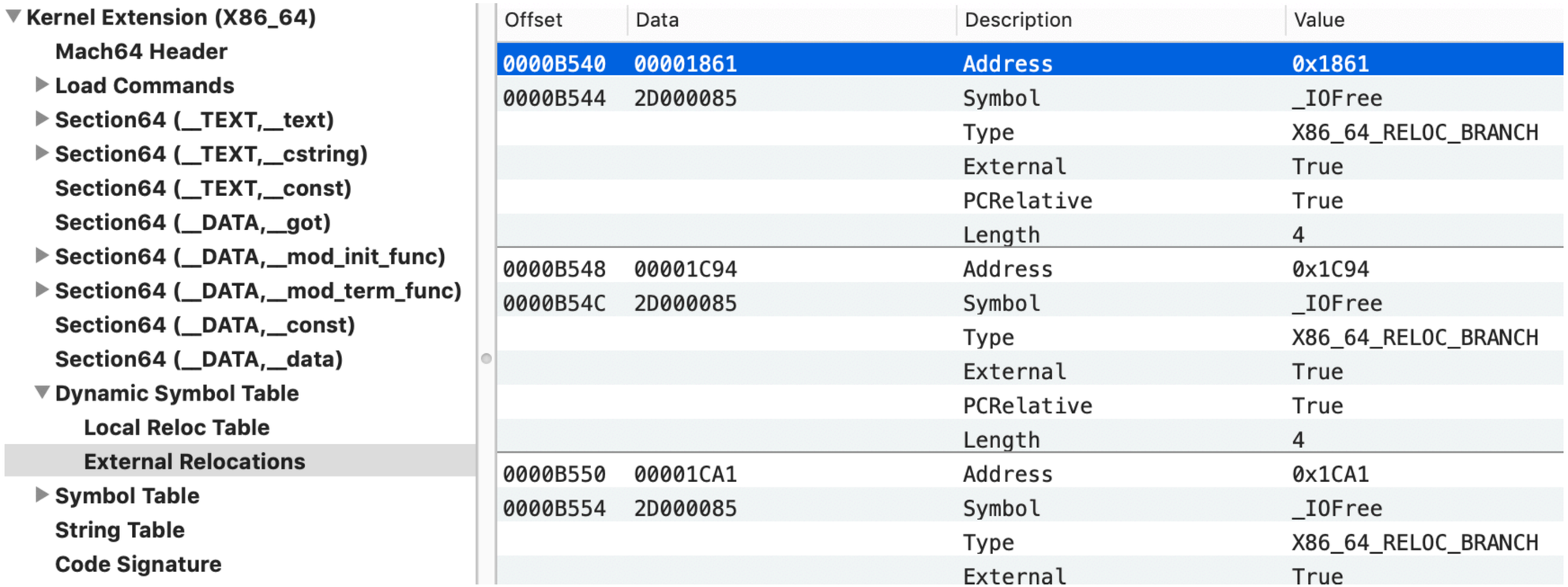
Relocation: What is relocation?





10.14.3 AppleFDEKeyStore

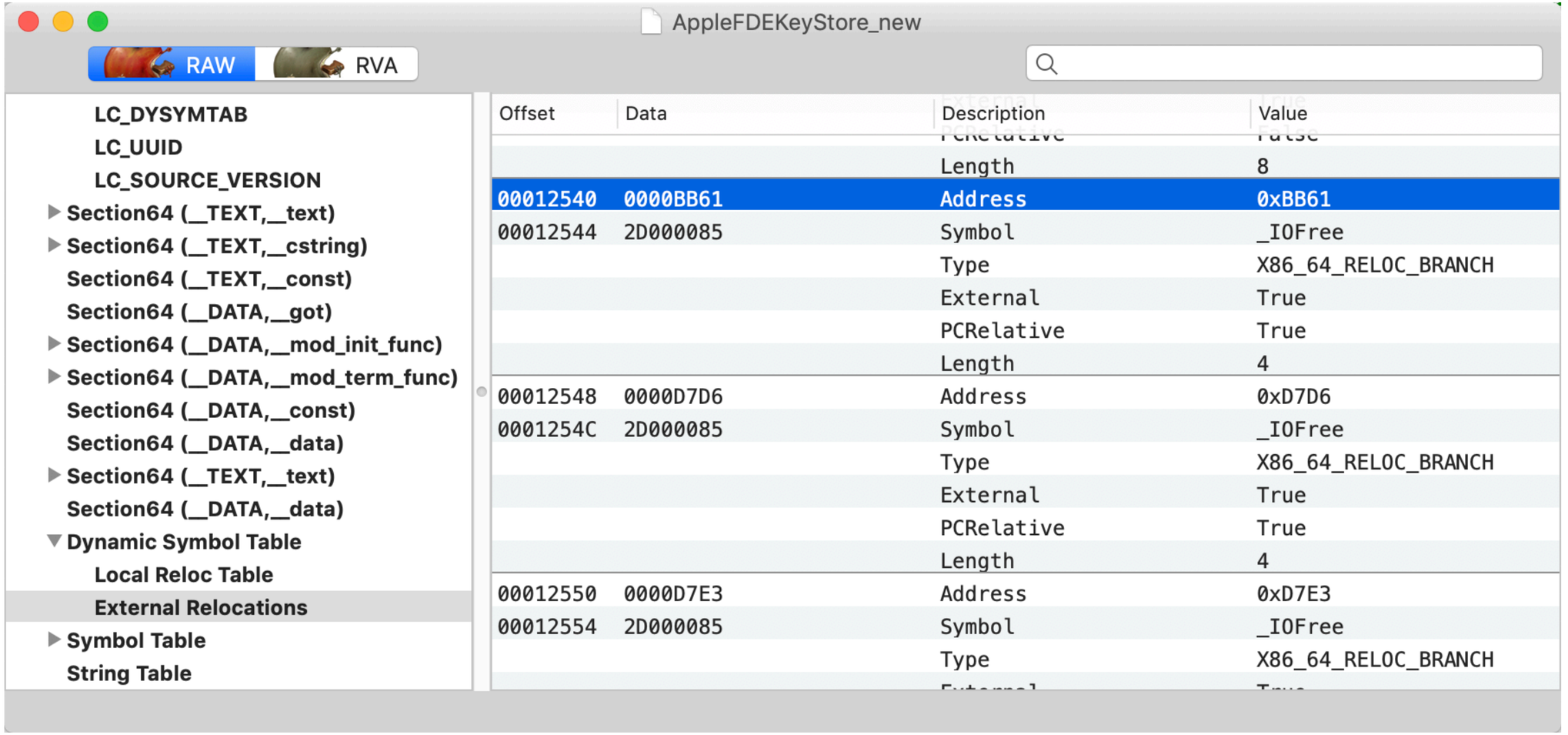
1. At file offset 0x1860, it calls _IOFree
2. But the address of _IOFree is 0 in raw file context
3. Actually _IOFree is relocated during module loading



Before SBI (10.14.3 AppleFDEKeyStore)

1. 0x1861 is described in Dynamic Symbol Table in MachO file
2. We need update the Dynamic Symbol Table to let the entry points to SBI text instead of original text section
3. If not update, the same function(setPassphraseWithUserID) in SBI text will ‘call 0x0’ instead of ‘call _IOFree’

Fix Relocation



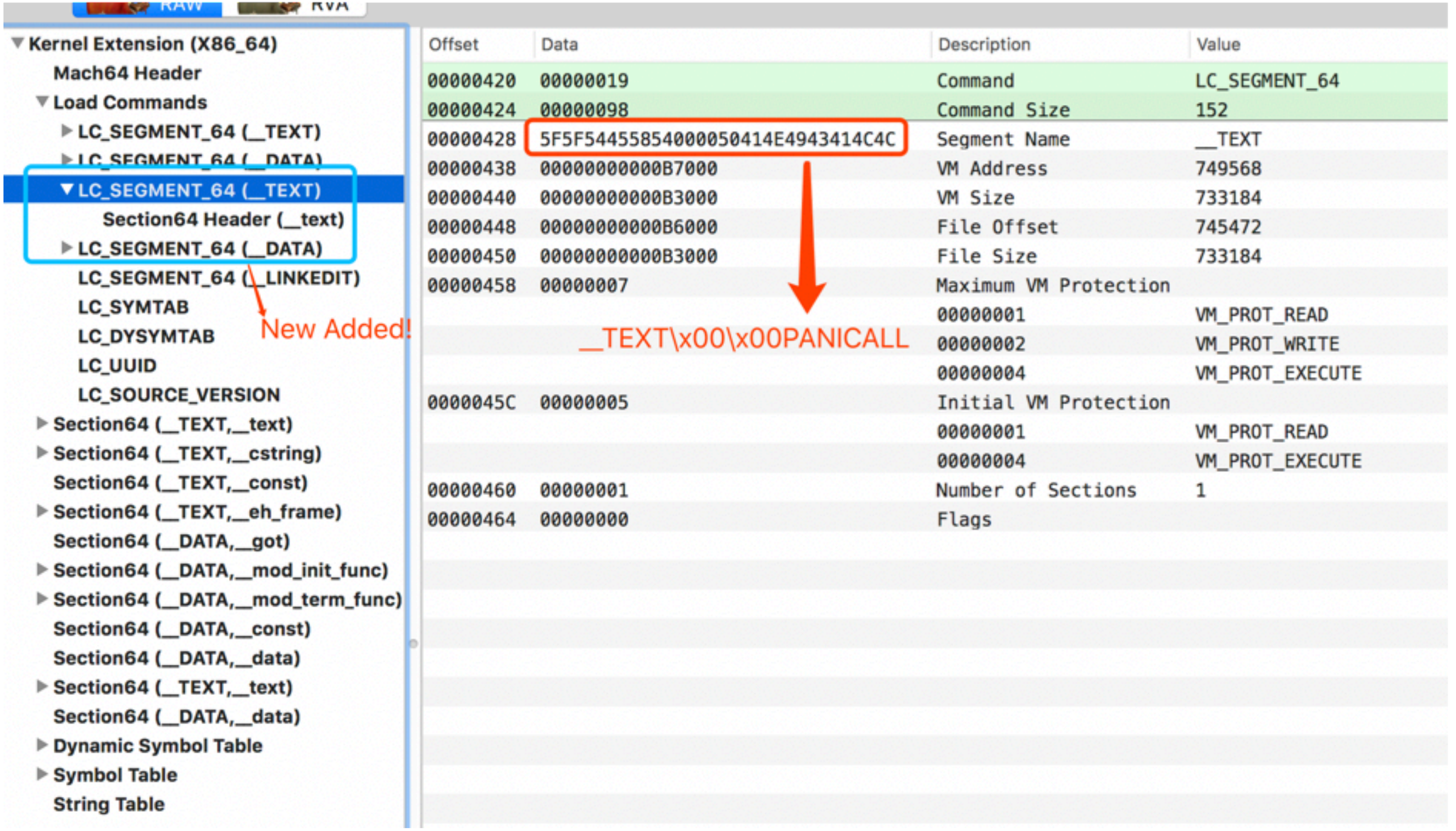
After SBI (10.14.3 AppleFDEKeyStore)

1. 0xb540 changed to 0x12540 since new SBI section added
2. 0x1861 is updated to 0xbb61

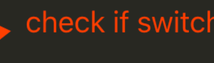
Update MachO file structure

- Add SBI TEXT section and SBI DATA section
- Add Shellcode
- Update MachO header, e.g. load commands(LC_SYMTAB, LC_DYSYMTAB, LC_DYLD_INFO, LC_DYLD_INFO_ONLY, LC_FUNCTION_STARTS, LC_DATA_IN_CODE)

Add SBI TEXT & DATA Sections



```
start:
pushfq
push rdi
push rax
call xxx
xxx:
pop rax
add rax, 0x1000
mov edi, [rax]
test edi, edi
jz abc
mov rdi, [rsp+0x18]
push rsi
push rdx
push rcx
push r8
push r9
push rbx
push r10
push r11
push r12
push r13
push r14
push r15
add rax,4
mov rsi, [rax]
add rax,8
mov rax, [rax]
call rax
pop r15
pop r14
pop r13
pop r12
pop r11
pop r10
pop rbx
pop r9
pop r8
pop rcx
pop rdx
pop rsi
abc:
pop rax
pop rdi
popfq
retn

section .data
X:  dd 0x12345678
Y:  dq 0x1234567887654321
Z:  dq 0x4141414141414141
```



Add Shellcode:

1) just a stub to call real recording function

2) Save & Restore all registers

3) You can implement your recording function in a standard kernel driver

```
def _add_trace_codes(self, body_size):
    sbi_text_offset = self.file_info['sbi_text_offset']
    fp_out = self.file_info['fp_out']

    trace_codes = b'\x9C\x57\x50\xE8\x00\x00\x00\x00\x58\x48\x05\x00\x10\x00\x00\x8B' \
                  b'\x38\x85\xFF\x74\x3D\x48\x8B\x7C\x24\x18\x56\x52\x51\x41\x50\x41' \
                  b'\x51\x53\x41\x52\x41\x53\x41\x54\x41\x55\x41\x56\x41\x57\x48\x83' \
                  b'\xC0\x04\x48\x8B\x30\x48\x83\xC0\x08\x48\x8B\x00\xFF\xD0\x41\x5F' \
                  b'\x41\x5E\x41\x5D\x41\x5C\x41\x5B\x41\x5A\x5B\x41\x59\x41\x58\x59' \
                  b'\x5A\x5E\x58\x5F\x9D\xC3'

    fp_out.seek(sbi_text_offset+body_size, 0)
    fp_out.write(trace_codes)

    new_body_size = len(trace_codes)

    if new_body_size % 0x10 != 0:
        new_body_size += 0x10 - new_body_size % 0x10

    return new_body_size
```

SBI Summary

- Most important work: Address and Offset Fixup
  - Fix o_mem and o_near instructions
  - Fix relocations
  - Fix file structure
- Add shellcode to record the RIP value
  - Use asm
  - As simple as you can

- Add 2 Segments:
  - __TEXT\x00\x00PANICALL
  - __DATA\x00\x00PANICALL

- Where to add the 2 segments?
  - Before __LINKEDIT segment
  - Can we put them at the end of load commands?

- My Answer:
  - If SBI an usermode MachO file, Yes.
  - If SBI a kernelmode MachO file, NO! Or system won’t boot.

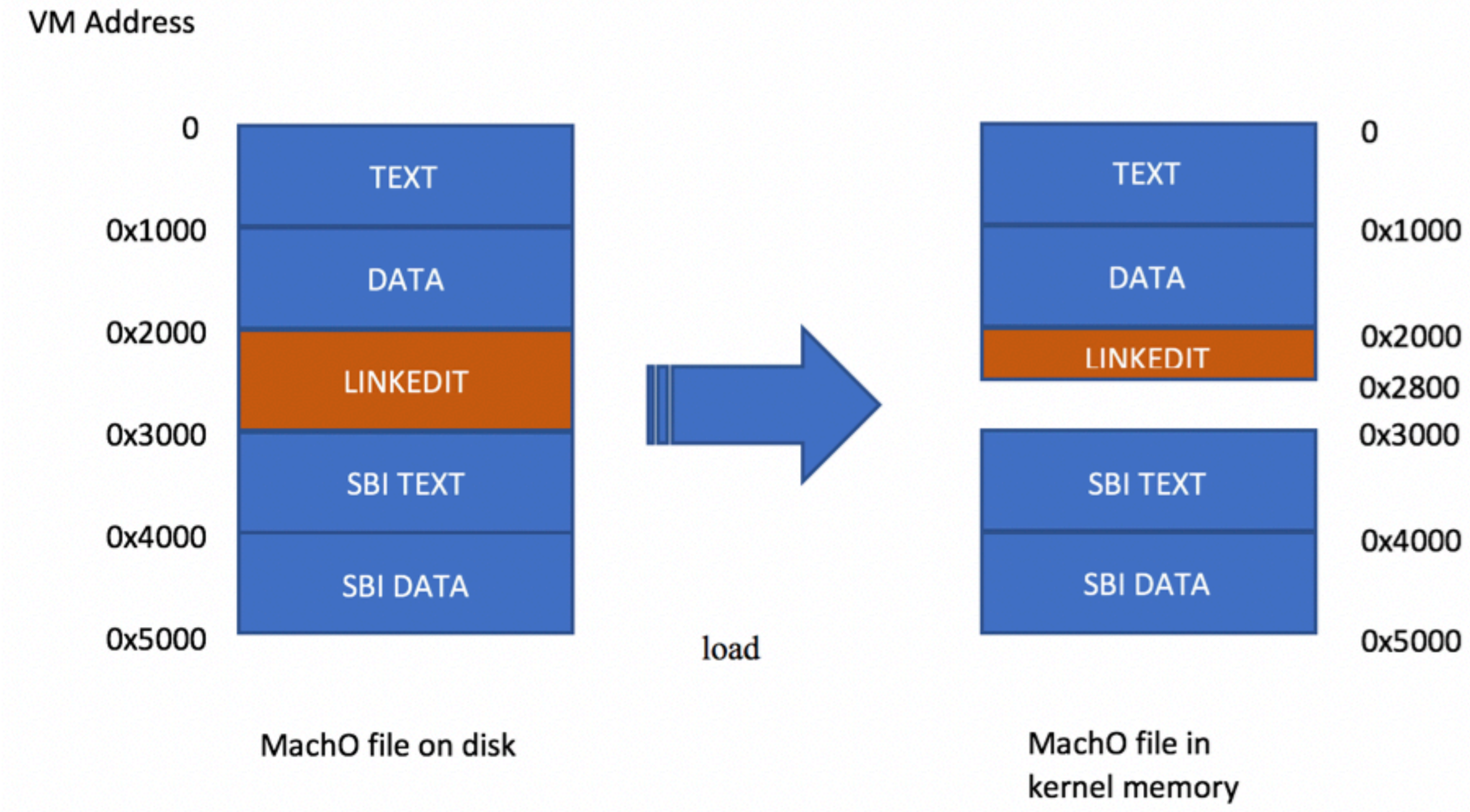
```
void
kxld_seg_populate_linkedit(KXLDSeg *seg, const KXLDSymtab *symtab,
boolean_t is_32_bit
#if KXLD_PIC_KEXTS
    , const KXLDArray *locrelocs
    , const KXLDArray *extrelocs
    , boolean_t target_supports_slideable_kexts
#endif  /* KXLD_PIC_KEXTS */
    , uint32_t splitinfolc_size
   )
{
    u_long size = 0;
    size += kxld_symtab_get_macho_data_size(symtab, is_32_bit);
#if KXLD_PIC_KEXTS
    if (target_supports_slideable_kexts) {
        size += kxld_reloc_get_macho_data_size(locrelocs, extrelocs);
    }
#endif  /* KXLD_PIC_KEXTS */
    // 0 unless this is a split kext
    size += splitinfolc_size;
    seg->vmsize = kxld_round_page_cross_safe(size); seg->vmsize is updated here
}
```

During initialization, linkedit load command can be updated to a smaller size.

This will make something unexpected if you append new segments after LINKEDIT.

Kernel will calculate the needed virtual memory before module loading, for example, the MachO in picture needs 0x4800 bytes memory since there is a 0x800 bytes optimization inside LINKEDIT.

if we append the 2 new segments at the end, the SBI DATA segment will be mapped exceeding 0x4800. Then panic!



Limitation

- Not support FAT macho
- Not support 32 bit macho
- Macho header must have free space for 2 more load commands(SBI TEXT and SBI DATA)
- Not perfect solution for jump table identification in text section
- Hardware Module Protection: Disable

SBI Partial Status in Fusion VM

Module	macOS 10.14	macOS 10.13.6	Module	macOS 10.14	macOS 10.13.6
IOThunderboltFamily	OK	OK	IOUSBMassStorageDriver	Unknown	OK
IOSurface	OK	OK	IOATAFamily	Unknown	OK
IOUSBFamily	OK	OK	IOSMBusFamily	Unknown	OK
IOGraphicsFamily	OK	OK	IOUSBHostFamily	Unknown	OK
AppleHDA	OK	OK	IOSCSIArchitectureModelFamily	Unknown	OK
DspFuncLib	OK	OK	IOACPIFamily	Unknown	OK
IOBluetoothFamily	Failure	OK	IOSlowAdaptiveClockingFamily	Unknown	OK
IOReportFamily	OK	OK	IOSerialFamily	Unknown	OK
IOStorageFamily	OK	OK	IOCDStorageFamily	Unknown	OK
IOAudioFamily	OK	OK	IONDRVSupport	Unknown	OK
IOAVBFamily	OK	OK	IOAHCIFamily	Unknown	Failure
IOTimeSyncFamily	OK	OK	IOBDStorageFamily	Unknown	Failure
IODVDStorageFamily	OK	OK			
VMwareGfx	OK	OK			
IOHIDFamily	OK	OK			
IONetworkingFamily	OK	OK			

If failure on your physical machine, goto recover mode

If you get a failure boot after using the tool, try the following steps to recover your system.

Take `AppleIntelKBLGraphics.kext` as an example:

1. recover the binary

```
mv AppleIntelKBLGraphics AppleIntelKBLGraphics_new
mv AppleIntelKBLGraphics_org AppleIntelKBLGraphics
```

2. add owner

   `chown -R root:wheel AppleIntelKBLGraphics.kext`

3. build kextcache

   `kextcache -arch x86_64 -kernel /Volumes/Macintosh\ HD/System/Library/Kernels/kernel -prelinked-kernel /Volumes/Macintosh\ HD/System/Library/PrelinkedKernels/prelinkedkernel -volume-root /Volumes/Macintosh\ HD/ /Volumes/Macintosh]\ HD/System/Library/Extensions/ /Volumes/Macintosh\ HD/Library/Extensions/`

4. reboot

IOKit Passive Fuzzing

- Passive Fuzzing commonly uses hooks to mutate parameters during normal function call.
- We just use hooks to dump parameters to mysql & mongodb
- Syzkaller uses concept ‘program’ so that we need convert the collected parameters to ‘program’

id	service_name	type	class_name	xnu_version_start	submit_time
76	IOBluetoothDevice	0	IOBluetoothDeviceUserClient	1802	2018-12-27 17:40:12
77	IOBluetoothL2CAPChan…	0	IOBluetoothL2CAPChannelU…	1802	2018-12-27 17:40:12
78	AppleHSBluetoothDevice	4737348	IOHIDLibUserClient	1802	2018-12-27 17:40:13
79	AppleHSBluetoothHIDDr…	4737348	IOHIDLibUserClient	1802	2018-12-27 17:40:13
80	AppleHIDKeyboardEven…	1212761156	IOHIDEventServiceUserClient	1802	2018-12-27 17:40:14
81	AppleMultitouchDevice	0	AppleMultitouchDeviceUserCl…	1802	2018-12-27 17:41:06
82	AppleActuatorDevice	0	AppleActuatorDeviceUserClient	1802	2018-12-27 17:41:06
83	AMDRadeonX4000_AM…	3	AMDRadeonX4000_AMDSIVi…	1802	2018-12-28 11:19:16
84	IOUserEthernetResource	0	IOUserEthernetResourceUser…	1802	2019-01-01 18:46:26
85	AppleMobileFileIntegrity	0	AppleMobileFileIntegrityUser…	1802	2019-01-16 14:50:45
86	VMwareFramebuffer	1	IOFramebufferSharedUserCli…	1802	2019-01-16 14:56:17
87	AppleUSBInterface	0	IOUSBInterfaceUserClientV3	1802	2019-01-16 15:35:36
88	AppleUSBInterface	0		1802	2019-01-16 15:46:19
89	AppleIntelFramebuffer	1	IOFramebufferSharedUserCli…	1802	2019-01-16 15:46:45
90	AudioAUUCDriver	0	AudioAUUC	1802	2019-01-16 15:48:35
91	AppleUpstreamUserClie…	0	AppleUpstreamUserClient	1802	2019-01-16 15:48:35
92	IntelAccelerator	258	IGAccelVideoContextVEBox	1802	2019-01-16 15:58:17
93	IntelAccelerator	257	IGAccelVideoContextMedia	1802	2019-01-16 15:58:17
94	IntelAccelerator	256	IGAccelVideoContextMain	1802	2019-01-16 15:58:17
95	AppleHPMIECS	0	AppleHPMUserClient	1802	2019-01-17 09:24:20
96	AppleHSSPIHIDDriver	4737348	IOHIDLibUserClient	1802	2019-01-17 09:24:30
97	AirPort_BrcmNIC	1833193043	mDNSOffloadUserClient	1802	2019-01-17 10:09:44
98	AppleEmbeddedOSSup…	0	AppleEmbeddedOSSupportH…	1802	2019-01-17 10:12:02
99	IONVMeBlockStorageDe…	12	AppleNVMeSMARTUserClient	1802	2019-01-17 11:45:44
100	IOHIDUserDevice	4737348	IOHIDLibUserClient	1802	2019-01-21 14:55:36
101	AppleUSBDevice	0		1802	2019-01-22 18:02:47
102	com_epson_driver_EPS…	0	IOAudioEngineUserClient	1802	2019-01-23 10:12:00
103	ZoomAudioEngine	0	IOAudioEngineUserClient	1802	2019-01-23 10:12:00

IOKit Connect Open

id	class_name	selector	scalar_input_count	scalar_output_count	struct_input_size	struct_output_size	ool_input_size	ool_output_size	md5_no_content
447	IOUSBInterfaceUserClien…	27	1	0	0	177	0	0	e8f1125673b402fb74bd26d8104937d8
448	IOHIDLibUserClient	11	3	0	0	0	0	0	308ea821acdb98c4e2aa6acc629c69f5
449	SCSITaskUserClient	4	1	0	0	0	0	0	5cae538878a4b51002dfd3e5ff736d57
450	IOUSBDeviceUserClientV2	4	1	0	0	41	0	0	acb21b1ba5cfe35fe509c720a2ee9c1d
451	AudioAUUC	3	1	0	32	0	0	0	15ccd8738bb37427816ffa2fca42d872
452	AudioAUUC	3	1	0	40	0	0	0	a595a3857ea439dff81f4d6003544da2
453	AppleNVMeSMARTUser…	6	1	1	0	0	0	0	89671e54a8e1b6a9b92a0afa7601178d
454	AppleNVMeSMARTUser…	7	1	1	0	0	0	0	6c9778304bf253242a478b1d32cea46d
455	AudioAUUC	3	1	0	41	0	0	0	008efbe522616a8b908fb8c74e095796
456	IGAccelVideoContextMain	258	0	0	4	0	0	0	29199d1be6c9651630370e96b2facf7b
457	AudioAUUC	3	1	0	33	0	0	0	6687e2666681cdea3fb77f1e4744b84c
458	AudioAUUC	3	1	0	36	0	0	0	2f87b4b50b61e3a5c12877d3f4638cba
459	CCDataPipeUserClient	4	0	0	16	16	0	0	2c88bc94f4d054619d4682cccd757832
460	IOUSBInterfaceUserClien…	42	2	0	0	0	0	0	aab7a7eb2834d25e4bd034487154b58b
461	IOUSBInterfaceUserClien…	41	1	0	0	0	0	0	0a80125b74e339e1c702ad82c77a7d72
462	IGAccelDevice	11	0	0	187	187	0	0	b3dfb292dbc6e69c73798a10e5eafb31
463	IOSurfaceRootUserClient	0	0	0	472	2256	0	0	37e6911f979dfa74326f505fd4133aac
464	IGAccelDevice	11	0	0	0	187	0	0	6c9be01633112fde961c1a82ad490745
465	IOBluetoothL2CAPChann…	0	2	0	1	0	0	0	d00a8cdb086b73bba0fe07d91f1e4699
466	IOBluetoothL2CAPChann…	5	0	0	0	0	0	0	5d202f3c0de9950e71d5b4f0c5de4a9b
467	IOHIDLibUserClient	15	1	0	0	192	0	0	d745406dff52a71615aaa8b873e8615e
468	IOSurfaceRootUserClient	9	0	0	4040	4	0	0	591b221567902006b0b7ec107079c8ce
469	IOUSBInterfaceUserClien…	27	1	0	0	4	0	0	b170b1d843864e5d73c7ad51f9e8efe1
470	IOBluetoothL2CAPChann…	0	2	0	14	0	0	0	b760f7f44b99e10324e218b248396cdc
471	IOSurfaceRootUserClient	11	0	0	9	4	0	0	ffd3d5fa86d0babffd1f1e74fa21676b
472	IOSurfaceRootUserClient	0	0	0	272	2256	0	0	c01b6668b2e05e039e87104f7d60e2b9
473	IOSurfaceRootUserClient	0	0	0	608	2256	0	0	4e7b68409db3a662b6f081d3642b61df
474	IOSurfaceRootUserClient	0	0	0	464	2256	0	0	98ad1dbd1af79283f0d1dd50124c70ee
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

IOKit Connect Call

```
{ "_id" : ObjectId("5bd963e550ffb410cef3fc20"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOSurfaceRootUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5"
: "0ef85b86dd5ca879c0d1a2e2e05cefa8", "struct_output_size" : 1736, "selector" : 2, "api" : "iokit_connect", "scalar_input" : BinData(0,""),
"struct_input_size" : 12, "struct_input" : BinData(0,"BAAAAAEAAADoz7b/"), "md5" : "953cd99404e81417f979113b4d95f044" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc1e"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOHIDUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5" : "39d3
0a08dcf6a4293ab753fabc5e7bb", "struct_output_size" : 0, "selector" : 4, "api" : "iokit_connect", "scalar_input" : BinData(0,""), "struct_in
ut_size" : 12, "struct_input" : BinData(0,"b8cBAHexAACGAAAA"), "md5" : "b0bc7f011b191692034592879717a62" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc1d"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOSurfaceRootUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5"
: "0ef85b86dd5ca879c0d1a2e2e05cefa8", "struct_output_size" : 1736, "selector" : 2, "api" : "iokit_connect", "scalar_input" : BinData(0,""),
"struct_input_size" : 12, "struct_input" : BinData(0,"AQAAAAEAAADmzrX/"), "md5" : "28546b6b07710c180dc1ca8530c1fdba" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc21"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOSurfaceRootUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5"
: "0ef85b86dd5ca879c0d1a2e2e05cefa8", "struct_output_size" : 1736, "selector" : 2, "api" : "iokit_connect", "scalar_input" : BinData(0,""),
"struct_input_size" : 12, "struct_input" : BinData(0,"AQAAAAEAAADoz7b/"), "md5" : "a2f80e85e18b77fd512ee1af2ccfef3b" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc22"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOHIDUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5" : "39d3
0a08dcf6a4293ab753fabc5e7bb", "struct_output_size" : 0, "selector" : 4, "api" : "iokit_connect", "scalar_input" : BinData(0,""), "struct_in
ut_size" : 12, "struct_input" : BinData(0,"b70BAHmsAACGAAAA"), "md5" : "deec511718de200b85575a0104e6ce7c" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc23"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOHIDUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5" : "39d3
0a08dcf6a4293ab753fabc5e7bb", "struct_output_size" : 0, "selector" : 4, "api" : "iokit_connect", "scalar_input" : BinData(0,""), "struct_in
ut_size" : 12, "struct_input" : BinData(0,"d7oBAHuqAACGAAAA"), "md5" : "bc779792f42c7b2dbfa8ade6c76df639" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc25"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOSurfaceRootUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5"
: "0ef85b86dd5ca879c0d1a2e2e05cefa8", "struct_output_size" : 1736, "selector" : 2, "api" : "iokit_connect", "scalar_input" : BinData(0,""),
"struct_input_size" : 12, "struct_input" : BinData(0,"AQAAAAEAAADnz7b/"), "md5" : "a8a9a56d3afb7fcf6abbdbfd6d6b9af6" }
{ "_id" : ObjectId("5bd963e550ffb410cef3fc24"), "kern_ret" : 0, "ool_input" : BinData(0,""), "proc_name" : "WindowServer", "ool_input_size"
: 0, "scalar_input_count" : 0, "class_name" : "IOSurfaceRootUserClient", "ool_output_size" : 0, "scalar_output_count" : 0, "no_content_md5"
: "0ef85b86dd5ca879c0d1a2e2e05cefa8", "struct_output_size" : 1736, "selector" : 2, "api" : "iokit_connect", "scalar_input" : BinData(0,""),
"struct_input_size" : 12, "struct_input" : BinData(0,"BAAAAAEAAADnz7b/"), "md5" : "9355e6a32399f121646ef0e532f8b6c3" }
```


IOKit Connect Call Data

Passive API S Corpus

- Syzkaller uses ‘program’ to fuzz instead of single API, ‘program’ is a sequence of API calls.

```
setsockopt$inet_int(0xffffffffffffffff, 0x0, 0x2, &(0x7f0000000040), 0x2)
__semwait_signal_nocancel(0x0, 0xfffffffffffffe00, 0x0, 0x3, 0x1, 0x0)
psynch_rw_upgrade(&(0x7f0000000000)="d268a934bfdd49a4fda7a6414c03a3e3f7d884a11ad3f10119aeeab06d3ed75d19afad6219b0378b84d546e17b646e3aeae9"
, 0x4, 0x663, 0xfffffffffffffffd, 0x7f)
socket$inet_icmp(0x2, 0x2, 0x1)
socket$inet_icmp_raw(0x2, 0x3, 0x1)
pselect_nocancel(0x1, &(0x7f00000001c0)={0x81, 0x6, 0x75d, 0x6, 0x1, 0x7fffffff, 0xce5d, 0x9}, &(0x7f0000000300)={0x9, 0x5, 0x9,
0x80000001, 0xe53d, 0x4, 0x8001, 0x4}, &(0x7f00000008c0)={0x0, 0x7fffffff, 0x20, 0x3, 0x7, 0x2, 0x1}, &(0x7f0000000900)={0x1000},
&(0x7f0000000940))
fcntl$setflags(0xffffffffffffffff, 0x2, 0x1)
bind(0xffffffffffffffff, &(0x7f0000000c00)=@in={0x2, 0x3, @local={0xac, 0x14, 0x0}}, 0x3cc)
sysctl$KERN_COREFILE(&(0x7f0000000cc0), 0x2, &(0x7f0000000d00)="cb006e917f50affe0e950086043816", &(0x7f0000000d80)=0xf,
&(0x7f0000000dc0), 0x0)
workq_open(0x0)
pselect_nocancel(0x5, &(0x7f0000000740)={0xfffffffffffffffe, 0x6, 0x0, 0xffe0000000000000, 0xd83, 0x0, 0xaad, 0x5},
&(0x7f0000000780)={0x40, 0x9, 0x38000000000000, 0xe1, 0x1ff, 0x0, 0x7}, &(0x7f00000007c0)={0x4, 0x6619, 0x4, 0x10001, 0x2, 0xffff, 0x0,
0x77f}, &(0x7f0000000800)={0xf71b, 0x8001}, &(0x7f0000000840))
workq_kernreturn(0x0, &(0x7f0000000880)="08000200000000004ab1426cc122c6d300a52334c7381aad57", 0x0, 0x5)
r0 = socket$inet_udp(0x2, 0x2, 0x0)
sigaction(0xb, &(0x7f0000000000)={0x81}, &(0x7f0000000040))
linkat(0xffffffffffffffff, &(0x7f0000000180)='./file1\x00', 0xffffffffffffffff, &(0x7f0000000200)='./file1\x00', 0x0)
setsockopt$inet_MCAST_LEAVE_GROUP(0xffffffffffffffff, 0x0, 0x51, &(0x7f00000000c0)={0x0, {{0x2, 0x2, @multicast2}}}, 0x90)
getsockopt$inet_mreqn(0xffffffffffffffff, 0x0, 0xd, &(0x7f0000000500)={@remote, @rand_addr}, &(0x7f0000000080)=0xc)
stat(&(0x7f0000000340)='./file1\x00', &(0x7f0000000380))
workq_kernreturn(0x20, &(0x7f0000000040), 0x3, 0x80000ca6)
setrlimit(0x3, &(0x7f00000004c0)={0x8, 0x76})
bind(r0, &(0x7f0000000240)=@un=@abs, 0x8)
sysctl$VM_MACHFACTOR(&(0x7f00000002c0), 0x2, &(0x7f0000000400)="ee", &(0x7f0000000440)=0x1, &(0x7f0000000480), 0x0)
shmat(0x0, &(0x7f0000ffb000/0x4000)=nil, 0x2000)
```

Passive API S Corpus

- Convert: Record a sequence of API calls, not just a single API call.


```
{ "_id" : ObjectId("5c8764fa10925490898c018b"), "p0" : "2e276f82d53fdd5987360924875abb9e", "xnu_ver" : 1802, "length" : 1, "api" : "program"
, "signature" : "83600dcedb93b73faa3a828f4d49d5bc" }
{ "_id" : ObjectId("5c8764fa10925490898c018d"), "p2" : "6686cb221dc448a8f3f4ea74bf034435", "p3" : "8cb4bcfb7d447d732be182cfd8a7bf2c", "p0" :
 "5b5422b211c0fc4f37b7a258250f7ab1", "p1" : "d4fb2c6a468ee785262e315357767c5c", "xnu_ver" : 1802, "length" : 4, "api" : "program", "signatur
e" : "cc3667c9e7ffee790ed61e1f268c6390" }
{ "_id" : ObjectId("5c8764fa10925490898c0190"), "p2" : "dd21d73c40d33159bcc47dfdf2d46757", "p3" : "056c9d6906faf8324764f439cdea9bc5", "p0" :
 "ac75ea8f16e72fd303b4d5b8c1e77825", "p1" : "4296c72b40cfe3f5ffab816e52bccd0a", "p4" : "006164db57eec0702d93e6cb5cefef1e", "xnu_ver" : 1802,
 "length" : 5, "api" : "program", "signature" : "7cf7d4dd14d728d44258b44e47cded2d" }
{ "_id" : ObjectId("5c8764fb10925490898c0191"), "p0" : "1115961f47d7123cc0a9dbbce472721f", "xnu_ver" : 1802, "length" : 1, "api" : "program"
, "signature" : "2504fcca3ac535f71014966d2b1d3997" }
{ "_id" : ObjectId("5c8764fb10925490898c0194"), "p2" : "60db38d1274ac1c9ae5131bbaffb6c81", "p3" : "056c9d6906faf8324764f439cdea9bc5", "p0" :
 "4cab7e8ed4e0ecbc1155ac59f447557a", "p1" : "aa8385a5f904be3ae9445c1911efd324", "p4" : "006164db57eec0702d93e6cb5cefef1e", "xnu_ver" : 1802,
 "length" : 5, "api" : "program", "signature" : "77cd181d10e3627d3fe63ef13b464768" }
{ "_id" : ObjectId("5c8764fc10925490898c0197"), "p2" : "114cf03707a269a3a00ca9b49f57371b", "p3" : "6686cb221dc448a8f3f4ea74bf034435", "p0" :
 "6c8a185fe0212b5181db3842cbc6022a", "p1" : "f9db95514fb973549e0383916a242f90", "p4" : "c7f15c4d124b2334ffdf043ebfe23035", "xnu_ver" : 1802,
 "length" : 5, "api" : "program", "signature" : "44b8b1bfd32ce4e4ee81f7006479e2df" }
{ "_id" : ObjectId("5c8764ff10925490898c019a"), "p2" : "114cf03707a269a3a00ca9b49f57371b", "p3" : "6686cb221dc448a8f3f4ea74bf034435", "p0" :
 "bb7d6a8d48a7bea29c6783b5971f6d5a", "p1" : "f9db95514fb973549e0383916a242f90", "p4" : "e31fe6c7b469a3e226386e589c135c75", "xnu_ver" : 1802,
 "length" : 5, "api" : "program", "signature" : "e2bc749285a04b464bfe3566df463aae" }
{ "_id" : ObjectId("5c87650110925490898c019b"), "p0" : "7a8ddde33b15f05e09e25278c46e17e3", "xnu_ver" : 1802, "length" : 1, "api" : "program"
, "signature" : "6515807d0e6feacb19a9836e48994268" }
{ "_id" : ObjectId("5c87650210925490898c019f"), "p2" : "5b5422b211c0fc4f37b7a258250f7ab1", "p3" : "d4fb2c6a468ee785262e315357767c5c", "p0" :
 "064aff8ac68140b06de15b905fc2a8f1", "p1" : "e295118f1c935d0babbf9a654950c316", "p4" : "6686cb221dc448a8f3f4ea74bf034435", "p5" : "05a66f683
2863dec4f43af6aa188478b", "xnu_ver" : 1802, "length" : 6, "api" : "program", "signature" : "f0d81e40f61445a8217b5794f4c1ff02" }
{ "_id" : ObjectId("5c87650310925490898c01a1"), "p0" : "b11acbed172eef29ec367ed23ac403af", "xnu_ver" : 1802, "length" : 1, "api" : "program"
, "signature" : "32473c025ac9dda2cecb1f205aed6f2b" }
{ "_id" : ObjectId("5c87654910925490898c01a6"), "p2" : "132765b06d7ebc35bb72450e55c72fe1", "p0" : "5eeb56dddbdd4eb85ad0a2c795ff4ea3", "p1" :
 "55f4362f47d705c4b2af7bbd55512587", "xnu_ver" : 1802, "length" : 3, "api" : "program", "signature" : "e736f4f96d4ff7c964209b173eacf895" }
{ "_id" : ObjectId("5c87654910925490898c01a8"), "p2" : "006164db57eec0702d93e6cb5cefef1e", "p0" : "3766cfe029870129792aad3b3313f73f", "p1" :
 "056c9d6906faf8324764f439cdea9bc5", "xnu_ver" : 1802, "length" : 3, "api" : "program", "signature" : "bc3718533df5cfd0b923f74086965a6a" }
```

Passive API S Corpus

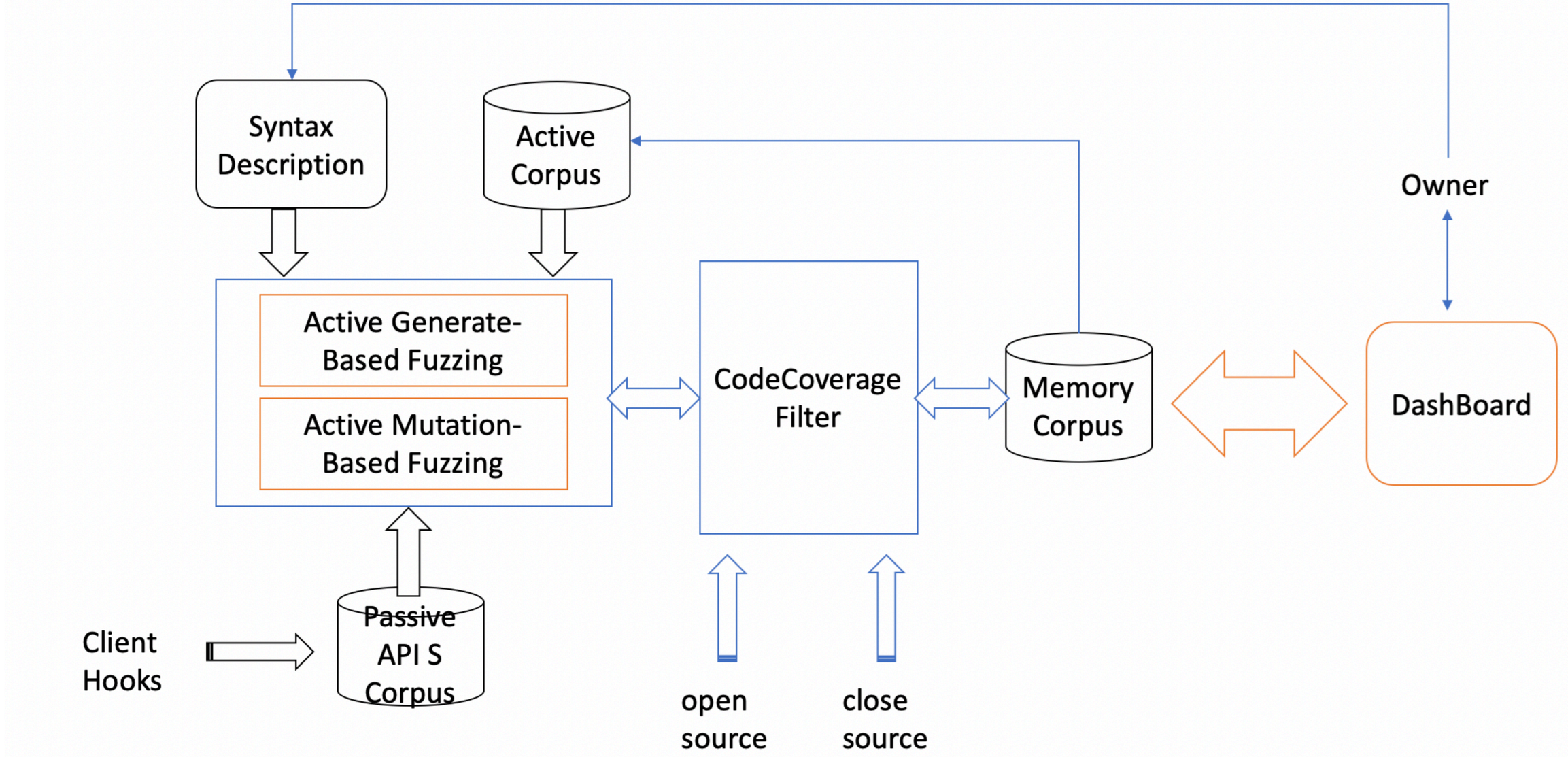
- Make: Use the above data to make syzkaller friendly corpus

```
r0 = syz_open_service$IGAccelGLContext(&(0x7f0000000100)='IntelAccelerator\x00', 0x1)
syz_call_method$submit_data_buffers(r0, 0x2, &(0x7f0000000200)=nil, 0x0, &(0x7f0000000200)="000000
0002000000d8080000000000c0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000", 0x88, &(0x7f0000000288),
0x0, &(0x7f0000000288), 0x78)
r1 = syz_open_service$IOSurfaceRootUserClient(&(0x7f0000000300)='IOSurfaceRoot\x00', 0x0)
syz_call_method$s_increment_surface_use_count(r1, 0xe, &(0x7f0000000400)=[0x11a], 0x1, &(0x7f00000
00408)=nil, 0x0, &(0x7f0000000408), 0x0, &(0x7f0000000408), 0x0)
syz_call_method$s_decrement_surface_use_count(r1, 0xf, &(0x7f0000000408)=[0xb0], 0x1, &(0x7f0000000
0410)=nil, 0x0, &(0x7f0000000410), 0x0, &(0x7f0000000410), 0x0)
syz_call_method$s_create_port_from_surface(r1, 0x1c, &(0x7f0000000410)=[0x115], 0x1, &(0x7f0000000
418)=nil, 0x0, &(0x7f0000000418), 0x1, &(0x7f0000000420), 0x0)
syz_call_method$s_get_value(r1, 0xa, &(0x7f0000000420)=nil, 0x0, &(0x7f0000000420)="1a010000fe7f00
0000", 0x9, &(0x7f0000000429), 0x0, &(0x7f0000000429), 0x0)
```

IOKit Fuzz Summary

- We can combine the active and passive IOKit fuzz now:
  - We have the IOKit open & call **syntax description** by R.E.
  - We have the passive sequence **corpus** by hooks
  - We have the **code coverage** of close source IOKit modules
  - We extend the ability of syzkaller to support fuzzing IOKit

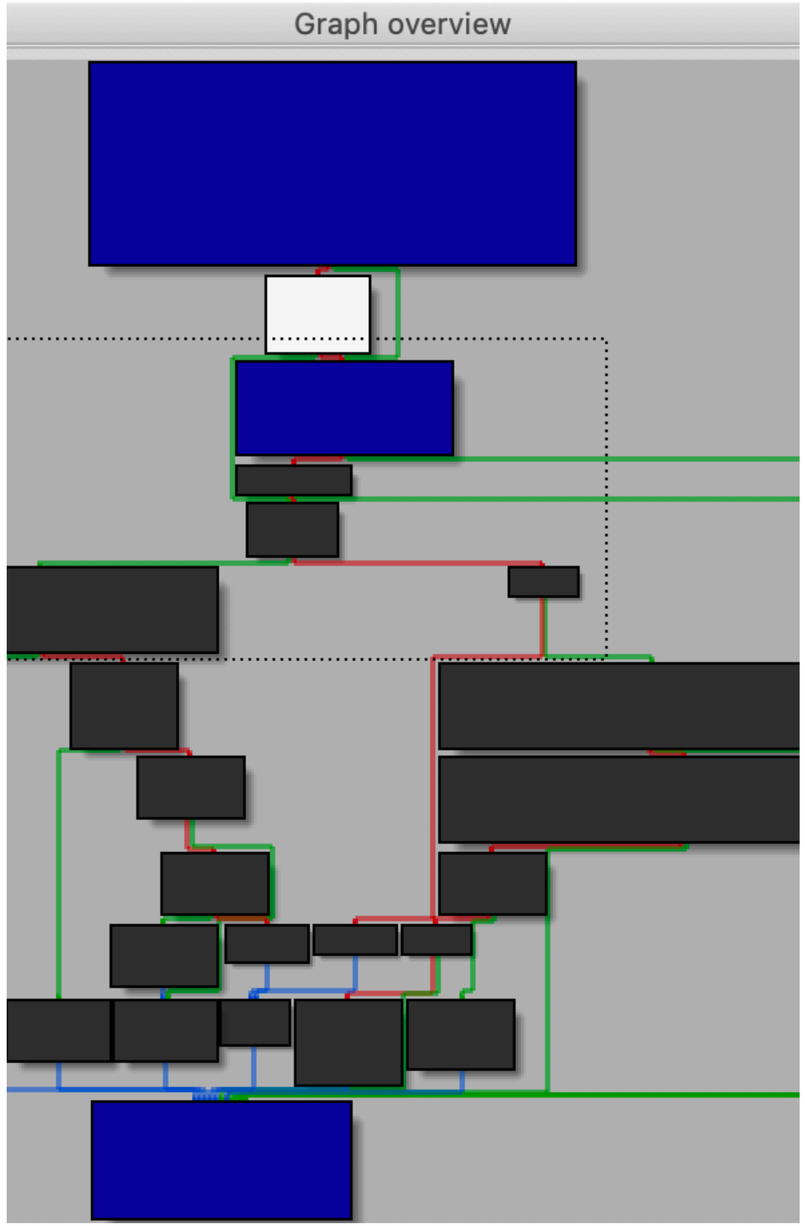
- The combination advantages:
  - Passive corpus can be used to fuzz system alone. It doesn’t need the active API patterns developed by researchers.
  - Passive corpus can help review if active patterns are correct by reverse engineering.
  - Passive corpus give the chances that even the patterns are wrong, we can still fuzz in the right direction based on genetic algorithm.



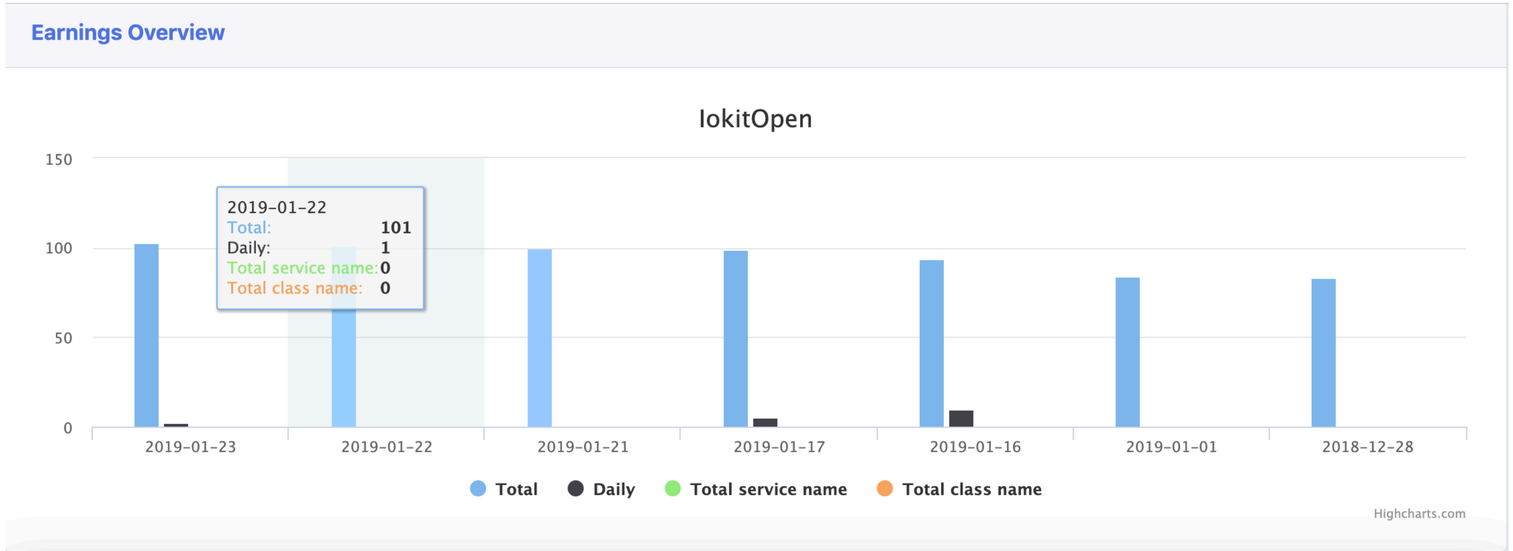
IOKit Fuzzing Architecture

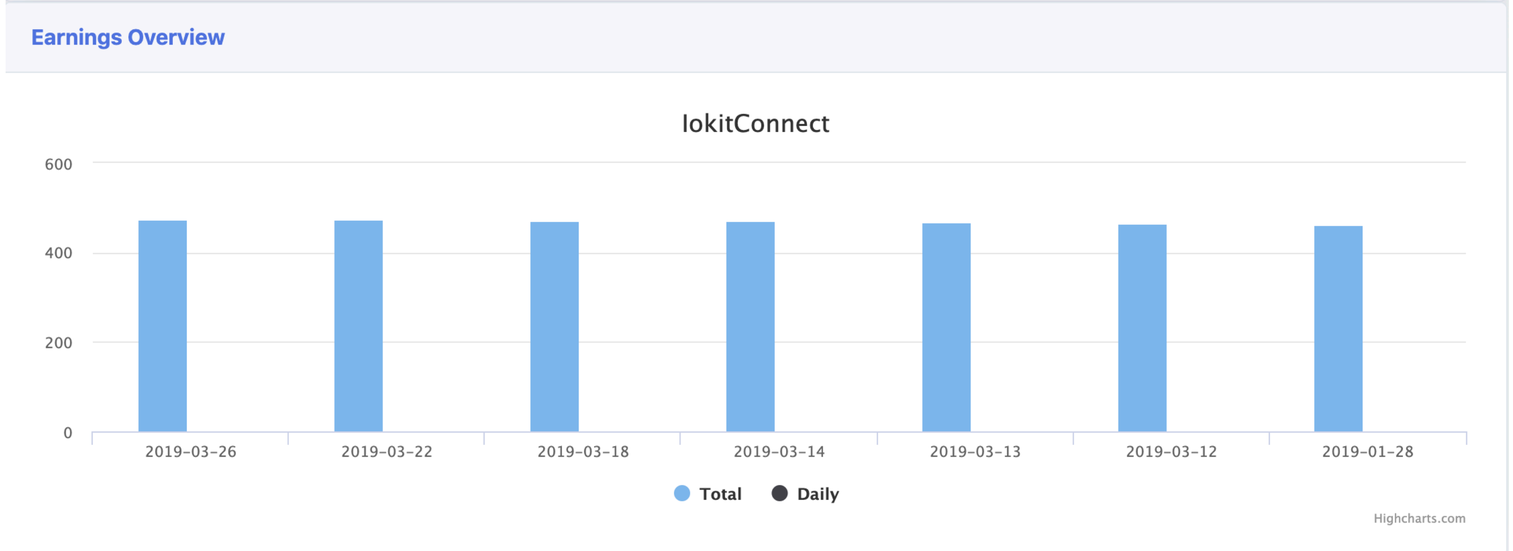
Fuzzing Visualization

Coverage: the first class citizen

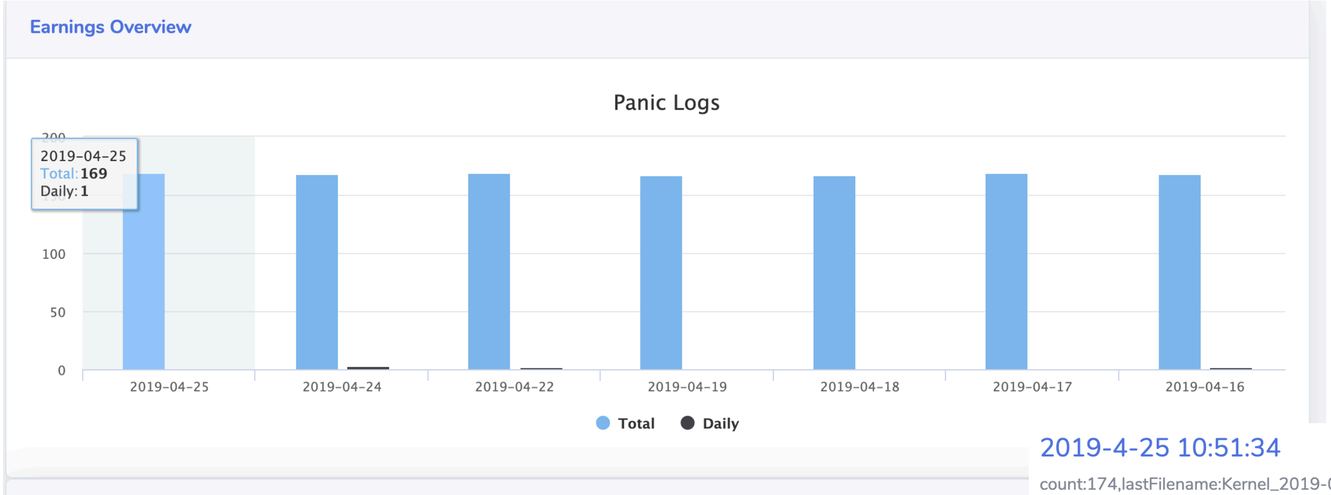


Passive Fuzz





Exciting updates has exciting crashes



Earnings Overview

2019-4-25 10:51:34
2019-4-25 5:11:28
2019-4-24 15:36:23
2019-4-24 14:48:26

2019-4-25 10:51:34

count:174,lastFilename:Kernel_2019-03-14-221147_panicall1s-Mac-4.panic,sigNature:8dc849664c2d1f3dd8b45b6ab

Anonymous UUID: 039B94D9-F271-8A3F-BAE4-C8BF9D6B5BEE

Thu Mar 14 22:11:47 2019

*** Panic Report ***
panic(cpu 0 caller 0xffffff800a0da29d): Kernel trap at 0xffffff7f8ad91ff9, type 14=page fault, registers:
CR0: 0x000000008001003b, CR2: 0x0000000000000000, CR3: 0x000000007076c04d, CR4: 0x00000000001606e(
RAX: 0xffffff7f8ad91fd8, RBX: 0x00000000e00002c2, RCX: 0xffffff7f8ad79ef8, RDX: 0xffffff886dccbb48
RSP: 0xffffff886dccbaf0, RBP: 0xffffff886dccbb30, RSI: 0x0000000000000002, RDI: 0x0000000000000000
R8: 0x0000000000000000, R9: 0x0000000000000000, R10: 0xffffff801801e61c, R11: 0xffffff8014fe860c
R12: 0xffffff80185f4250, R13: 0x0000000000000000, R14: 0xffffff80185f4250, R15: 0xffffff886dccbb48
RFL: 0x0000000000010246, RIP: 0xffffff7f8ad91ff9, CS: 0x0000000000000008, SS: 0x0000000000000010
Fault CR2: 0x0000000000000000, Error code: 0x0000000000000000, Fault CPU: 0x0 VMM, PL: 0, VF: 0

Backtrace (CPU 0), Frame : Return Address
0xffffff886dccb5c0 : 0xffffff8009faeb0d mach_kernel : _handle_debugger_trap + 0x48d
0xffffff886dccb610 : 0xffffff800a0e8653 mach_kernel : _kdp_i386_trap + 0x153
0xffffff886dccb650 : 0xffffff800a0da07a mach_kernel : _kernel_trap + 0x4fa
0xffffff886dccb6c0 : 0xffffff8009f5bca0 mach_kernel : _return_from_trap + 0xe0
0xffffff886dccb6e0 : 0xffffff8009fae527 mach_kernel : _panic_trap_to_debugger + 0x197
0xffffff886dccb800 : 0xffffff8009fae373 mach_kernel : _panic + 0x63
0xffffff886dccb870 : 0xffffff800a0da29d mach_kernel : _kernel_trap + 0x71d

Coverage in mind

Coverage in mind

IOAcceleratorFamily2_new

- Target Binary: IOAcceleratorFamily2_new
- Coverage Name: IOAcceleratorFamily2_new_drcov.log
- Coverage File: IOAcceleratorFamily2_new_drcov.log
- Database Coverage: 6.99%
- Table Coverage: 6.99%
- Timestamp: Thu [illegible] 10:21:57 2019

check update delete

AppleFDEKeyStore_new

- Target Bi[illegible]AppleFDEKeyStore_new
- Coverage Name: AppleFDEKeyStore_ne[illegible]
- Coverage File: AppleFDEKeyStore_new_
- Database Coverage: 12.87%
- Table Coverage: 12.87%
- Timestamp: Fri Mar 29 07:50:54 2019

check update delete

AppleMobileFileIntegrity_new

- Target Binary: AppleMobileFileIntegrity_new
- Coverage Name: AppleMobileFileIntegrity_new_drcov.log
- Coverage File: AppleMobileFileIntegrity_new_drcov.log
- Database Coverage: 2.44%
- Table Coverage: 2.44%
- Timestamp: Fri Mar 29 07:53:18 2019

check update delete

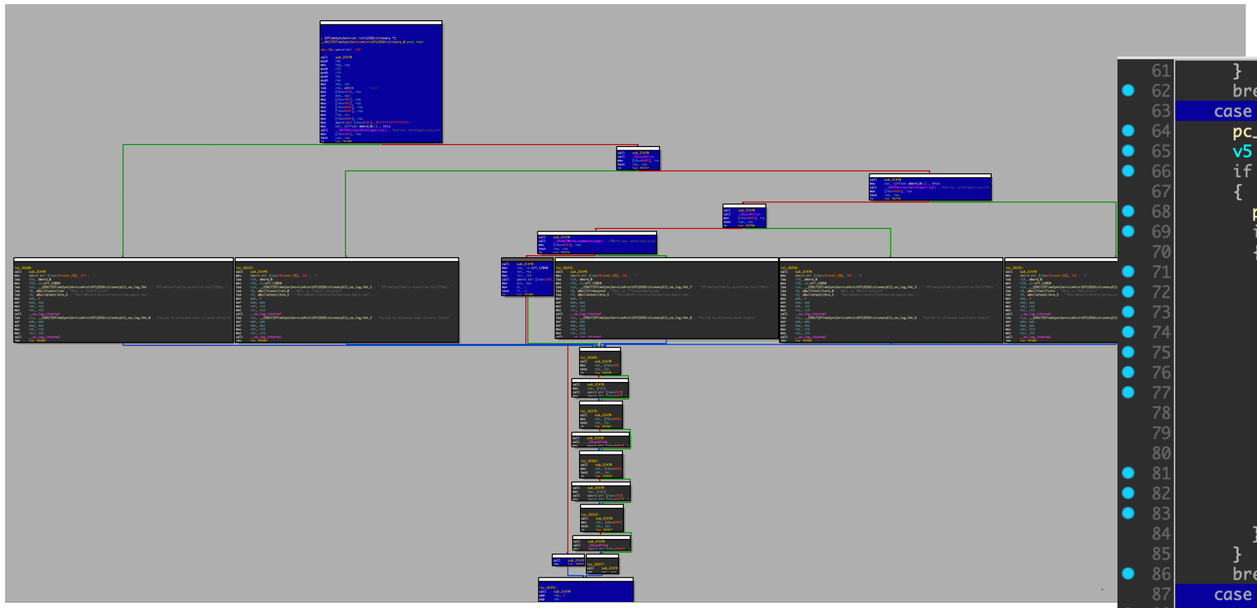
AppleHDA

- Target Binary: AppleHDA
- Coverage Name: AppleHDA_drcov.log
- Coverage File: AppleHDA_drcov.log
- Database Coverage: 0.17%
- Table Coverage: 0.17%
- Timestamp: Fri Mar 29 03:59:54 2019

check update delete

60.49	IOAccelMemoryMap::prepare(void)
60.00	IOAccelMemoryInfoUserClient::s_allowed_to_gather(void)
56.00	IOAccelDisplayPipeUserClient2::s_transaction_set_pipe_pregamma_table(IOAccelDisplayPipeUserClient2*,void *,IOExternalMethodArguments *)
55.56	IOAccelSysMemory::complete(void)
55.56	IOAccelSurface2::pickPresentType(uint)
55.48	IOAccelSurface2::set_scaling(uint,IOAccelSurfaceScaling *)
53.54	IOAccelSharedUserClient2::externalMethod(uint,IOExternalMethodArguments *,IOExternalMethodDispatch *,OSObject *,void *)
51.61	IOAccelMemoryInfoUserClient::s_purge_all_vid_memory(OSObject *,void *,IOExternalMethodArguments *)
51.58	IOAccelDevice2::externalMethod(uint,IOExternalMethodArguments *,IOExternalMethodDispatch *,OSObject *,void *)
50.87	IOAccelSysMemory::withOptions(IOGraphicsAccelerator2 *,task *,IOAccelShared2 *,IOAccelResource2 *,uint,ulong long,bool)
50.10	IOGraphicsAccelerator2::newUserClient(task *,void *,uint,IOUserClient **)
49.48	IOAccel2DContext2::set_surface(uint,eIOAccelContextModeBits)
47.22	IOAccelMemoryInfoUserClient::s_gather_memory_data_totals(OSObject *,void *,IOExternalMethodArguments *)
46.67	IOAccelSysMemoryList::removeMemory(IOAccelSysMemory *)
46.43	IOAccelGLContext2::externalMethod(uint,IOExternalMethodArguments *,IOExternalMethodDispatch *,OSObject *,void *)
45.00	IOAccelCommandQueue::externalMethod(uint,IOExternalMethodArguments *,IOExternalMethodDispatch *,OSObject *,void *)
44.00	IOAccelSurface2::reset_req_bits(void)
43.90	IOAccelSurface2::set_shape(eIOAccelSurfaceShapeBits,uint,IOAccelDeviceRegion *,ulong long)
42.70	IOAccelSurface2::attach_buffer_at_index(int,IOAccelResource2 *)
41.86	IOAccelBufferMemoryDescriptorPool2::newWiredDescriptorsWithLength(ulong long,uint *,task *,IOAccelEvent *,bool,bool *)
41.67	IOAccelMemory::add_mapping(IOAccelMemoryMap *)
39.47	IOAccelSurface2::prune_buffers(void)
38.82	IOAccelSysMemory::lockForCPUAccess(task *,uint)
37.50	IOAccelMemory::release(void)
37.14	IOAccelSurface2::get_current_backbuffer_index(void)
35.21	IOAccelSurface2::set_id_mode(uint,uint)
35.21	IOAccelSharedUserClient2::get_resource_info(uint,IOAccelGetResourceInfoReturnData *,uint *)
35.20	IOAccelSurface2::surface_lock_options(eLockType,uint,IOAccelSurfaceInformation *,ulong long)
34.22	IOAccelSharedUserClient2::set_resource_purgeable(uint,eIOAccelResourcePurgeable,eIOAccelResourcePurgeable*)
33.69	IOAccelMemory::createMappingInTask(IOAccelTask *,uint)
33.33	IOAccelMemoryMap::getGPUVirtualAddress(void)
30.36	IOAccelEventMachineFast2::finishEvent(IOAccelEvent *)
29.41	IOGraphicsAccelerator2::acceleratorDidLock(char const*,int)
29.41	IOGraphicsAccelerator2::acceleratorWillUnlock(char const*,int)
29.29	IOAccelEventMachineFast2::finishStamp(int)
28.74	IOAccelMemoryInfoUserClient::s_gather_memory_data(OSObject *,void *,IOExternalMethodArguments *)
26.76	IOGraphicsAccelerator2::find_vram_descriptor(void)

Lighthouse



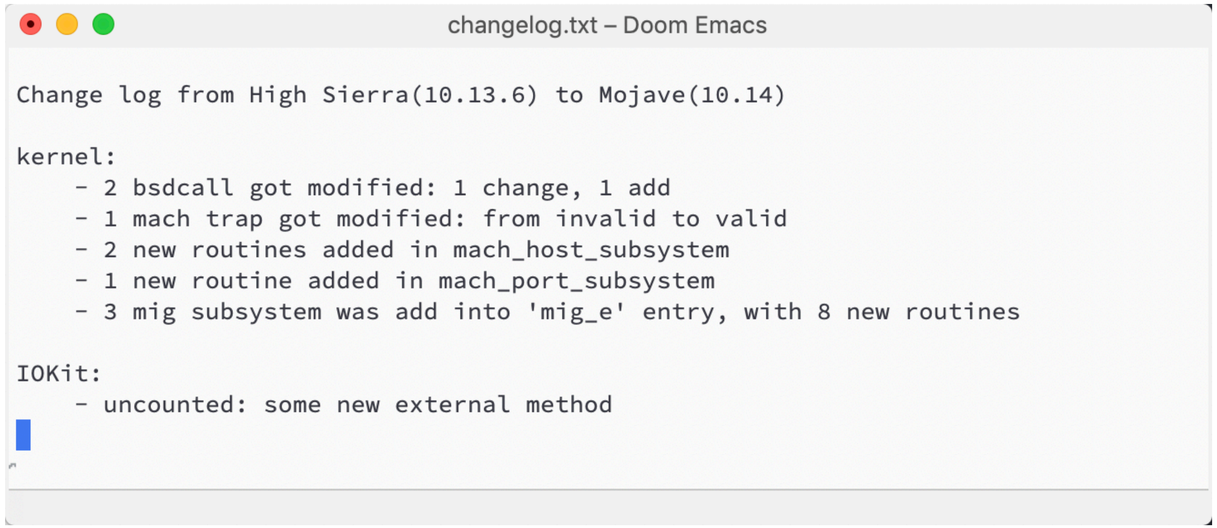
```
61      }
62      break;
63    case 1:
64      pc_trace(a1);
65      v5 = -536870206;
66      if ( *(_DWORD *)(v3 + 40) == 2 )
67      {
68        pc_trace(a1);
69        if ( *(_DWORD *)(v3 + 80) == 2 )
70        {
71          pc_trace(a1);
72          v8 = *(_QWORD **)(v3 + 32);
73          v9 = v8[1];
74          v32 = v8[1];
75          v10 = *(_QWORD **)(v3 + 72);
76          a1 = (IOTimeSyncUserClient *)*((_QWORD *)a1 + 28);
77          *v10 = (*(__int64 (__fastcall **)(IOTimeSyncUserClient *, _QWORD, unsigned int *))(*(_QWORD *)a1 + 2136LL))(
78                   a1,
79                   *v8,
80                   &v32);
81          v32 = v9;
82          *(_QWORD *)(*(_QWORD *)(v3 + 72) + 8LL) = (unsigned int)v9;
83          goto LABEL_42;
84        }
85      }
86      break;
87    case 2:
88      pc_trace(a1);
89      v5 = -536870206;
90      if ( *(_DWORD *)(v3 + 80) == 1 )
91      {
92        pc_trace(a1);
93        a1 = (IOTimeSyncUserClient *)*((_QWORD *)a1 + 28);
94        **(_QWORD **)(v3 + 72) = (*(unsigned int (__fastcall **)(IOTimeSyncUserClient *))(*(_QWORD *)a1 + 2160LL))(a1);
95        goto LABEL_42;
96      }
97      break;
98    case 3:
99      pc_trace(a1);
100     v5 = -536870206;
101     if ( *(_DWORD *)(v3 + 80) == 2 )
0001FFAD __ZN20IOTimeSyncUserClient14externalMethodEjP25IOExternalMethodArgumentsP24IOExternalMethodDispatchP8OSObjec
```

After update

For Programmer:

- Fix old bugs

- Introduce new feature, and new API

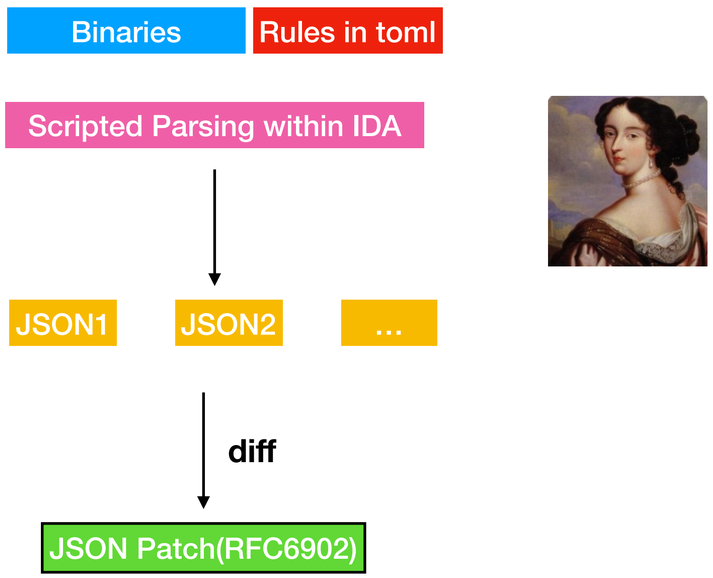


For Security Researcher:

- Do diff to find n-Day

- Do diff to find new BUG and new Attack Surface

Diff on key structure



kernel.min.toml – Doom Emacs

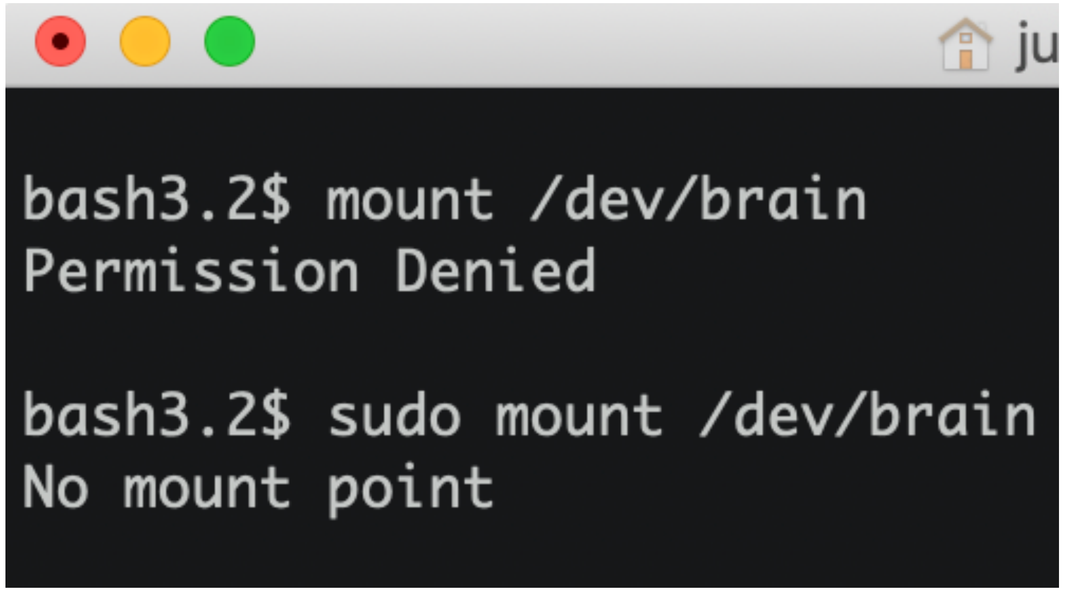
```
[info]
    name = "kernel"
    module = "kernel"
    version = "universal"

[entries]
    [entries.mach_trap]
        parse_method = "table"
        base = 'addr_of_sym/_mach_trap_table/0x00'
        count = 'sym_val/_mach_trap_count/0x00:uint32'
        type = 'array/struct.mach_trap_t'

    [entries.bsd_call]
        parse_method = "table"
        base = 'addr_xref_to_sym/_exit/-0x18'
        count = 'sym_val/_nsysent/0x00:uint32'
        type = 'array/struct.sysent'

[struct]
    [struct.mach_trap_t]
        mach_trap_arg_count = '0x00:int32'
        padding_1 = '0x04:padding/uint32'
        mach_trap_function = '0x08:ptr/func'
        mach_trap_arg_munge32 = '0x10:ptr/func'
        mach_trap_u32_words = '0x18:int32'
        padding_2 = '0x1b:padding/uint32'

    [struct.sysent]
        sy_call = '0x00:ptr/func'
        sys_arg_munge32 = '0x08:ptr/func'
        sy_return_type = '0x10:int32'
        sy_narg = '0x14:int16'
        sy_arg_bytes = '0x16:int16'


978  Monitor/worker/config/rules/kernel.min.toml  32:29 All_4 LF
```

Output — junzhi@pwn-orz-macbook-pro — ..worker/Output — -zsh — 126×29

```
➜  Output git:(master) ✗ jsondiff 10.13.6_17G65/kernel.json 10.14_18A391/kernel.json | jq
[
  {
    "op": "add",
    "path": "/mig_subsystems/_memory_entry_subsystem",
    "value": {
      "reserved": 0,
      "server": "_memory_entry_server_routine",
      "start": 4900,
      "maxsize": 56,
      "routine": {
        "0": {
          "max_reply_msg": 48,
```

Fuzzing Daily Operation

We are not developing this to see just pictures.



After looking lots of charts

Further reverse engineering

Quenching: Modify API Syntax Description for fuzz and run again

Add some tweaks for the fuzzing environment.

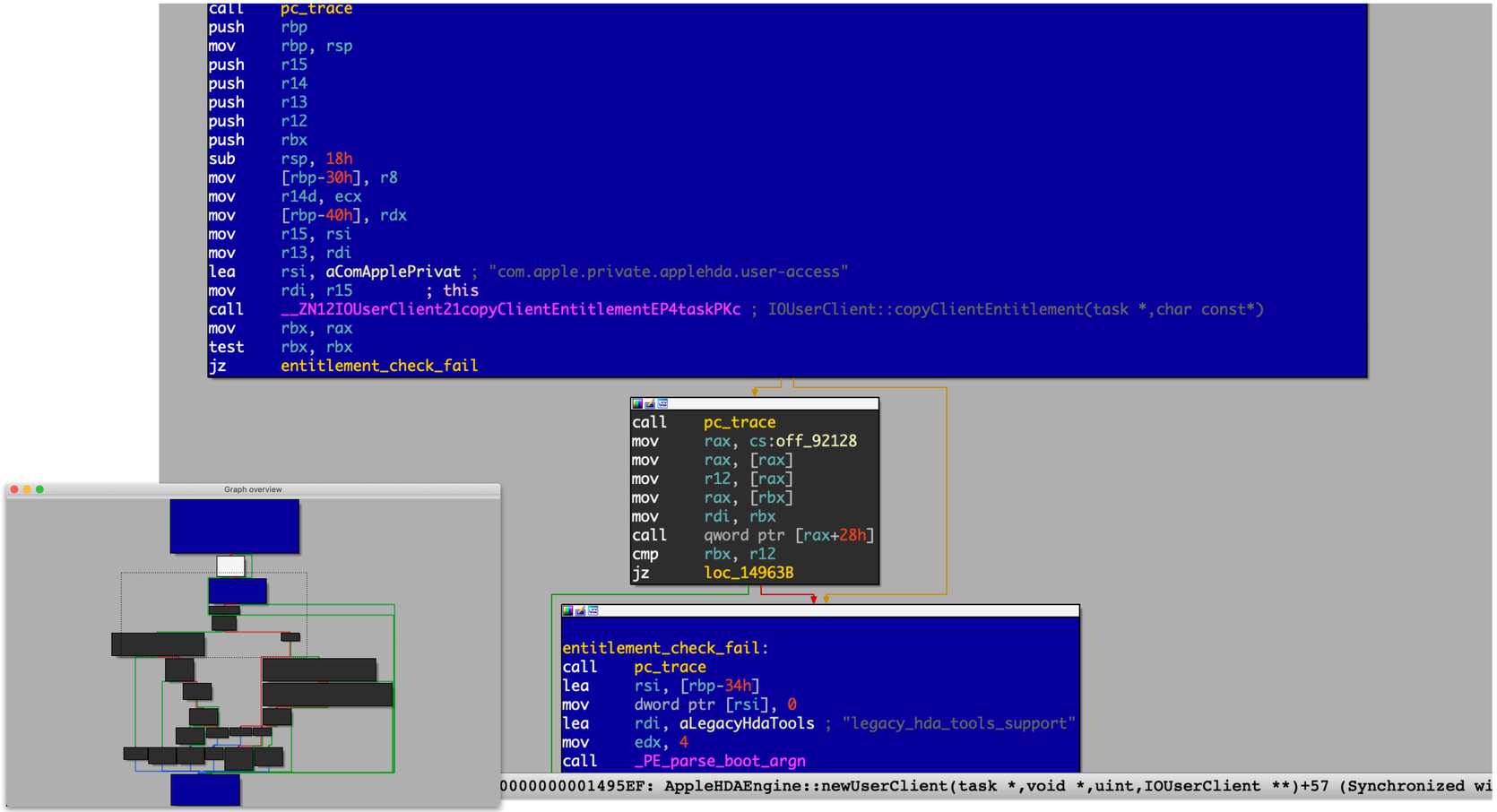
Demo: AppleHDA

Limitation:

1. Check entitlement
2. Check Boot Arguments

Tweak:

1. Add an entitlement(private? fail?)
2. Change boot args



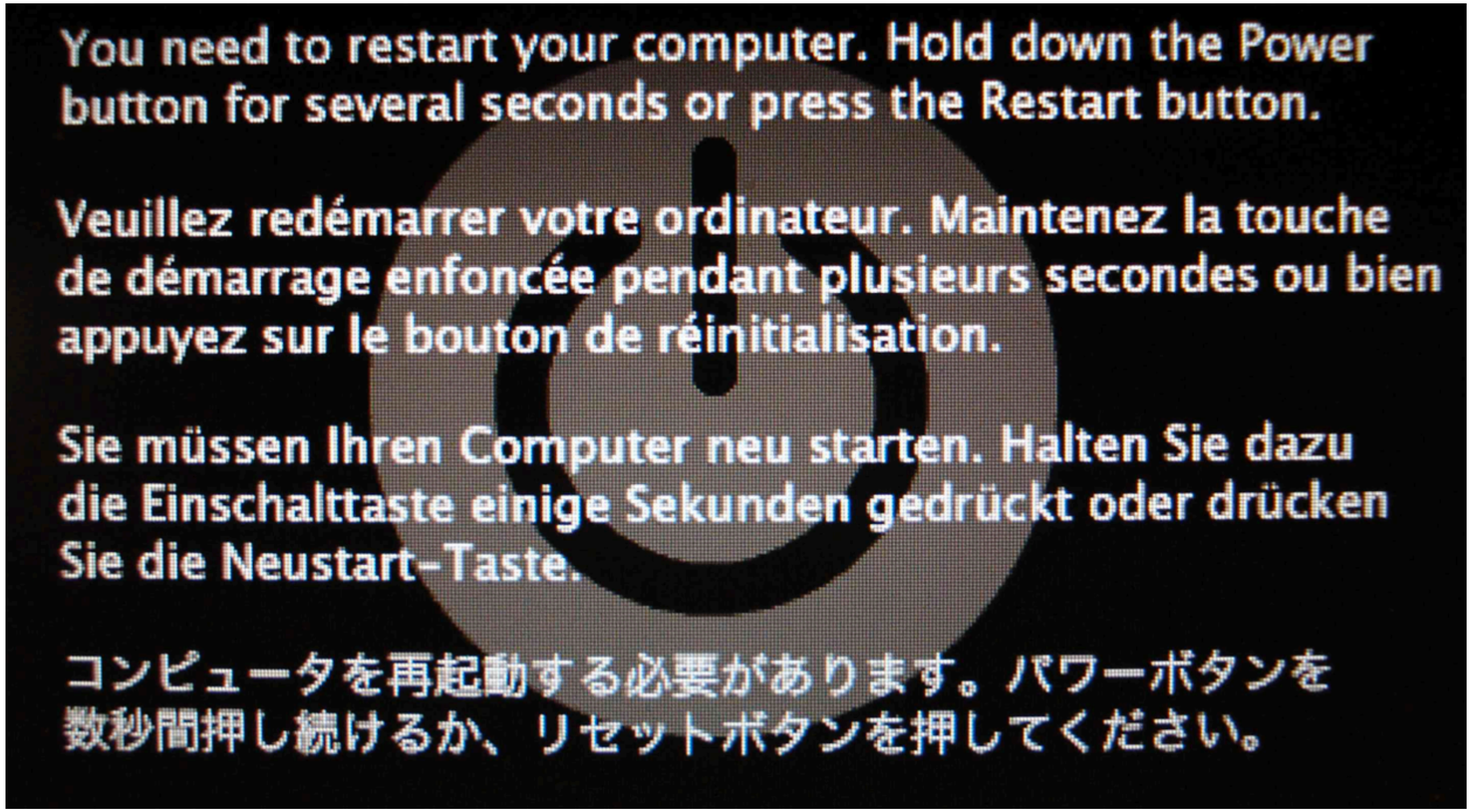
Demo: An exciting update

A diff between High Sierra(10.13.6) & Mojave(10.14)

In facet, the implementation code of these mig subsystem already lies in kernel since High Sierra but remains unreferenced.

It was then add added as the element of array “mig_e” since Mojave. Which means now it come to work.

So we add some new **API Syntax Description**


You need to restart your computer. Hold down the Power button for several seconds or press the Restart button.
Veuillez redémarrer votre ordinateur. Maintenez la touche de démarrage enfoncée pendant plusieurs secondes ou bien appuyez sur le bouton de réinitialisation.
Sie müssen Ihren Computer neu starten. Halten Sie dazu die Einschalttaste einige Sekunden gedrückt oder drücken Sie die Neustart-Taste.
コンピュータを再起動する必要があります。パワーボタンを数秒間押し続けるか、リセットボタンを押してください。

Case Study

CVE-2018-4435

```
int
shmget(struct proc *p, struct shmget_args *uap, int32_t *retval)
{
        int segnum, mode, error;
        int shmget_ret = 0;

        /* Auditing is actually done in shmget_allocate_segment() */

        SYSV_SHM_SUBSYS_LOCK();

        if ((shmget_ret = shminit())) {     ---------------------(1.a)
                goto shmget_out;
        }

        mode = uap->shmflg & ACCESSPERMS;
        if (uap->key != IPC_PRIVATE) {
        again:
                segnum = shm_find_segment_by_key(uap->key);
                if (segnum >= 0) {
                        error = shmget_existing(uap, mode, segnum, retval);
                        if (error == EAGAIN)
                                goto again;
                        shmget_ret = error;
                        goto shmget_out;
                }
                if ((uap->shmflg & IPC_CREAT) == 0) {
                        shmget_ret = ENOENT;
                        goto shmget_out;
                }
        }
        shmget_ret = shmget_allocate_segment(p, uap, mode, retval); ---(1.b)
shmget_out:
        SYSV_SHM_SUBSYS_UNLOCK();
        return shmget_ret;
}
```

```
int
shminit(void)
{
        size_t sz;
        int i;

        if (!shm_inited) {
                /*
                 * we store internally 64 bit, since if we didn't, we would
                 * be unable to represent a segment size in excess of 32 bits
                 * with the (struct shmid_ds)->shm_segsz field; also, POSIX
                 * dictates this filed be a size_t, which is 64 bits when
                 * running 64 bit binaries.
                 */
                if (os_mul_overflow(shminfo.shmmni, sizeof(struct shmid_kernel), &sz)) {
                        return ENOMEM;
                }

                MALLOC(shmsegs, struct shmid_kernel *, sz, M_SHM, M_WAITOK);    -----------(2.a)
                if (shmsegs == NULL) {
                        return ENOMEM;
                }
                for (i = 0; i < shminfo.shmmni; i++) {
                        shmsegs[i].u.shm_perm.mode = SHMSEG_FREE;
                        shmsegs[i].u.shm_perm._seq = 0;
#if CONFIG_MACF
                        mac_sysvshm_label_init(&shmsegs[i]);
#endif
                }
                shm_last_free = 0;
                shm_nused = 0;
                shm_committed = 0;
                shm_inited = 1;
        }

        return 0;
}
```

CVE-2018-4435

```
static int
shmget_allocate_segment(struct proc *p, struct shmget_args *uap, int mode,
        int *retval)
{
        int i, segnum, shmid;
        kauth_cred_t cred = kauth_cred_get();
        struct shmid_kernel *shmseg;
        struct shm_handle *shm_handle;
        kern_return_t kret;
        mach_vm_size_t total_size, size, alloc_size;
        void * mem_object;
        struct shm_handle *shm_handle_next, **shm_handle_next_p;

    ...

    shmseg = &shmsegs[segnum];

    ...

    shmseg->u.shm_segsz = uap->size;
        shmseg->u.shm_cpid = p->p_pid;
        shmseg->u.shm_lpid = shmseg->u.shm_nattch = 0; ---------(3.a)
        shmseg->u.shm_atime = shmseg->u.shm_dtime = 0;

    ...

}
```

```
mov     [r8], ax
mov     rax, [r13+8]
mov     [rbx+r15+18h], rax ; shm_segsz
mov     rax, [rbp-80h]
mov     eax, [rax+10h]
mov     [rbx+r15+24h], eax ; shm_cpid
mov     word ptr [rbx+r15+28h], 0 ; shm_nattch, 0x28-0x2a
mov     dword ptr [rbx+r15+20h], 0 ; shm_lpid
mov     qword ptr [rbx+r15+34h], 0 ; shm_dtime
mov     qword ptr [rbx+r15+2Ch], 0 ; shm_atime
mov     rsi, [rbp-30h]
mov     r13, r8
call    _mac_sysvshm_label_associate
```

asm code around location 3.a

Field at Offset 0x2a is not set;
Actually it is the padding of structure.

CVE-2018-4435

```
#pragma pack(4)

struct user_shmid_ds {
        struct ipc_perm shm_perm;       /* operation permission structure */
        user_size_t     shm_segsz;      /* size of segment in bytes */
        pid_t           shm_lpid;       /* PID of last shared memory op */
        pid_t           shm_cpid;       /* PID of creator */
        short           shm_nattch; //@panicaII: offset 0x28, len: 2
        user_time_t     shm_atime;      //@panicalII: offset 0x2c
        user_time_t     shm_dtime;      /* time of last shmdt() */
        user_time_t     shm_ctime;      /* time of last change by shmctl() */
        user_addr_t     shm_internal;   /* reserved for kernel use */
};

struct user32_shmid_ds {
        struct ipc_perm shm_perm;       /* operation permission structure */
        uint32_t        shm_segsz;      /* size of segment in bytes */
        pid_t           shm_lpid;       /* PID of last shared memory op */
        pid_t           shm_cpid;       /* PID of creator */
        short           shm_nattch;     /* number of current attaches */
        uint32_t                shm_atime;      /* time of last shmat() */
        uint32_t                shm_dtime;      /* time of last shmdt() */
        uint32_t                shm_ctime;      /* time of last change by shmctl() */
        user32_addr_t   shm_internal;   /* reserved for kernel use */
};

#pragma pack()
```

The alignment makes the hole.

CVE-2018-4435

- Type: Kernel heap memory disclosure
- Thought: Can be used to search the XNU with such a pattern:
  - Kernel structure memory allocated without clear
  - Not all bytes are filled, e.g. padding, union structure
  - copy back to user mode

Found Case

```
int
_kernelrpc_mach_port_get_attributes_trap(struct _kernelrpc_mach_port_get_attributes_args *args)
{
    ...

    typeof(MACH_PORT_INFO_OUT[0]) info[max_count];  ---(1.a)

    ...

    rv = mach_port_get_attributes(task->itk_space, args->name, args->flavor, info, &count);
        if (rv == KERN_SUCCESS)
                rv = copyout(&count, CAST_USER_ADDR_T(args->count), sizeof(count));
        if (rv == KERN_SUCCESS && count > 0)
                rv = copyout(info, CAST_USER_ADDR_T(args->info), count * sizeof(info[0]));  ---(1.b

}
```

At location 1.a, max_count here is 17. At location 1.b, count can be 11.

```
kern_return_t
mach_port_get_attributes(
        ipc_space_t             space,
        mach_port_name_t        name,
        int                     flavor,
        mach_port_info_t        info,
        mach_msg_type_number_t  *count)
{
    ...

    case MACH_PORT_INFO_EXT: {
                mach_port_info_ext_t *mp_info = (mach_port_info_ext_t *)info;   ---(2.a)
                if (*count < MACH_PORT_INFO_EXT_COUNT)
                        return KERN_FAILURE;

                if (!MACH_PORT_VALID(name))
                        return KERN_INVALID_RIGHT;

                kr = ipc_port_translate_receive(space, name, &port);
                if (kr != KERN_SUCCESS)
                        return kr;
                /* port is locked and active */
                mach_port_get_status_helper(port, &mp_info->mpie_status);   ---(2.b)
                mp_info->mpie_boost_cnt = port->ip_impcount;    ---(2.c)
                *count = MACH_PORT_INFO_EXT_COUNT;  ---(2.d)
                ip_unlock(port);
                break;
        }

    ...
}
```

```
typedef struct mach_port_info_ext {
        mach_port_status_t      mpie_status;
        mach_port_msgcount_t    mpie_boost_cnt;
        uint32_t                reserved[6];
} mach_port_info_ext_t;
```

MACH_PORT_INFO_EXT_COUNT is 17;

reserved[6] is not filled and disclosed.

This case was fixed internally before my submission so no CVE.

CVE-2019-8529

```
mov     rax, [r15+20h]  ; scalarInput
mov     esi, [rax]      ; int
mov     rdi, r12        ; this
call    __ZN18SCSITaskUserClient20ReleaseTaskReferenceEi ;
jmp     loc_1BC4
```

```
; __int64 __fastcall SCSITaskUserClient::ReleaseTaskReference(SCSITaskUserClient *this, int)
public __ZN18SCSITaskUserClient20ReleaseTaskReferenceEi
__ZN18SCSITaskUserClient20ReleaseTaskReferenceEi proc near
push    rbp
mov     rbp, rsp
push    r15
push    r14
push    r13
push    r12
push    rbx
push    rax
mov     r13d, esi
mov     r15, rdi
movsxd  rbx, r13d
mov     edi, 5275011h   ; unsigned __int64
xor     ecx, ecx        ; unsigned __int64
mov     rsi, r15        ; unsigned __int64
mov     rdx, rbx        ; unsigned __int64
call    __ZL19RecordSTUCTimeStampmmmmm ; RecordSTUCTimeStamp(ulong,ulong,ulong,ulong,ulong)
lea     r14, [r15+rbx*4+170h]  ------(a)
xor     edi, edi
mov     esi, 1
mov     rdx, r14
call    _OSCompareAndSwap
```

Out of boundary read

Acknowledgements

- Syzkaller
- Lighthouse
- My Two Interns: 董正禹、杨朝明 for their help of IOKit patterns and Fuzzing visualization
- Mickey Jin of TrendMicro for his help of KASAN enable

Q & A