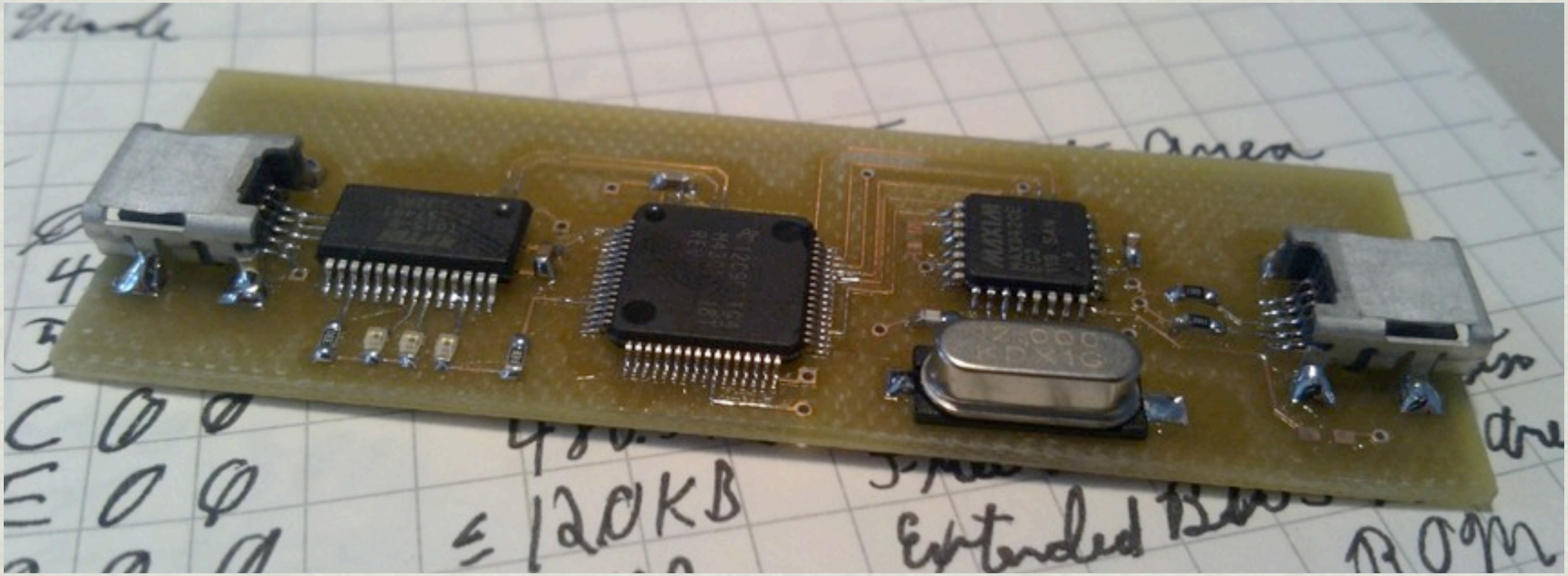
You wouldn't share a syringe. Would you share a USB port?

Travis Goodspeed, Sergey Bratus

Thank you kindly

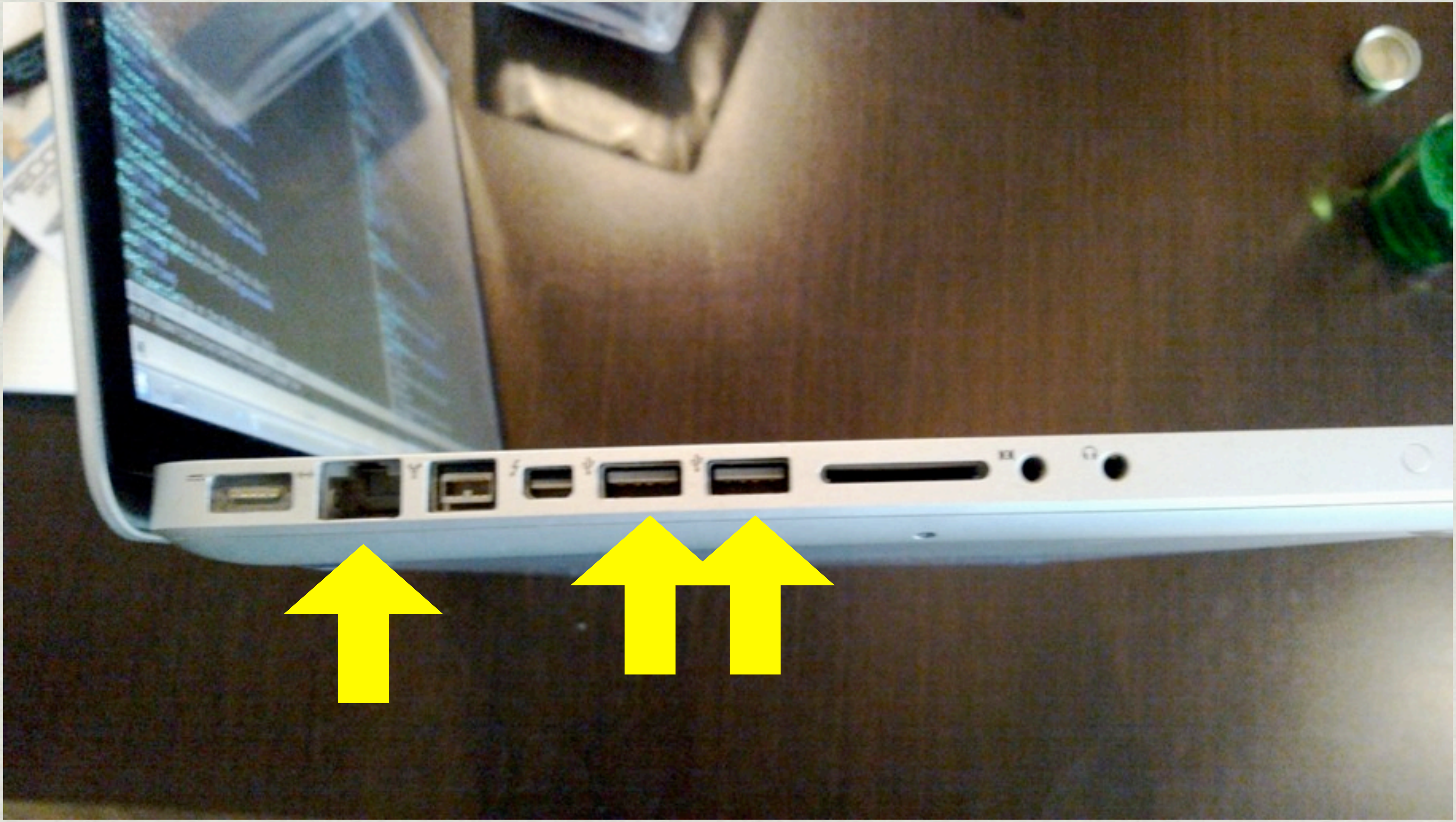
- Searchio
- Dmitry Nedospasov
- Shout-out:
  Andy Davis “50 Lessons learned from USB bugs”
  http://www.nccgroup.com/en/blog/2013/01/lessons-learned-from-50-usb-bugs/

Wright's Law



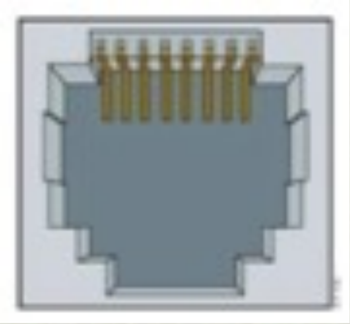
"Security doesn't get better until tools for practical exploration of the attack surface are made available" - Joshua Wright

Which port is scarier?



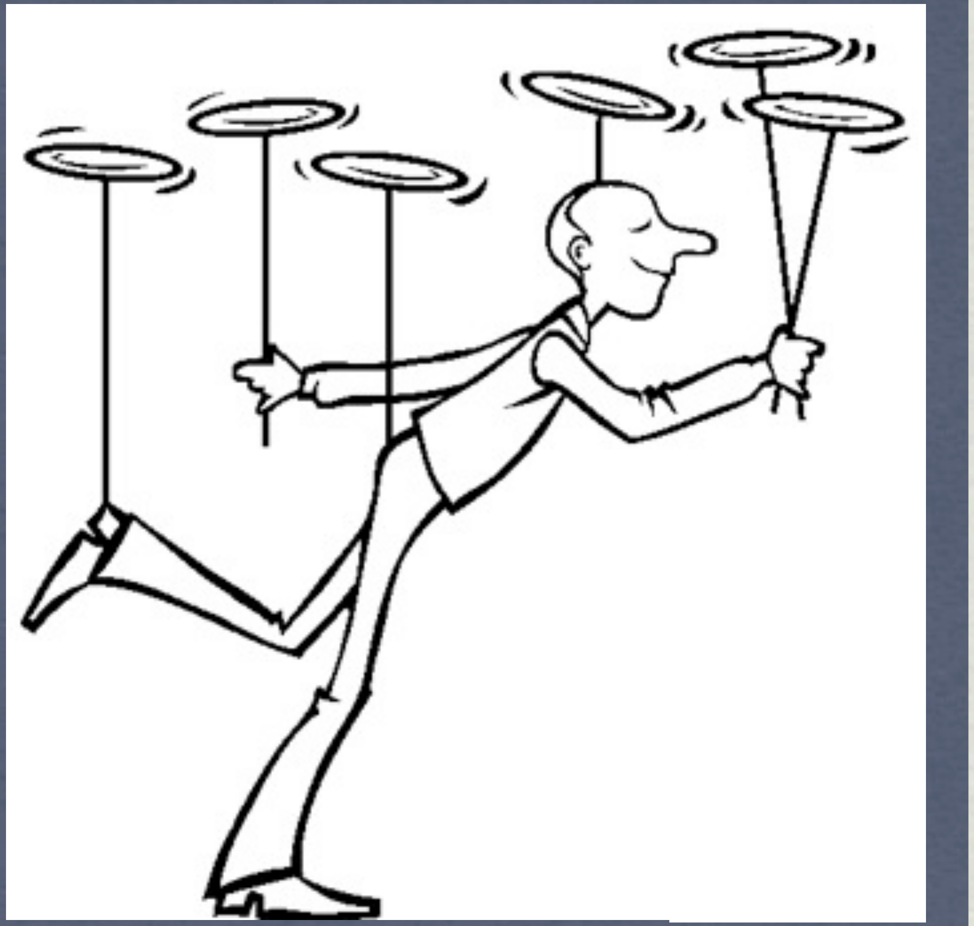
“It’s all a network!”

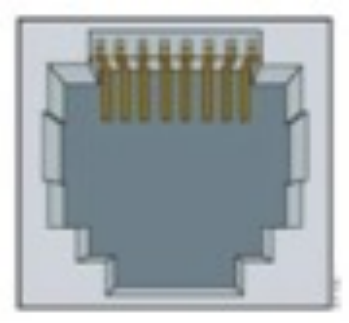
- *Networks:*
  - packets are routed based on data in them
  - have layers of abstraction (OSI)
  - we scan them for vulnerable endpoints
  - we inject crafted packets into them



- *Buses:*
  - well... ***all of the above?***

Which stack is higher?





Wi Fi™
www.THECOLOR.com

More brittle stacks, angrier packets

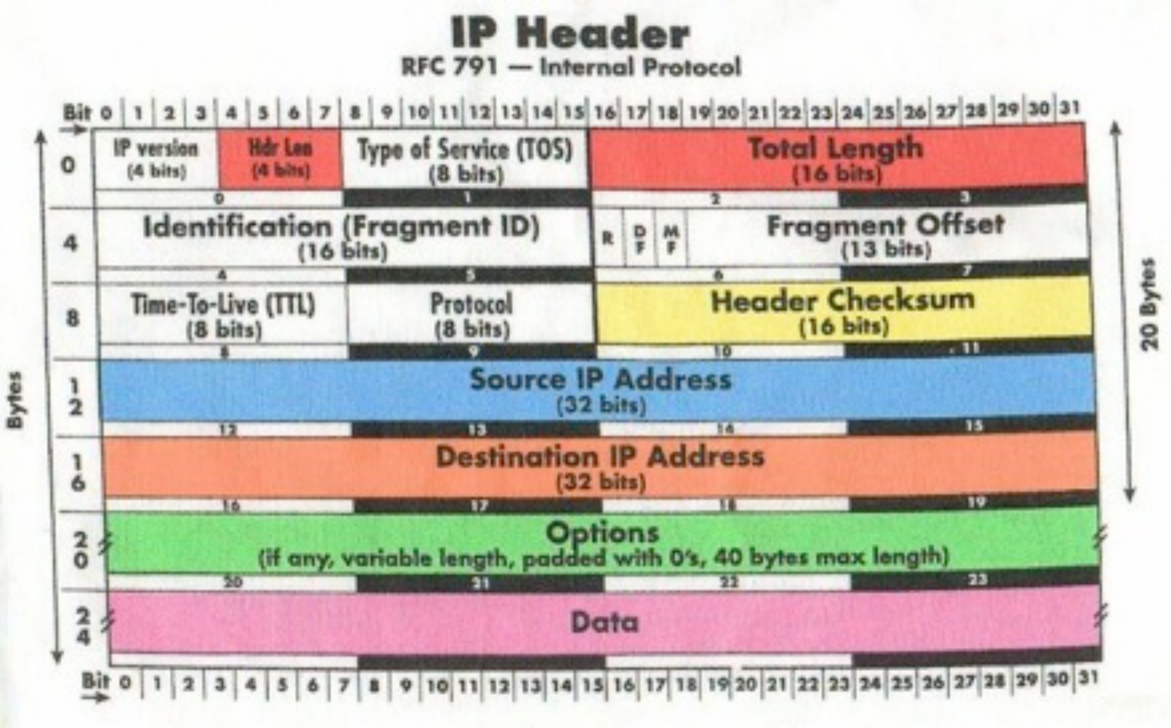
PACKET

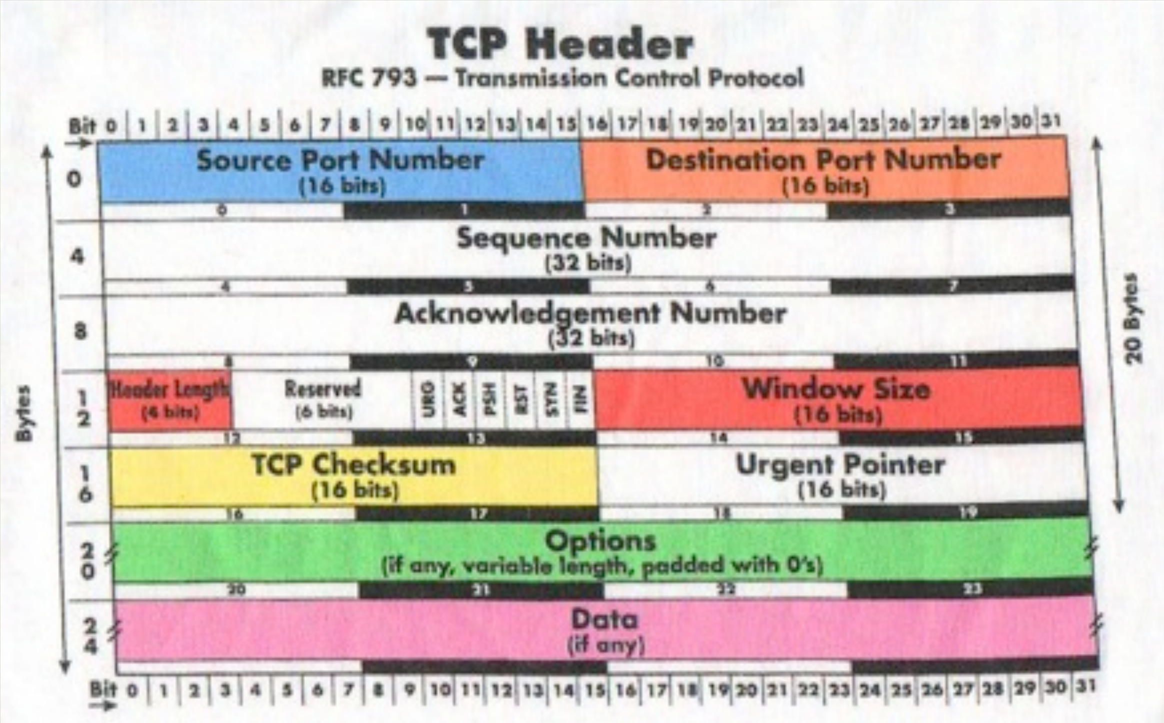
STACK

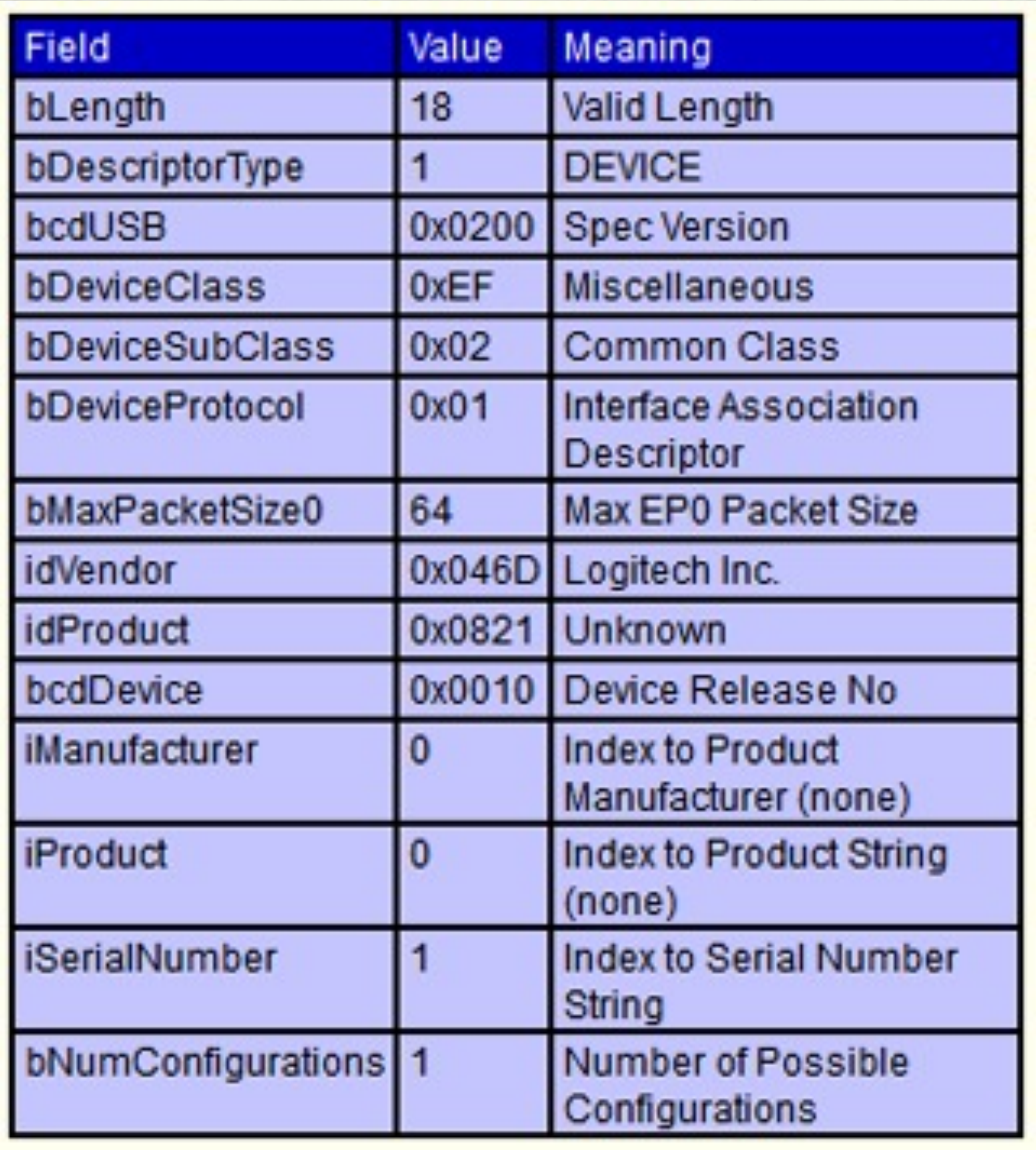
These birds are so damn angry

- Angry birds glorify attackers!
- To improve cyber, we need “Peaceful Pigs Building Solid Defensive Structures”
- Those birds are so damn angry.

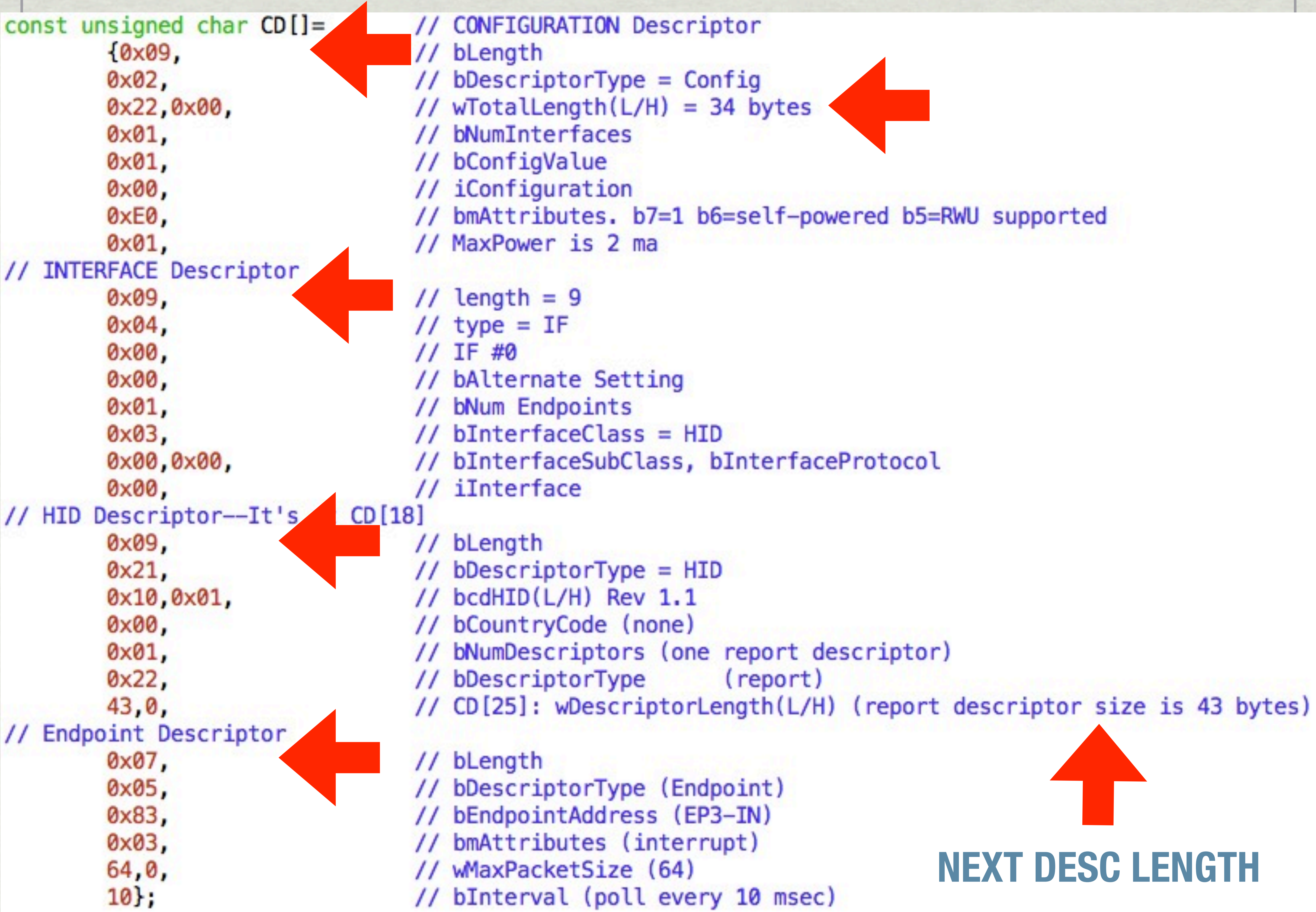
Not your tame TCP/IP birds...


IP Header
RFC 791 — Internal Protocol
Bit 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
IP version (4 bits)
Hdr Len (4 bits)
Type of Service (TOS) (8 bits)
Total Length (16 bits)
Identification (Fragment ID) (16 bits)
R
DF
MF
Fragment Offset (13 bits)
Time-To-Live (TTL) (8 bits)
Protocol (8 bits)
Header Checksum (16 bits)
Source IP Address (32 bits)
Destination IP Address (32 bits)
Options (if any, variable length, padded with 0's, 40 bytes max length)
Data
Bytes
0
4
8
12
16
20
24
20 Bytes


TCP Header
RFC 793 — Transmission Control Protocol
Bit 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Source Port Number (16 bits)
Destination Port Number (16 bits)
Sequence Number (32 bits)
Acknowledgement Number (32 bits)
Header Length (4 bits)
Reserved (6 bits)
URG
ACK
PSH
RST
SYN
FIN
Window Size (16 bits)
TCP Checksum (16 bits)
Urgent Pointer (16 bits)
Options (if any, variable length, padded with 0's)
Data (if any)
Bytes
0
4
8
12
16
20
24
20 Bytes



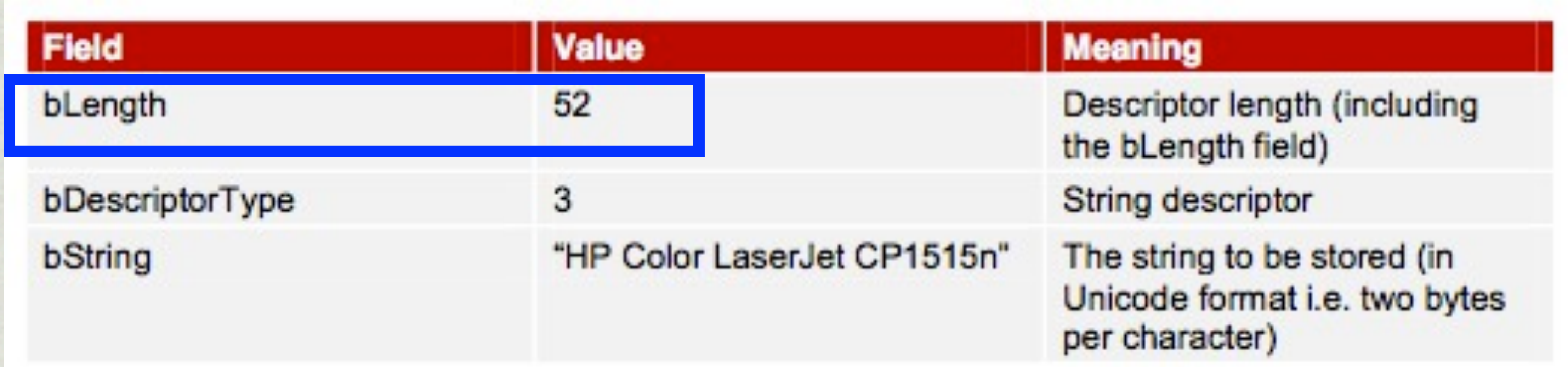
Field	Value	Meaning
bLength	18	Valid Length
bDescriptorType	1	DEVICE
bcdUSB	0x0200	Spec Version
bDeviceClass	0xEF	Miscellaneous
bDeviceSubClass	0x02	Common Class
bDeviceProtocol	0x01	Interface Association Descriptor
bMaxPacketSize0	64	Max EP0 Packet Size
idVendor	0x046D	Logitech Inc.
idProduct	0x0821	Unknown
bcdDevice	0x0010	Device Release No
iManufacturer	0	Index to Product Manufacturer (none)
iProduct	0	Index to Product String (none)
iSerialNumber	1	Index to Serial Number String
bNumConfigurations	1	Number of Possible Configurations



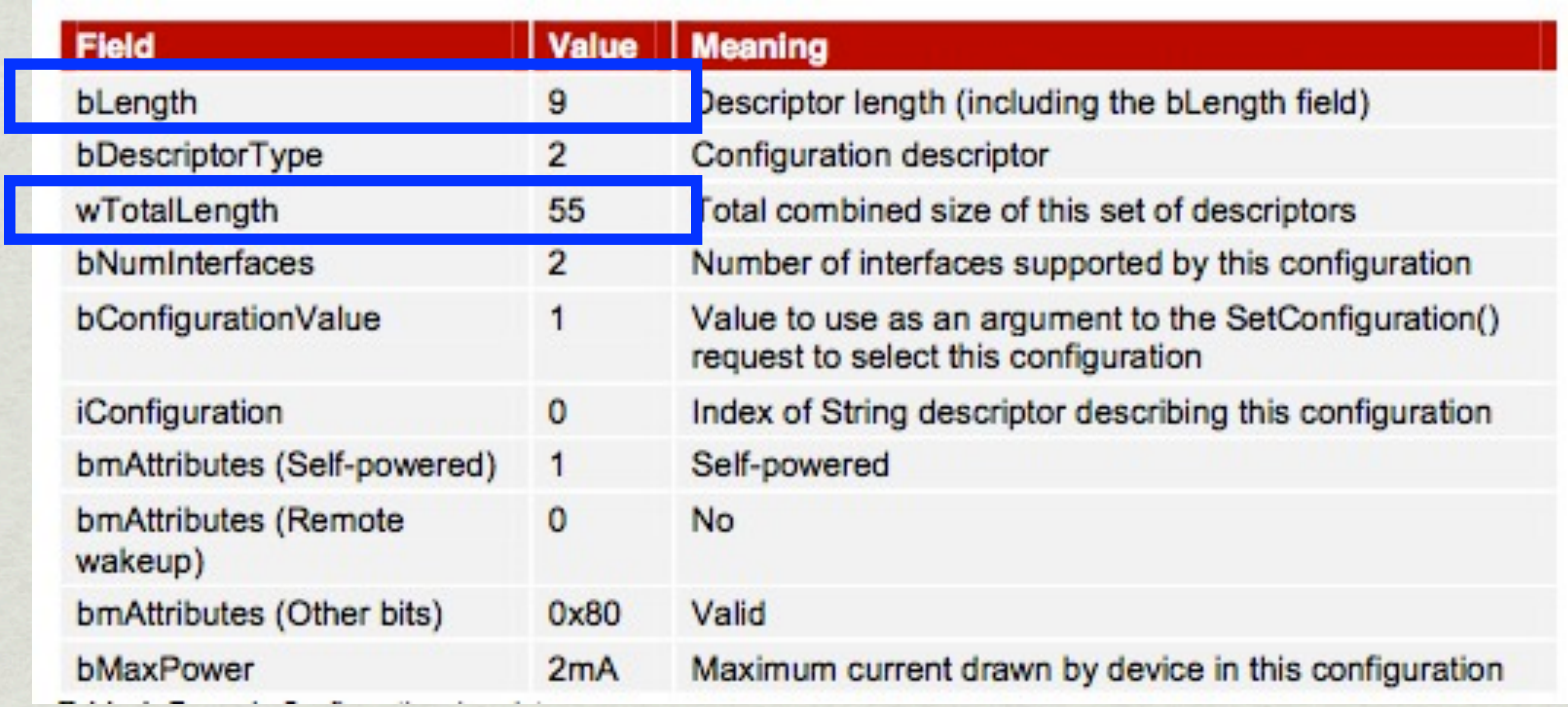
```
const unsigned char CD[]=           // CONFIGURATION Descriptor
        {0x09,                      // bLength
        0x02,                       // bDescriptorType = Config
        0x22,0x00,                  // wTotalLength(L/H) = 34 bytes
        0x01,                       // bNumInterfaces
        0x01,                       // bConfigValue
        0x00,                       // iConfiguration
        0xE0,                       // bmAttributes. b7=1 b6=self-powered b5=RWU supported
        0x01,                       // MaxPower is 2 ma
// INTERFACE Descriptor
        0x09,                       // length = 9
        0x04,                       // type = IF
        0x00,                       // IF #0
        0x00,                       // bAlternate Setting
        0x01,                       // bNum Endpoints
        0x03,                       // bInterfaceClass = HID
        0x00,0x00,                  // bInterfaceSubClass, bInterfaceProtocol
        0x00,                       // iInterface
// HID Descriptor--It's    CD[18]
        0x09,                       // bLength
        0x21,                       // bDescriptorType = HID
        0x10,0x01,                  // bcdHID(L/H) Rev 1.1
        0x00,                       // bCountryCode (none)
        0x01,                       // bNumDescriptors (one report descriptor)
        0x22,                       // bDescriptorType      (report)
        43,0,                       // CD[25]: wDescriptorLength(L/H) (report descriptor size is 43 bytes)
// Endpoint Descriptor
        0x07,                       // bLength
        0x05,                       // bDescriptorType (Endpoint)
        0x83,                       // bEndpointAddress (EP3-IN)
        0x03,                       // bmAttributes (interrupt)
        64,0,                       // wMaxPacketSize (64)
        10};                        // bInterval (poll every 10 msec)
```

NEXT DESC LENGTH

Guess the parser bug



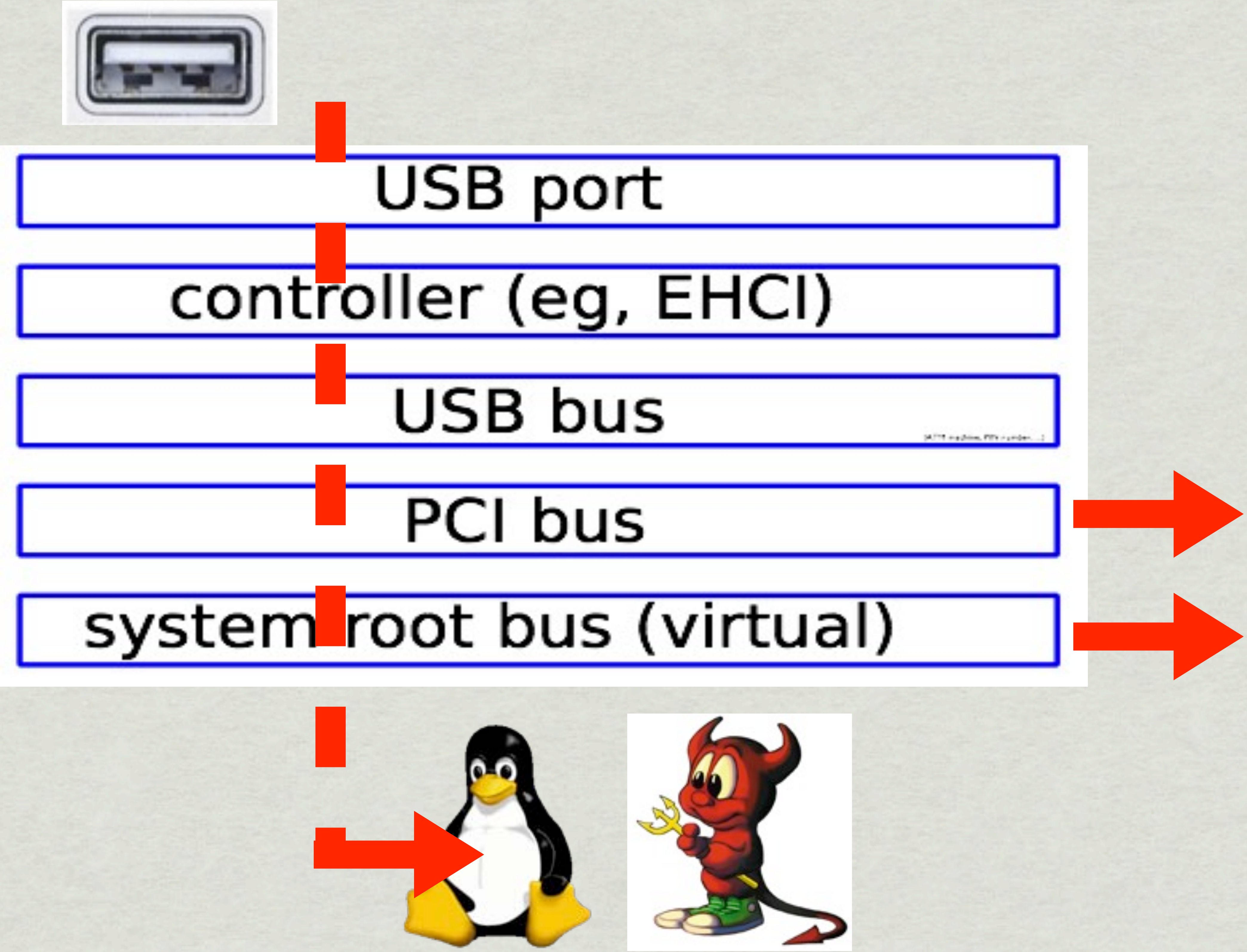
Field	Value	Meaning
bLength	52	Descriptor length (including the bLength field)
bDescriptorType	3	String descriptor
bString	“HP Color LaserJet CP1515n”	The string to be stored (in Unicode format i.e. two bytes per character)



Field	Value	Meaning
bLength	9	Descriptor length (including the bLength field)
bDescriptorType	2	Configuration descriptor
wTotalLength	55	Total combined size of this set of descriptors
bNumInterfaces	2	Number of interfaces supported by this configuration
bConfigurationValue	1	Value to use as an argument to the SetConfiguration() request to select this configuration
iConfiguration	0	Index of String descriptor describing this configuration
bmAttributes (Self-powered)	1	Self-powered
bmAttributes (Remote wakeup)	0	No
bmAttributes (Other bits)	0x80	Valid
bMaxPower	2mA	Maximum current drawn by device in this configuration

ANDY DAVIS ’50 BUGS’

What’s behind a USB port?



USB port

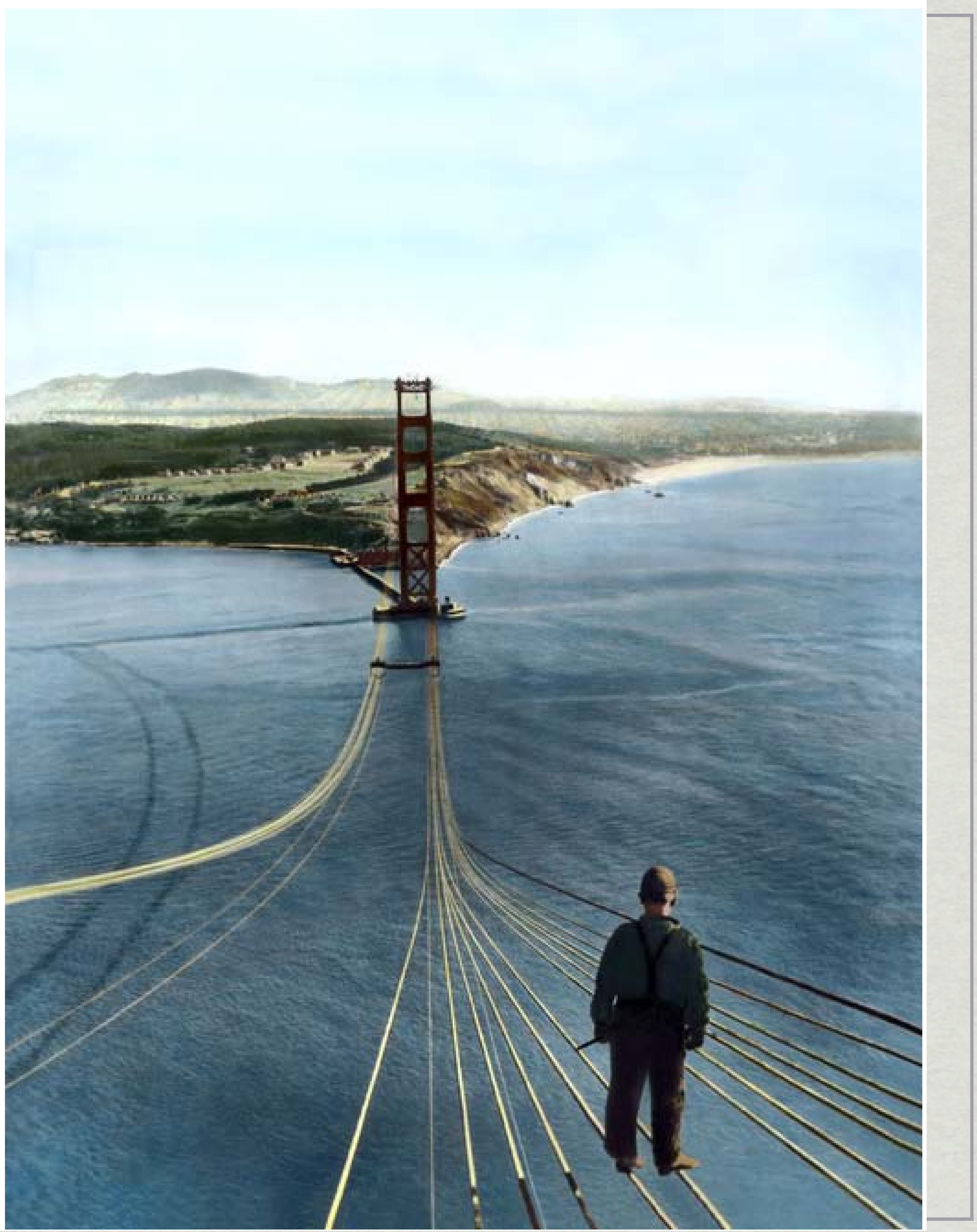
controller (eg, EHCI)

USB bus

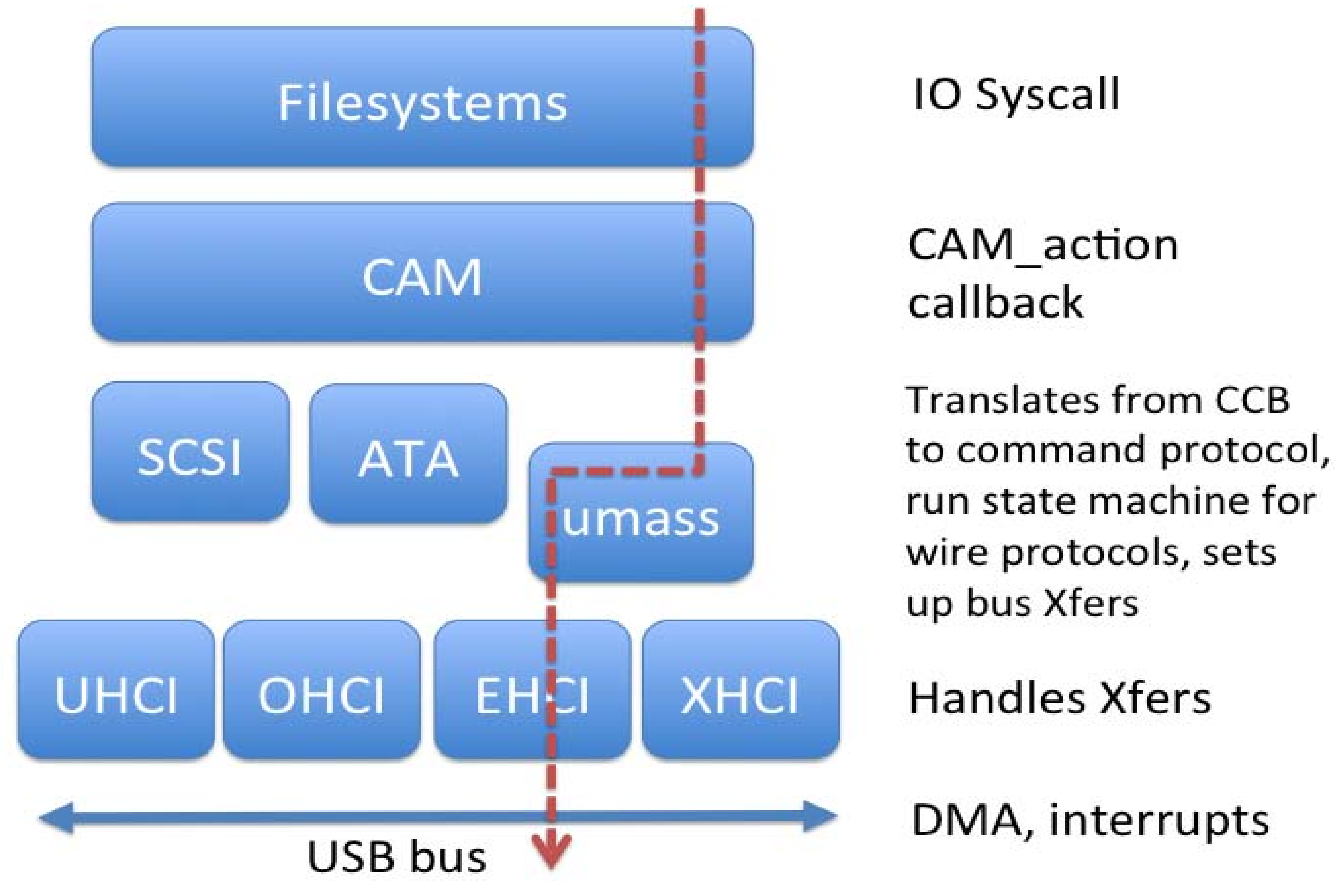
PCI bus

system root bus (virtual)

A lot hangs on these wires



System programmer view



IO Syscall

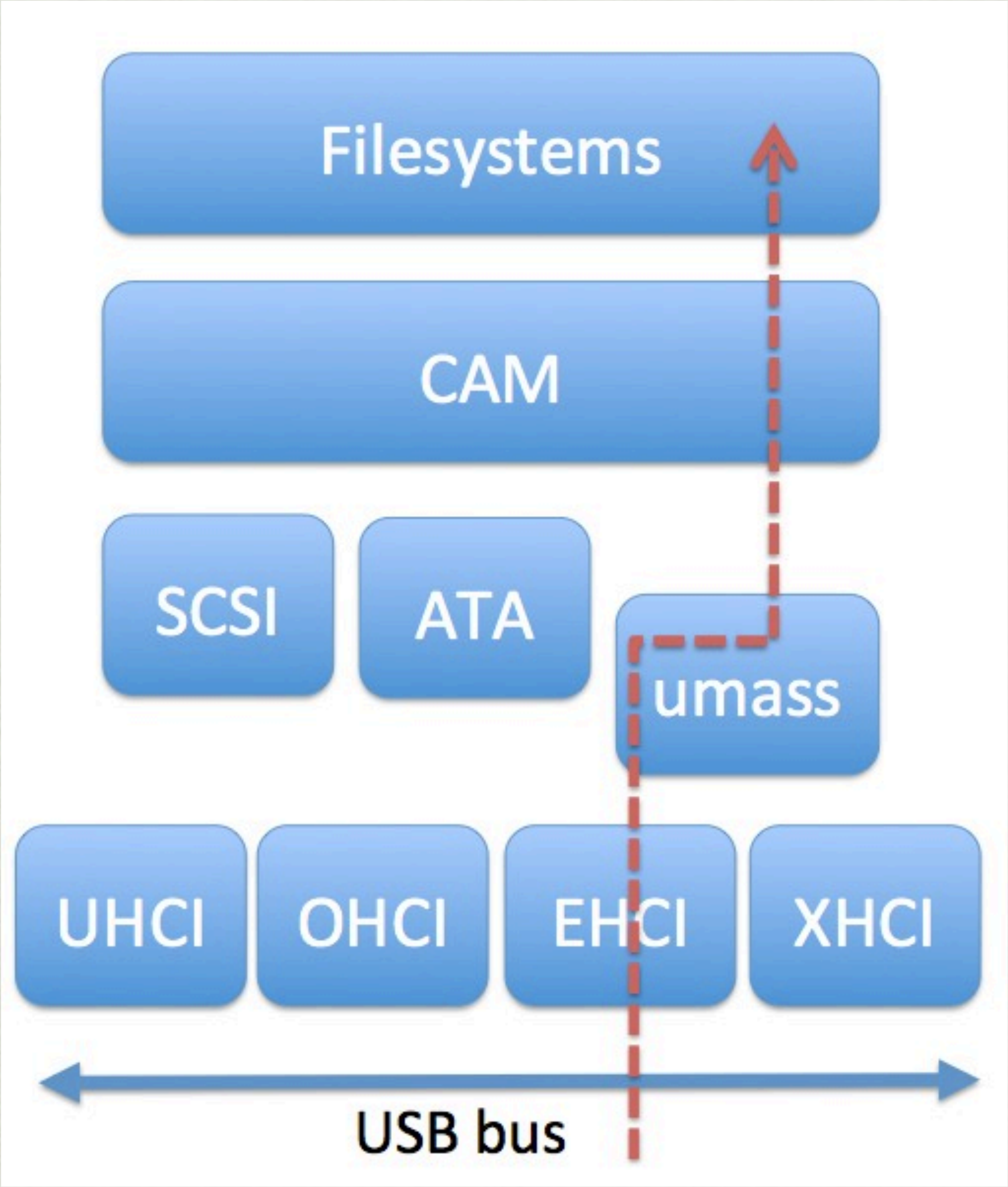
CAM_action callback

Translates from CCB to command protocol, run state machine for wire protocols, sets up bus Xfers

Handles Xfers

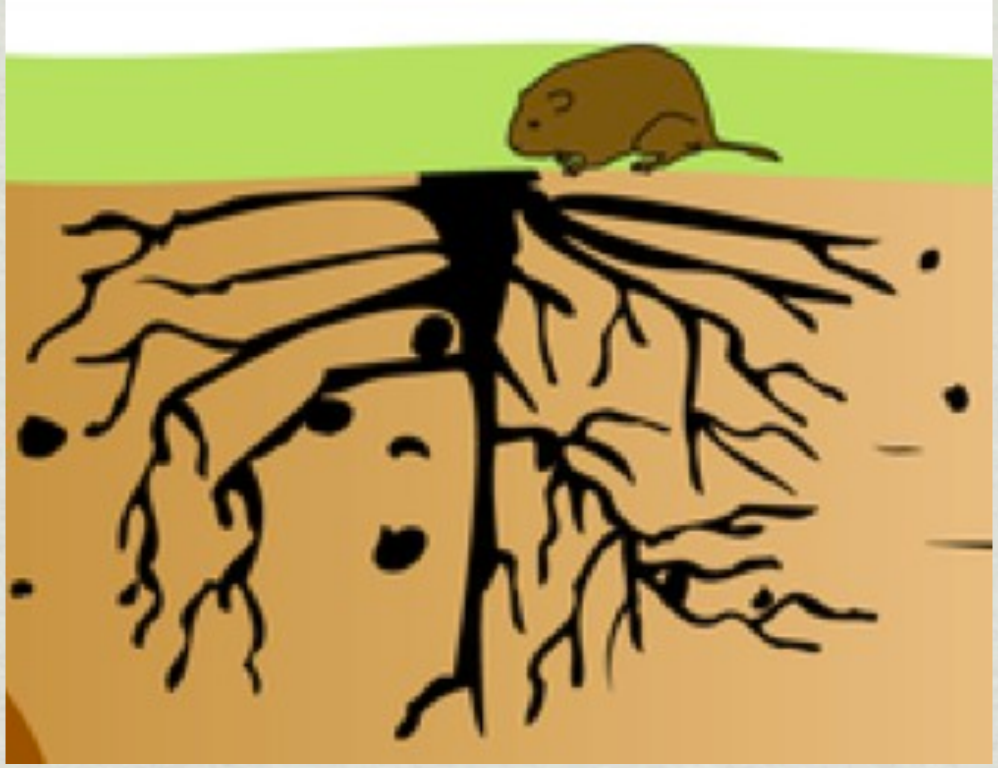
DMA, interrupts

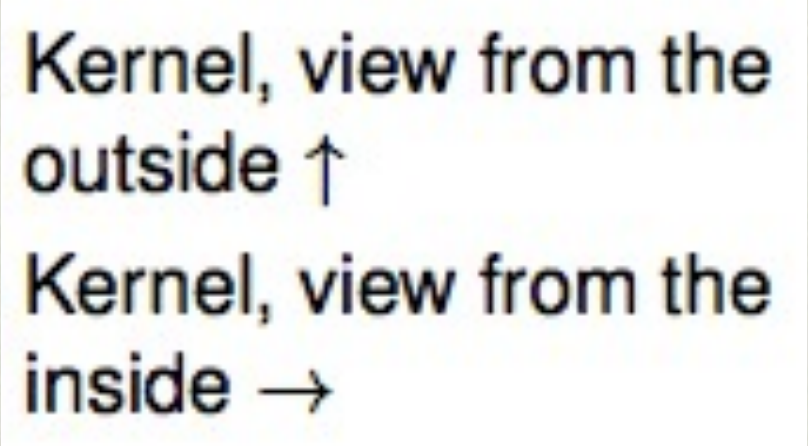
Port-side view



- All kinds of subsystems and drivers are reachable from USB
- “Sanity checks” are haphazard; data is trusted
- “Go anywhere in the kernel”

Through the port, down the rabbit hole





Kernel, view from the outside ↑

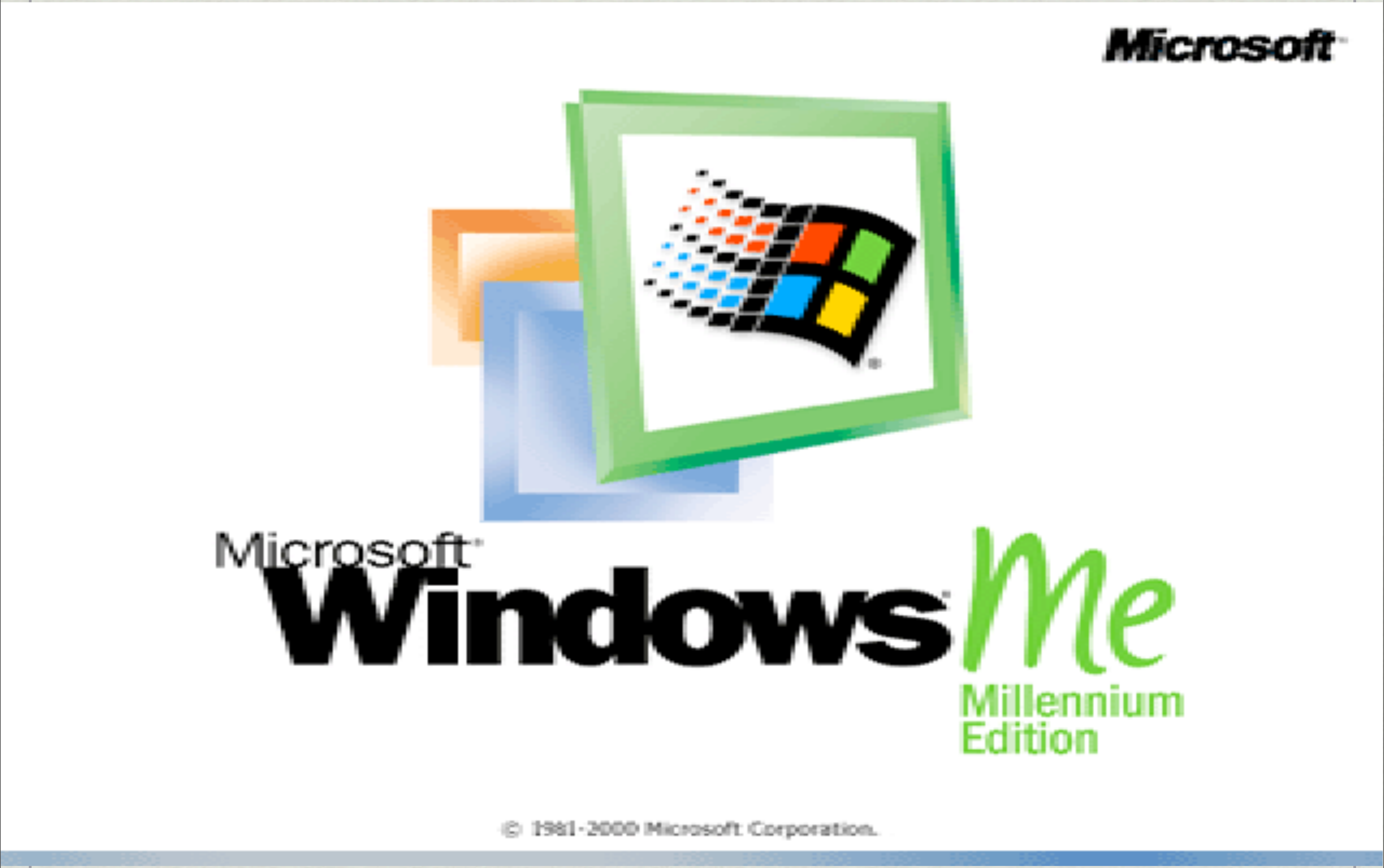
Kernel, view from the inside →

Are you firewalling this?

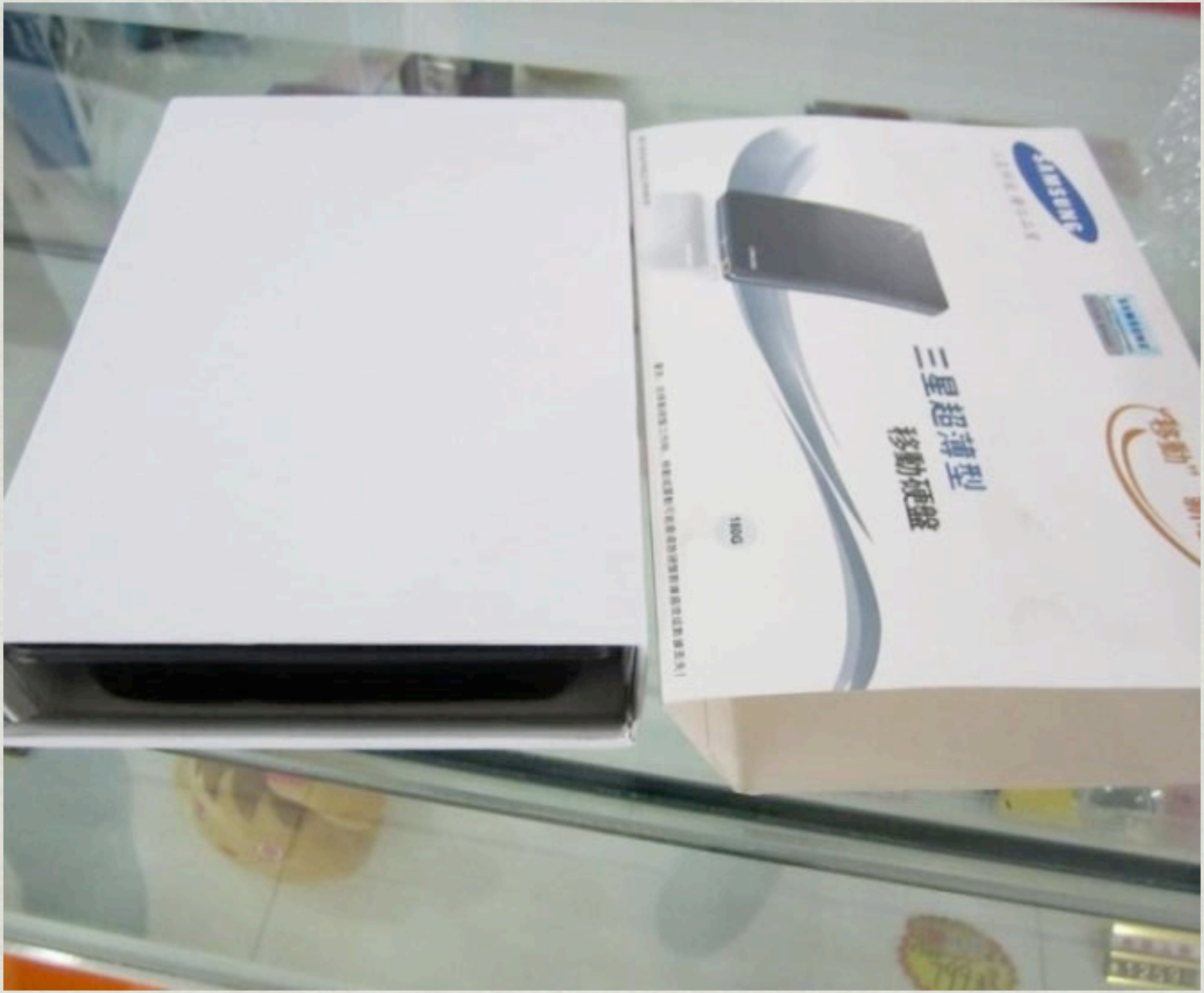
- More targets
- Richer data structures
- Looser code
- Higher privilege (Kernel/Ring0 until recent userland USB stacks)

“I see dead drivers”

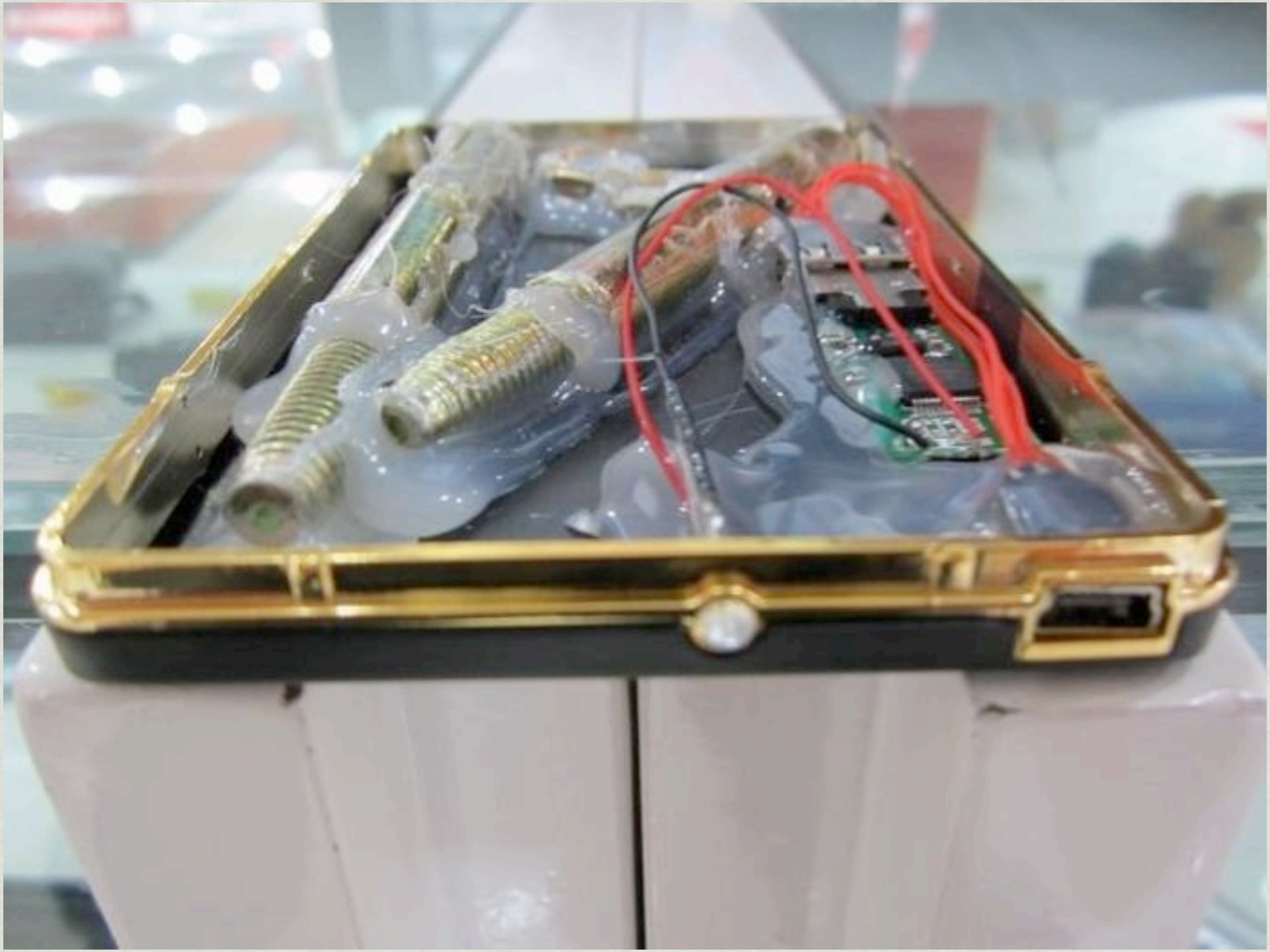
- 1999, conforms to no standards
- Ubuntu includes drivers
- “Works great with Windows ME!”


Microsoft
Microsoft
Windows
Me
Millennium Edition
© 1981-2000 Microsoft Corporation.

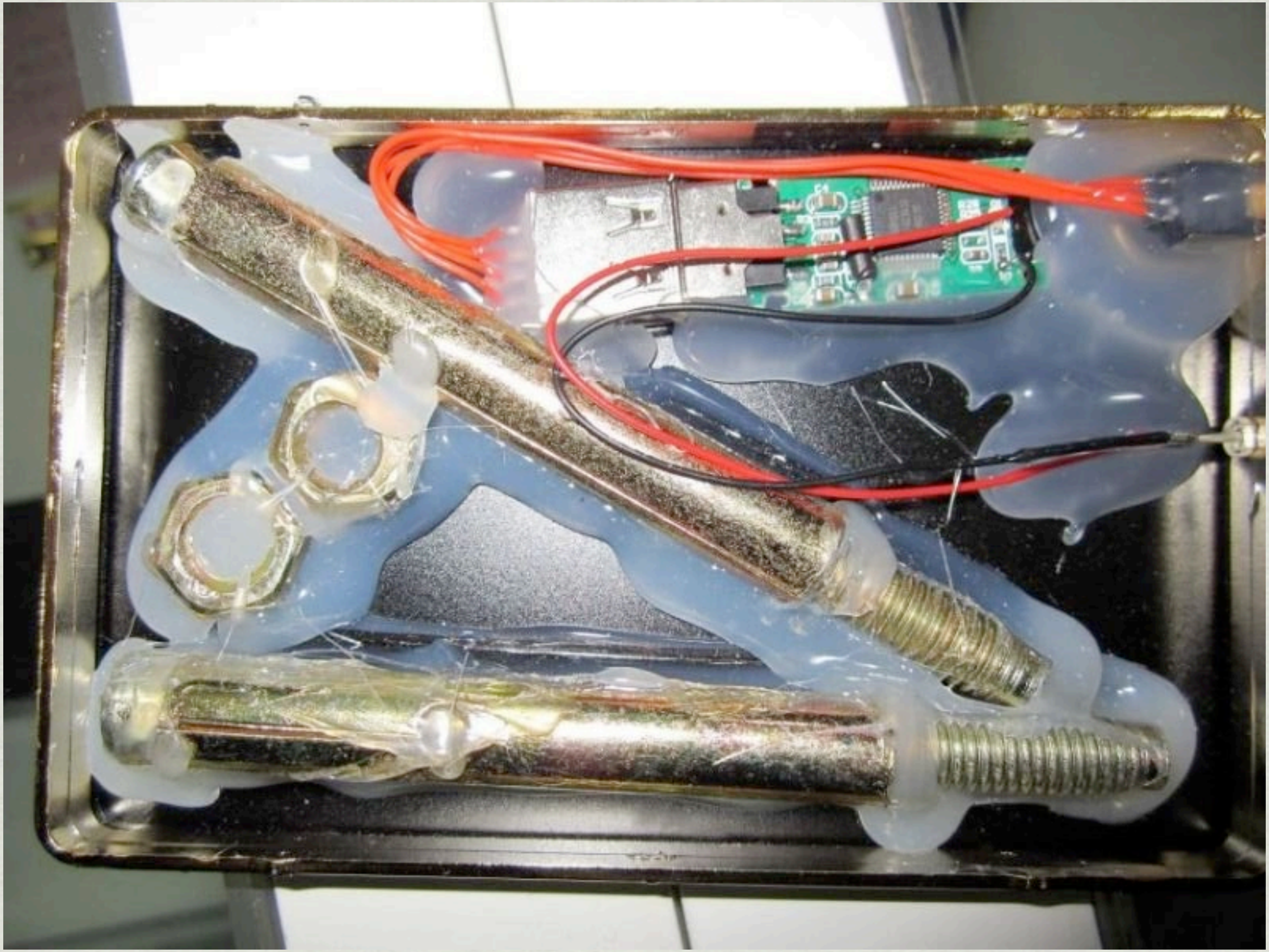
“APT”



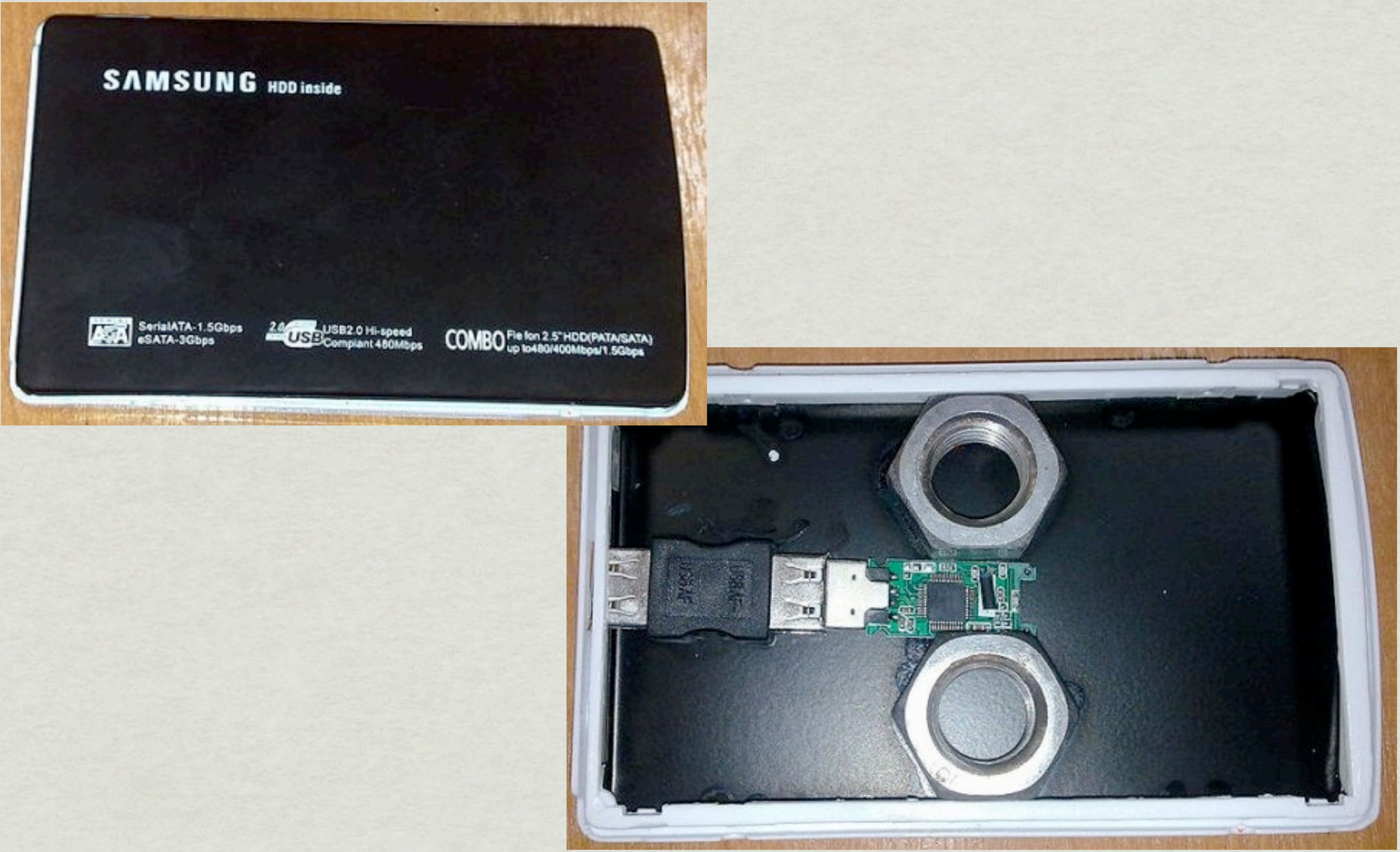
“APT”



“APT”



“APT”

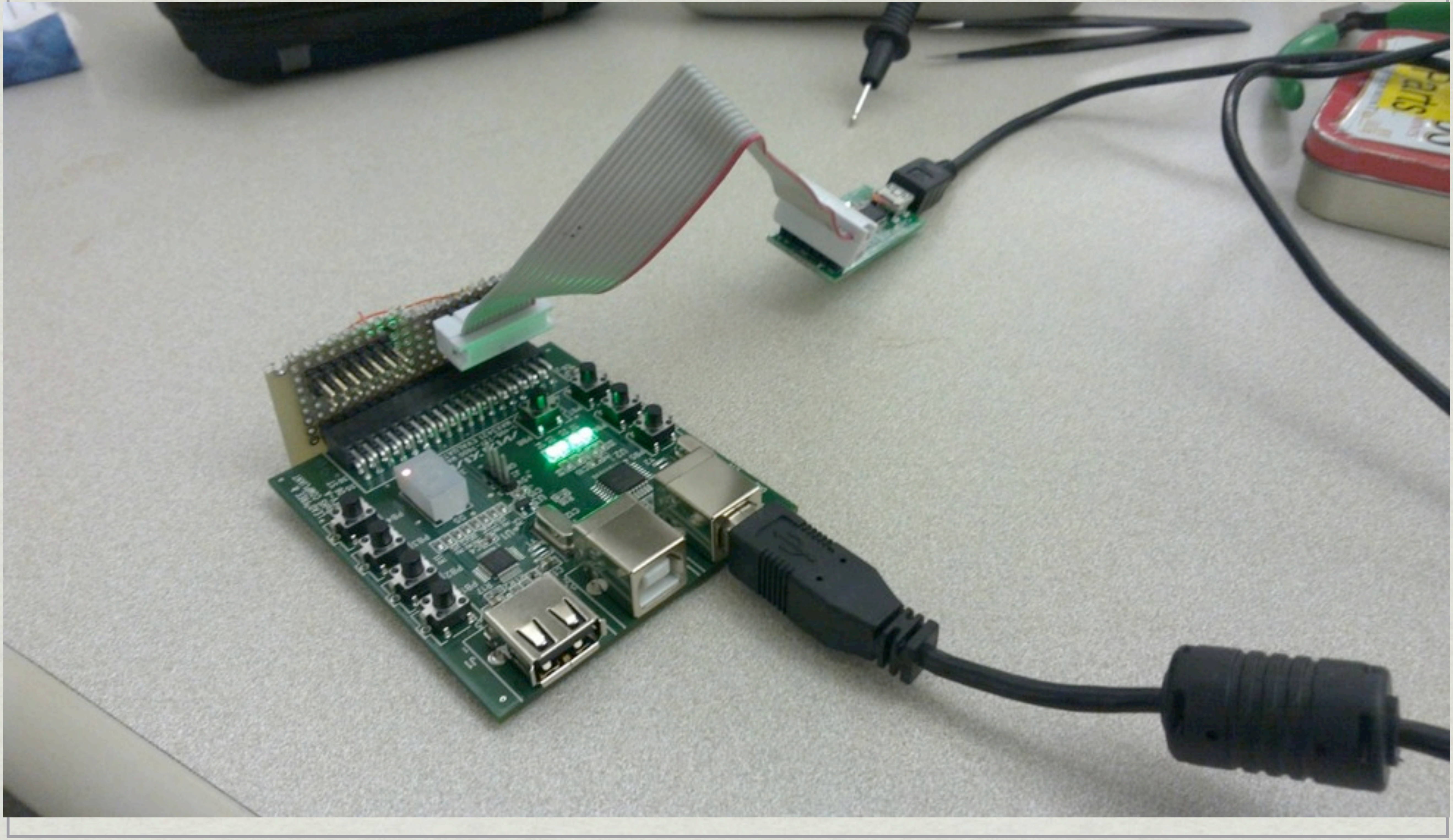

SAMSUNG HDD inside
SerialATA-1.5Gbps
eSATA-3Gbps
2.0 USB
USB2.0 Hi-speed
Compiant 480Mbps
COMBO
Fie fon 2.5" HDD(PATA/SATA)
up to480/400Mbps/1.5Gbps

Why aren’t we firewalling that, again?

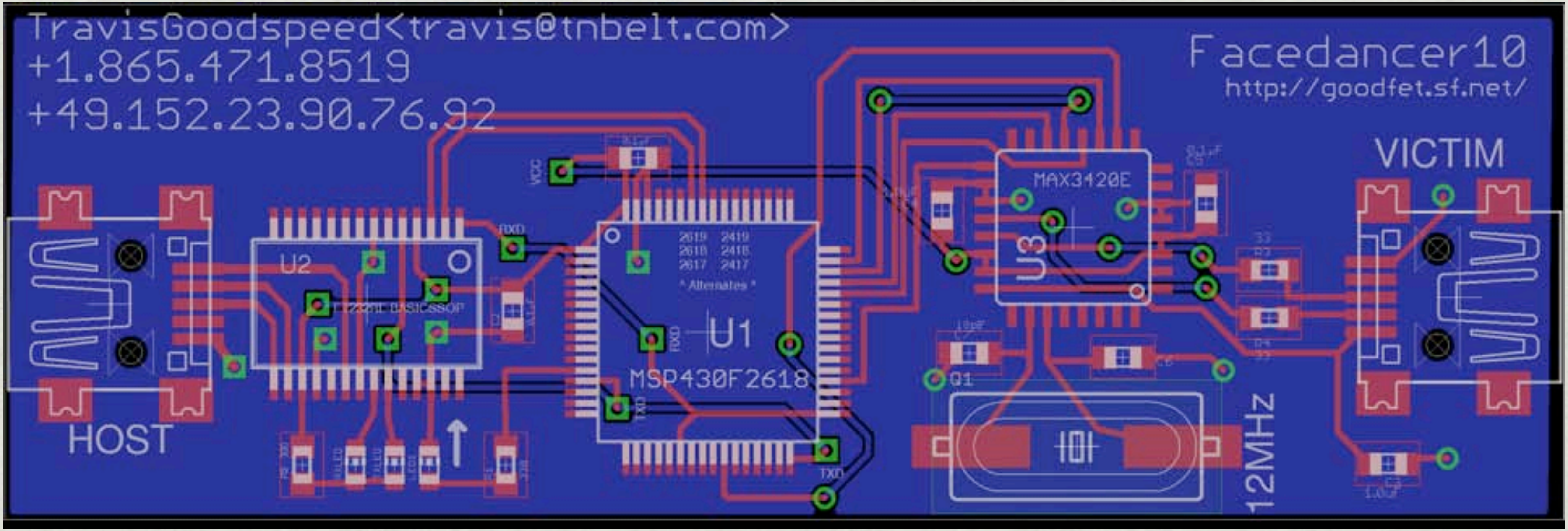
- Payload delivered over USB can pick any target in the kernel - it will **pick** & **choose** the **loosest** code
  - “Sloppy webcam 0.1” driver?
- How easy it is to firewall all the “bad” commands across SCSI, ATAPI, ...?
  - **s/**Application Firewalls**/**Driver Firewalls**/g**
  - ...
  - Profit!

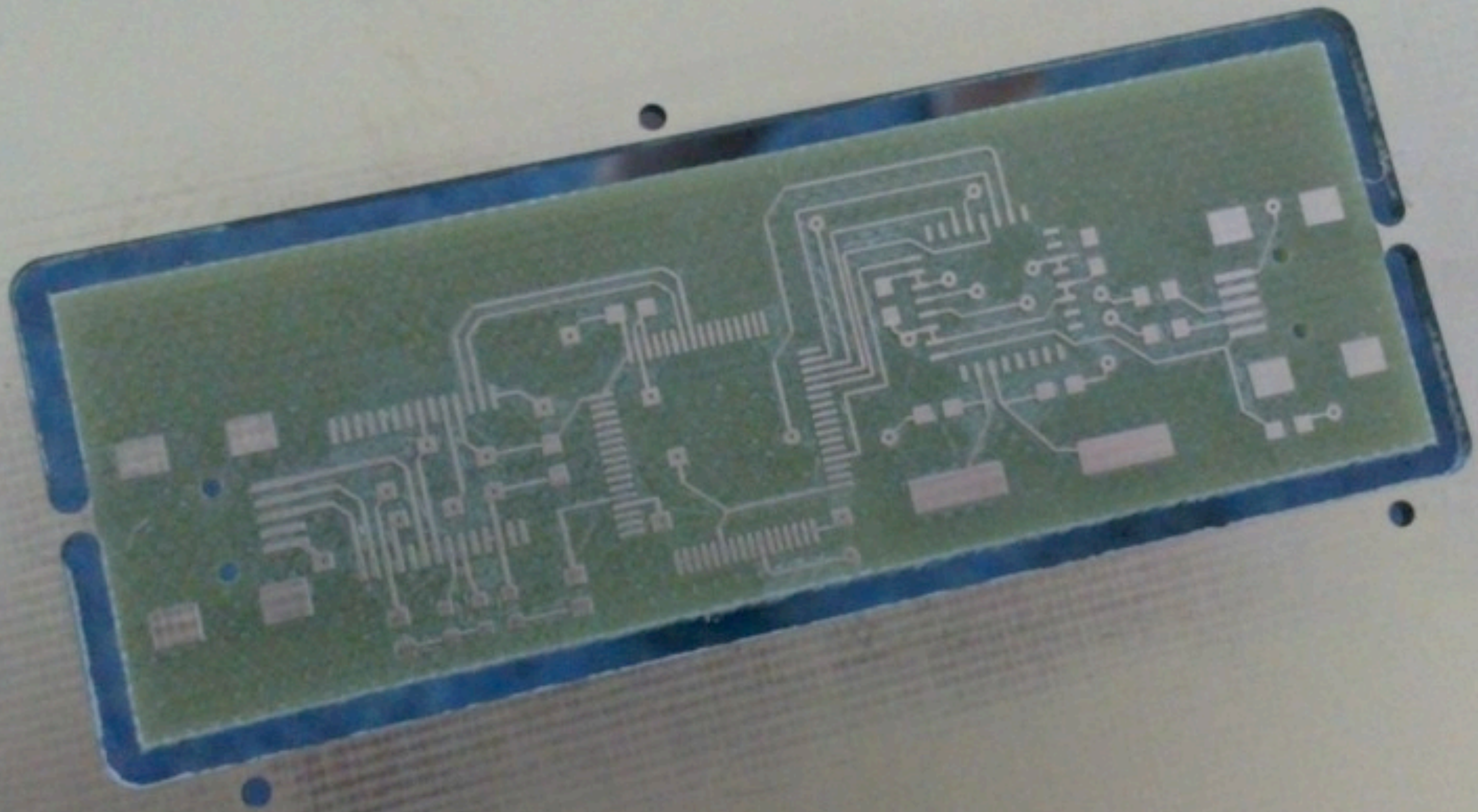
USB	Ethernet	Assumption	Violation	Attack Use
Transfer	One round-trip, maybe NAK-ed	**Intended** device will reply to the transfer	Non-compliant controller	Hijack session, change state under the feet of the host
Transaction	One set of transfers, all but the last NAK-ed	Host controller complies with the USB spec on transactions	Hijack session on disconnect	Use of trusted session context
Packet	Packet Fragment	**Implicit** length of concatenated frames will match **explicit** length of transaction	Non-compliant device	Memory corruption, info leak
Controller	Ethernet Card	—	—	—
Bus	D+/D- Pair	Electrically legal signals, but in reality those **widely outside** of spec are accepted	Non-compliant controller	Damage frames for session hijack, jamming

Same-day prototype:

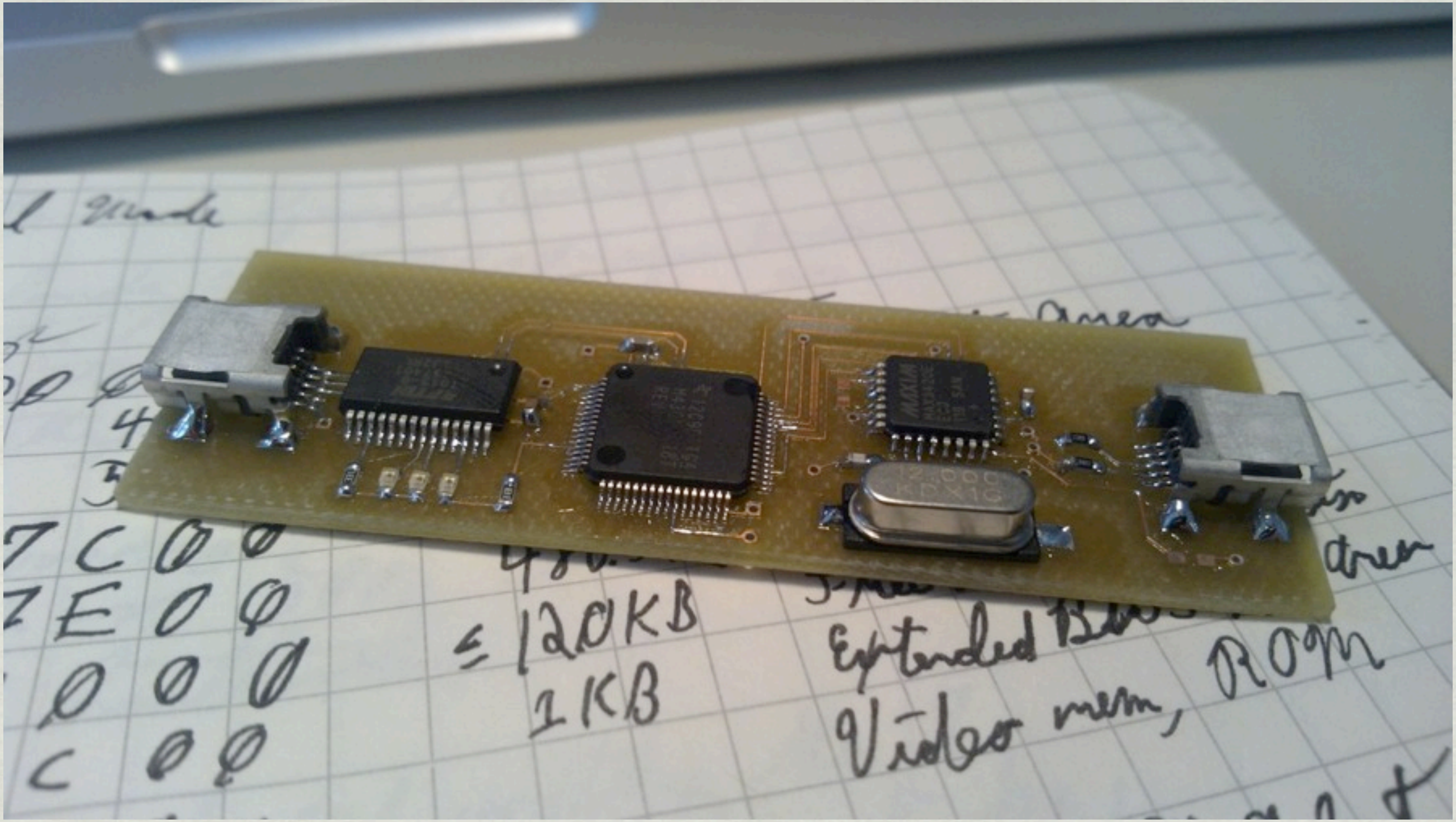


Custom PCB

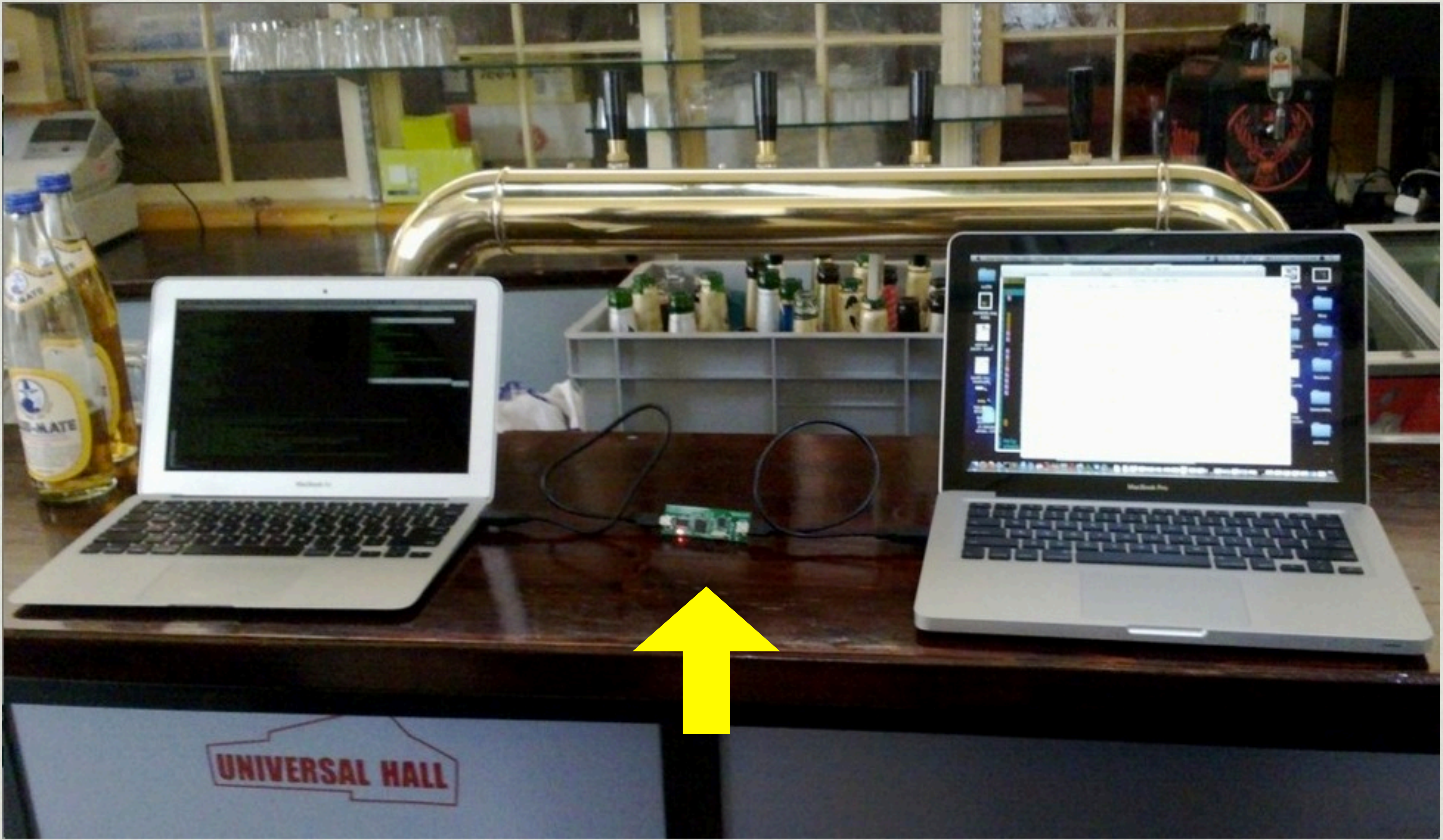

TravisGoodspeed<travis@tnbelt.com>
+1.865.471.8519
+49.152.23.90.76.92
Facedancer10
http://goodfet.sf.net/
VICTIM
HOST
U2
FT232RL BASICSSOP
VCC
RXD
2619 2419
2618 2418
2617 2417
* Alternates *
U1
RXD
TXD
MSP430F2618
TXD
MAX3420E
U3
Q1
12MHz



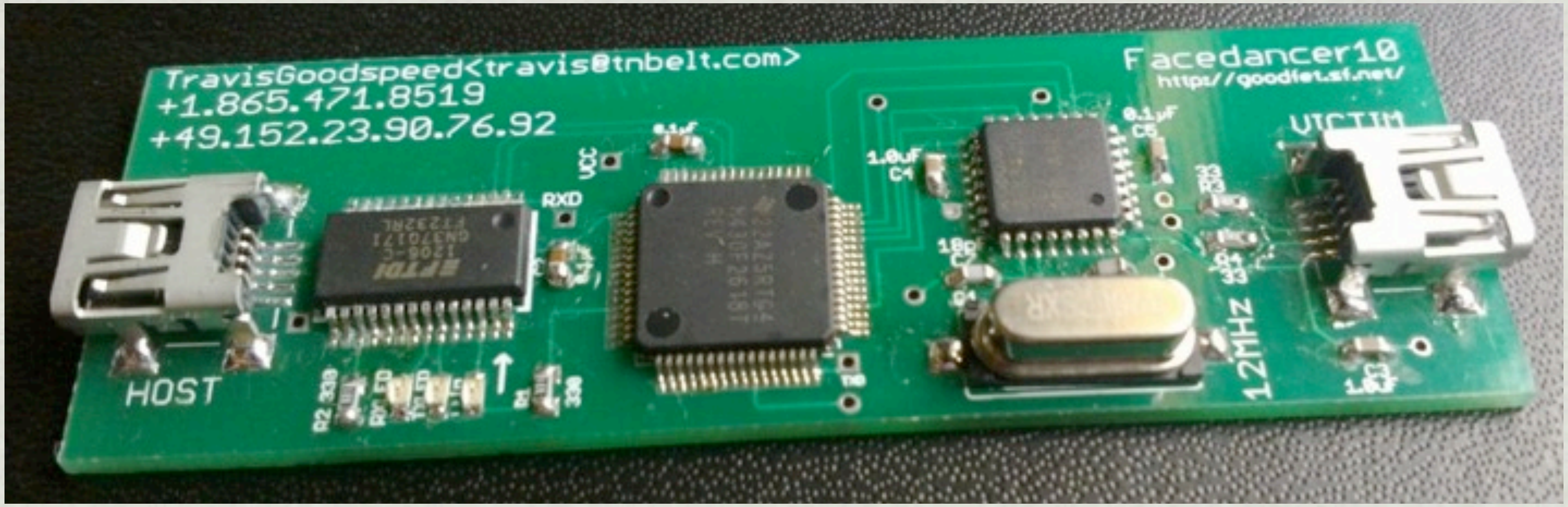
Facedancer 0.1

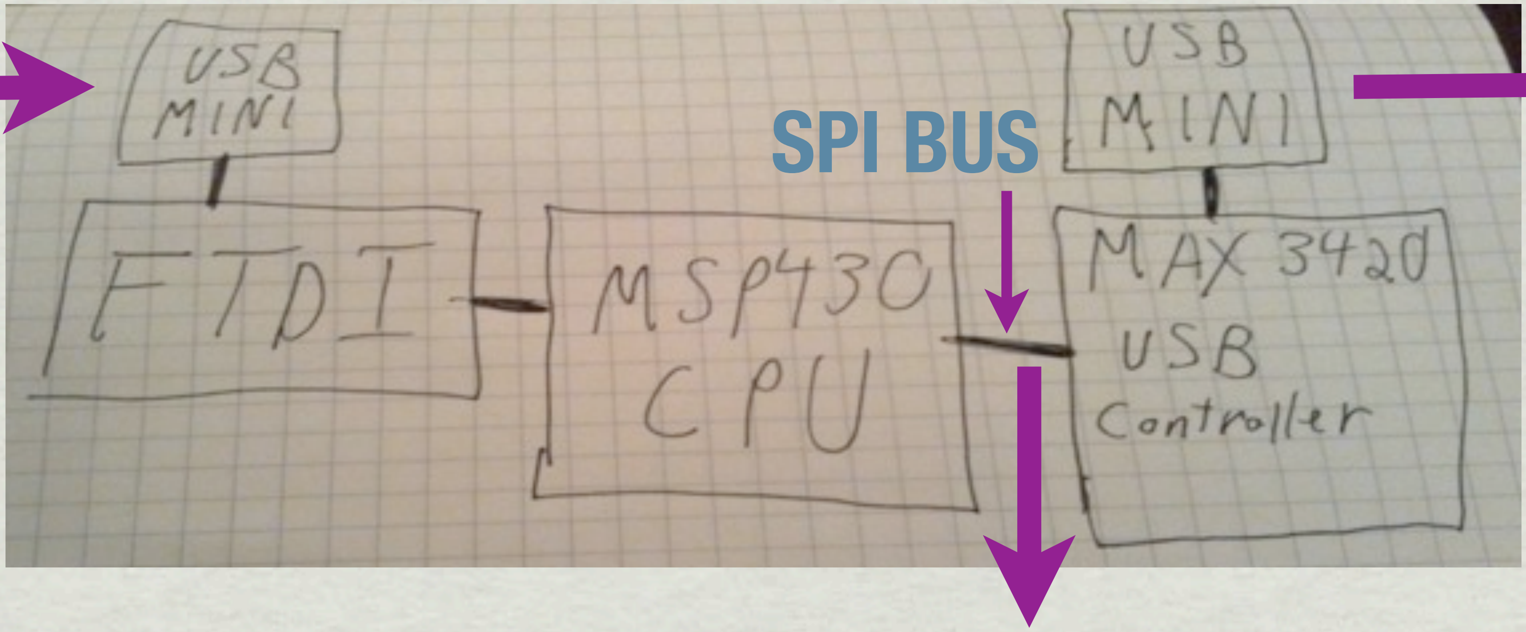


Let’s network them!



The Router/Injector/Facedancer



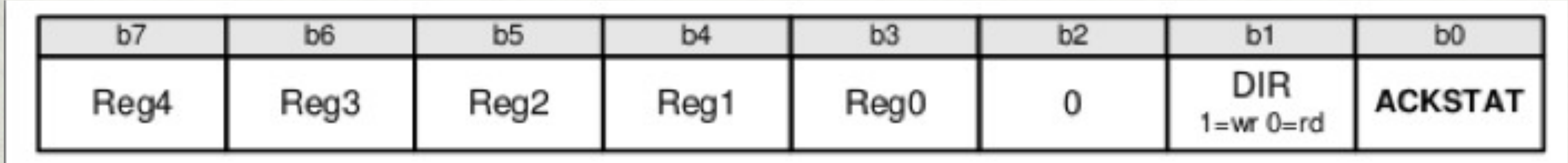


FROM HOST, RAW PACKET (IN PYTHON)

SPI BUS

USB TO VICTIM

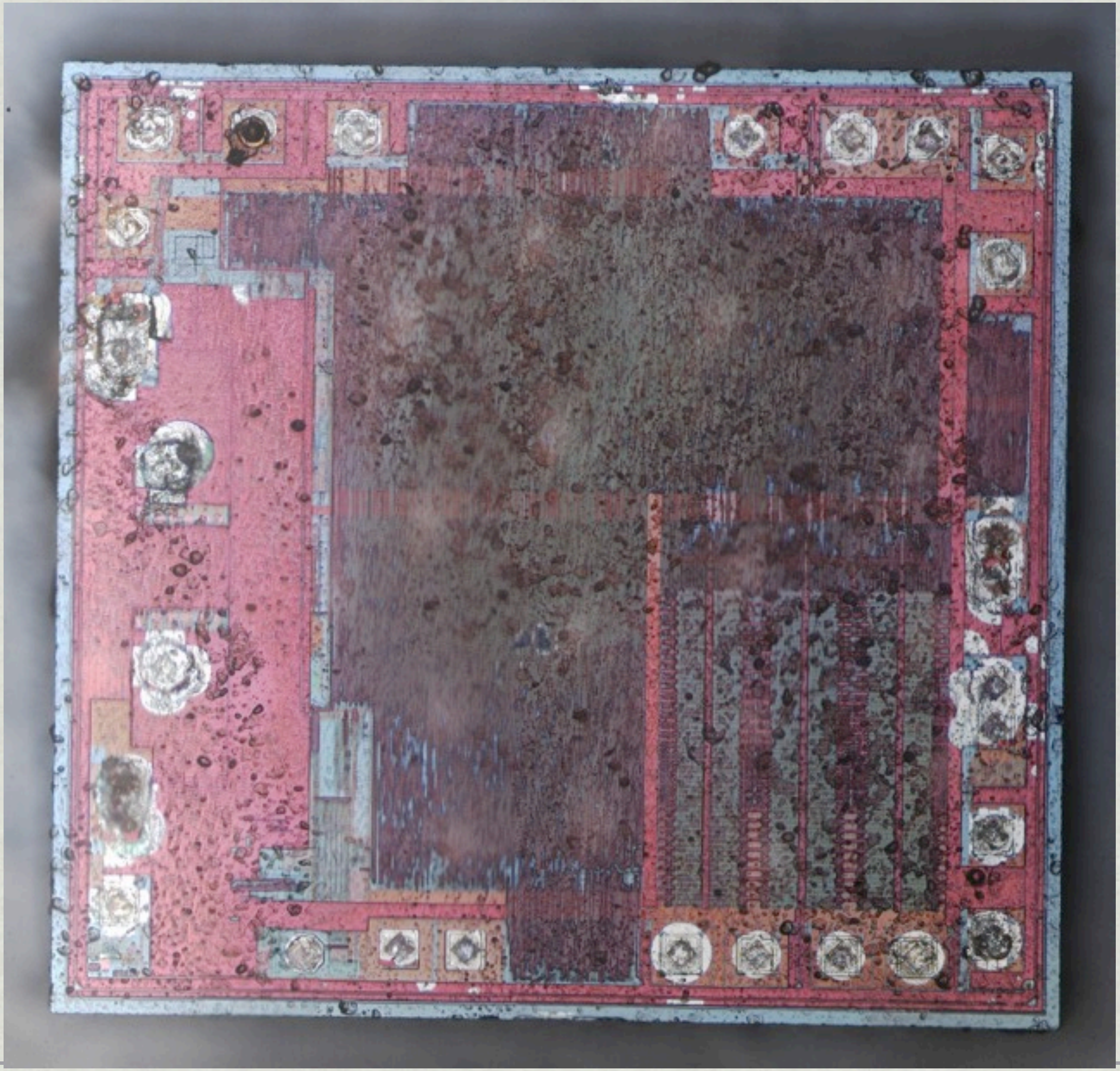
"SEND BUFFER NOW"

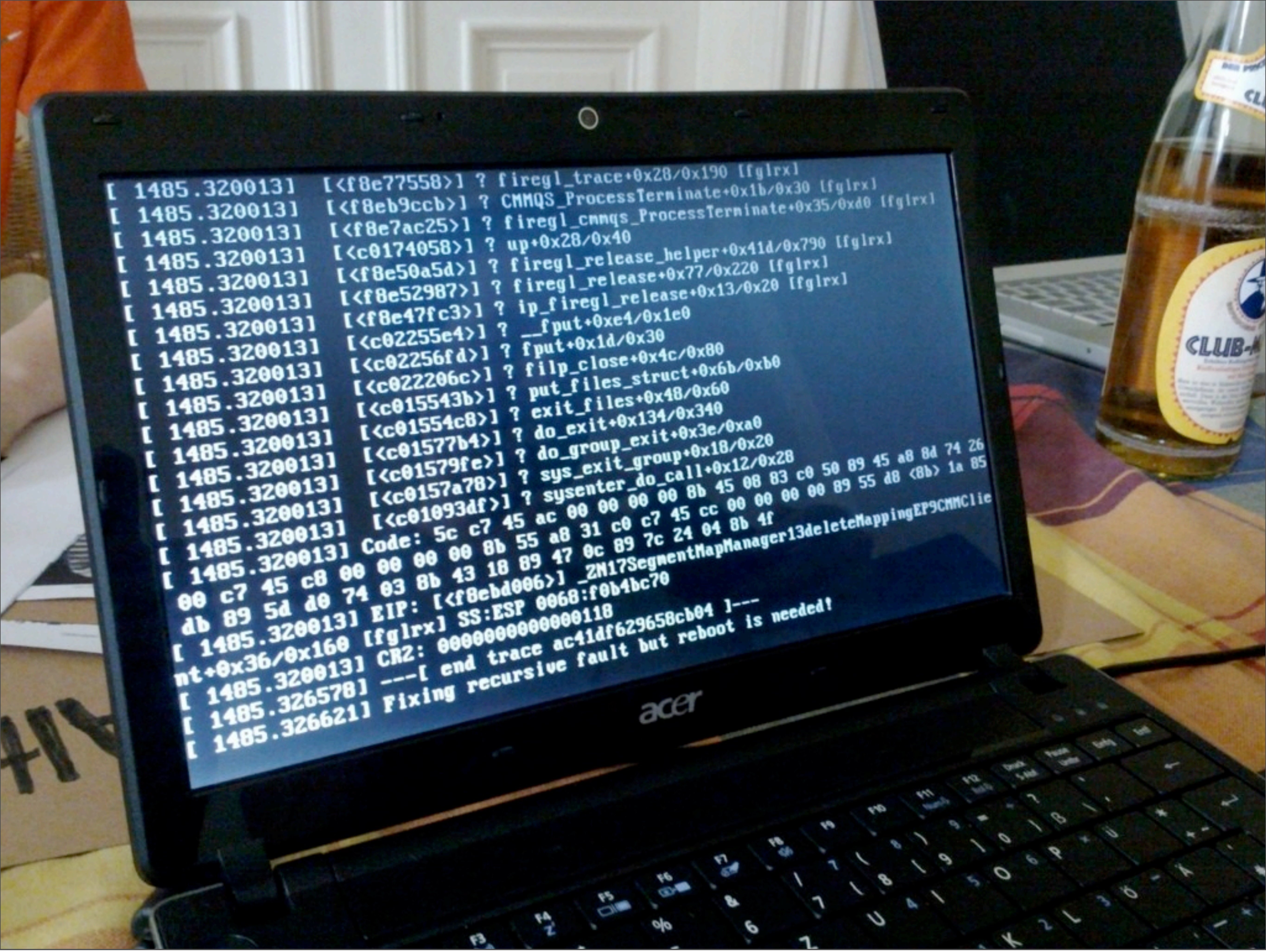


b7	b6	b5	b4	b3	b2	b1	b0
Reg4	Reg3	Reg2	Reg1	Reg0	0	DIR 1=wr 0=rd	ACKSTAT

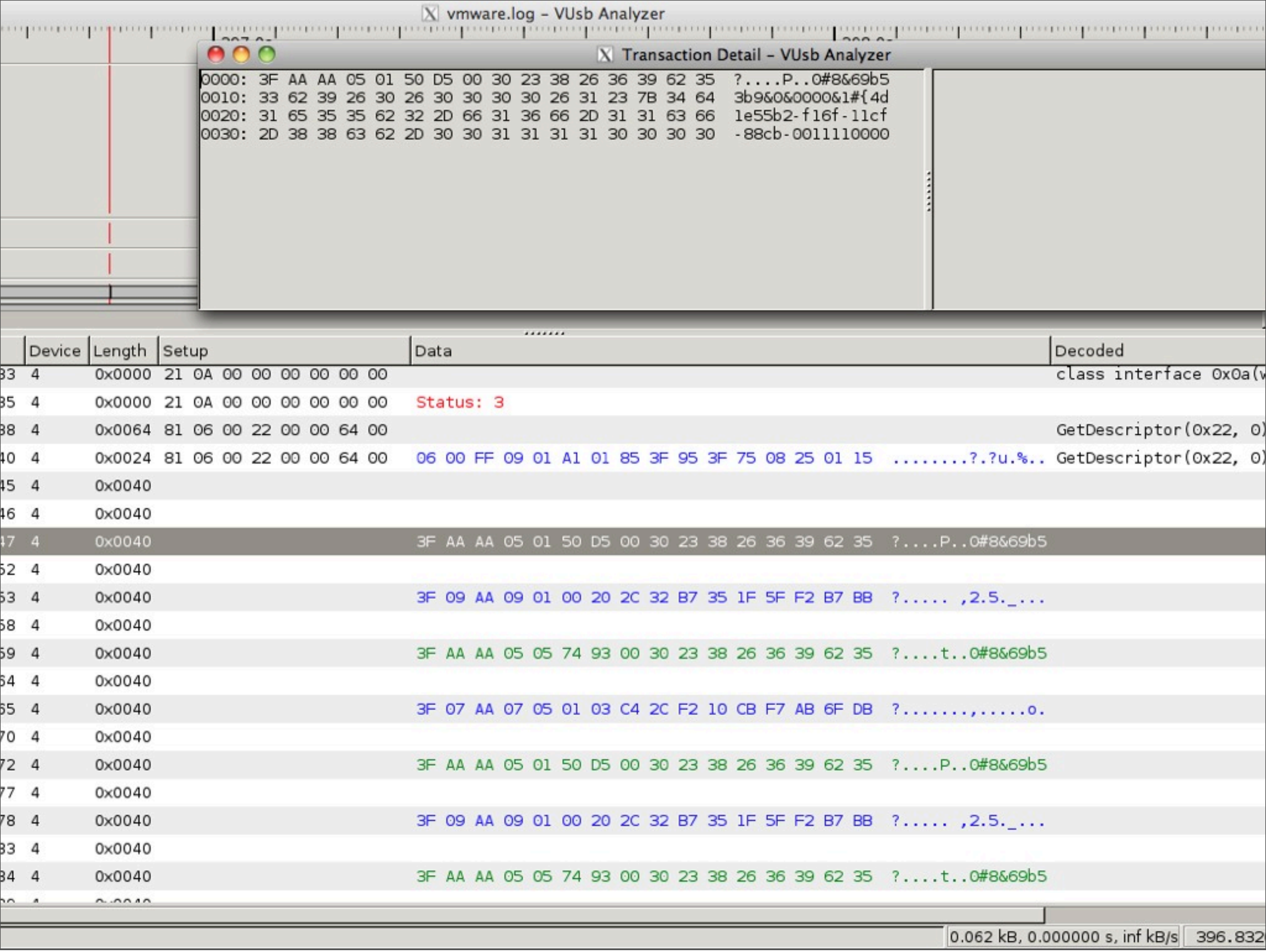
Figure 8. The MAX3420E SPI command byte.

Maxim MAX3420E





```
[ 1485.320013]  [<f8e77558>] ? firegl_trace+0x28/0x190 [fglrx]
[ 1485.320013]  [<f8eb9ccb>] ? CMMQS_ProcessTerminate+0x1b/0x30 [fglrx]
[ 1485.320013]  [<f8e7ac25>] ? firegl_cmmqs_ProcessTerminate+0x35/0xd0 [fglrx]
[ 1485.320013]  [<c0174058>] ? up+0x28/0x40
[ 1485.320013]  [<f8e50a5d>] ? firegl_release_helper+0x41d/0x790 [fglrx]
[ 1485.320013]  [<f8e52987>] ? firegl_release+0x77/0x220 [fglrx]
[ 1485.320013]  [<f8e47fc3>] ? ip_firegl_release+0x13/0x20 [fglrx]
[ 1485.320013]  [<c02255e4>] ? __fput+0xe4/0x1e0
[ 1485.320013]  [<c02256fd>] ? fput+0x1d/0x30
[ 1485.320013]  [<c022206c>] ? filp_close+0x4c/0x80
[ 1485.320013]  [<c015543b>] ? put_files_struct+0x6b/0xb0
[ 1485.320013]  [<c01554c8>] ? exit_files+0x48/0x60
[ 1485.320013]  [<c01577b4>] ? do_exit+0x134/0x340
[ 1485.320013]  [<c01579fe>] ? do_group_exit+0x3e/0xa0
[ 1485.320013]  [<c0157a78>] ? sys_exit_group+0x18/0x20
[ 1485.320013]  [<c01093df>] ? sysenter_do_call+0x12/0x28
[ 1485.320013] Code: 5c c7 45 ac 00 00 00 00 8b 45 08 83 c0 50 89 45 a8 8d 74 26
00 c7 45 c8 00 00 00 00 8b 55 a8 31 c0 c7 45 cc 00 00 00 00 89 55 d8 <8b> 1a 85
db 89 5d d0 74 03 8b 43 18 89 47 0c 89 7c 24 04 8b 4f
[ 1485.320013] EIP: [<f8ebd006>] _ZN17SegmentMapManager13deleteMappingEP9CMMClie
nt+0x36/0x160 [fglrx] SS:ESP 0068:f0b4bc70
[ 1485.320013] CR2: 0000000000000118
[ 1485.326578] ---[ end trace ac41df629658cb04 ]---
[ 1485.326621] Fixing recursive fault but reboot is needed!
```



vmware.log – VUsb Analyzer

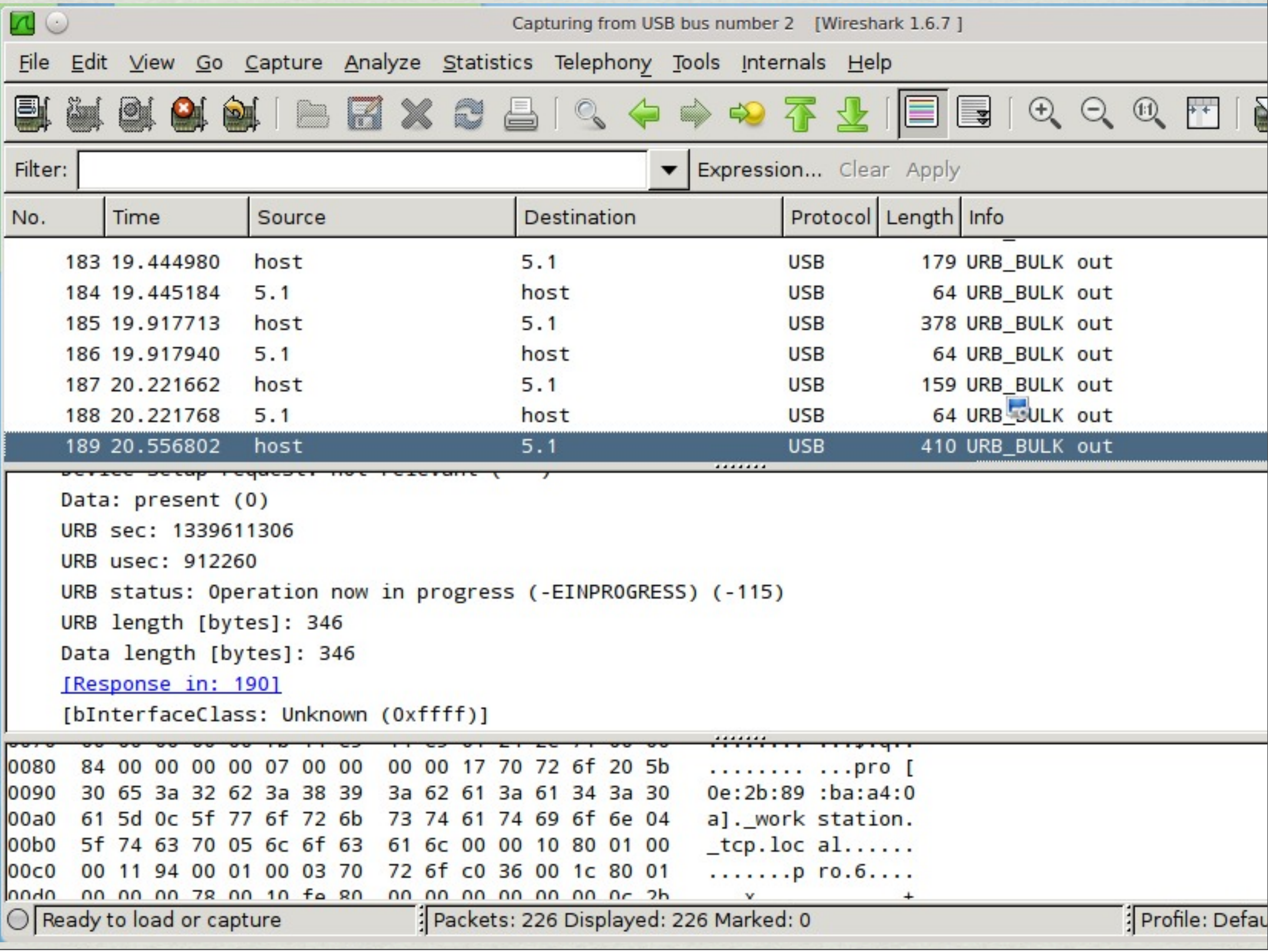
Transaction Detail – VUsb Analyzer

```
0000:  3F AA AA 05 01 50 D5 00 30 23 38 26 36 39 62 35  ?....P..0#8&69b5
0010:  33 62 39 26 30 26 30 30 30 30 26 31 23 7B 34 64  3b9&0&0000&1#{4d
0020:  31 65 35 35 62 32 2D 66 31 36 66 2D 31 31 63 66  1e55b2-f16f-11cf
0030:  2D 38 38 63 62 2D 30 30 31 31 31 31 30 30 30 30  -88cb-0011110000
```

	Device	Length	Setup	Data	Decoded
33	4	0x0000	21 0A 00 00 00 00 00 00		class interface 0x0a(v
35	4	0x0000	21 0A 00 00 00 00 00 00	Status: 3	
38	4	0x0064	81 06 00 22 00 00 64 00		GetDescriptor(0x22, 0)
40	4	0x0024	81 06 00 22 00 00 64 00	06 00 FF 09 01 A1 01 85 3F 95 3F 75 08 25 01 15 ....?.?u.%..	GetDescriptor(0x22, 0)
45	4	0x0040			
46	4	0x0040			
47	4	0x0040		3F AA AA 05 01 50 D5 00 30 23 38 26 36 39 62 35 ?....P..0#8&69b5	
52	4	0x0040			
53	4	0x0040		3F 09 AA 09 01 00 20 2C 32 B7 35 1F 5F F2 B7 BB ?..... ,2.5._...	
58	4	0x0040			
59	4	0x0040		3F AA AA 05 05 74 93 00 30 23 38 26 36 39 62 35 ?....t..0#8&69b5	
64	4	0x0040			
65	4	0x0040		3F 07 AA 07 05 01 03 C4 2C F2 10 CB F7 AB 6F DB ?.......,.....o.	
70	4	0x0040			
72	4	0x0040		3F AA AA 05 01 50 D5 00 30 23 38 26 36 39 62 35 ?....P..0#8&69b5	
77	4	0x0040			
78	4	0x0040		3F 09 AA 09 01 00 20 2C 32 B7 35 1F 5F F2 B7 BB ?..... ,2.5._...	
83	4	0x0040			
84	4	0x0040		3F AA AA 05 05 74 93 00 30 23 38 26 36 39 62 35 ?....t..0#8&69b5	

0.062 kB, 0.000000 s, inf kB/s | 396.832

Thursday, April 25, 13



```
No.   Time        Source   Destination   Protocol  Length  Info
  183 19.444980   host     5.1           USB          179  URB_BULK out
  184 19.445184   5.1      host          USB           64  URB_BULK out
  185 19.917713   host     5.1           USB          378  URB_BULK out
  186 19.917940   5.1      host          USB           64  URB_BULK out
  187 20.221662   host     5.1           USB          159  URB_BULK out
  188 20.221768   5.1      host          USB           64  URB_BULK out
  189 20.556802   host     5.1           USB          410  URB_BULK out
```

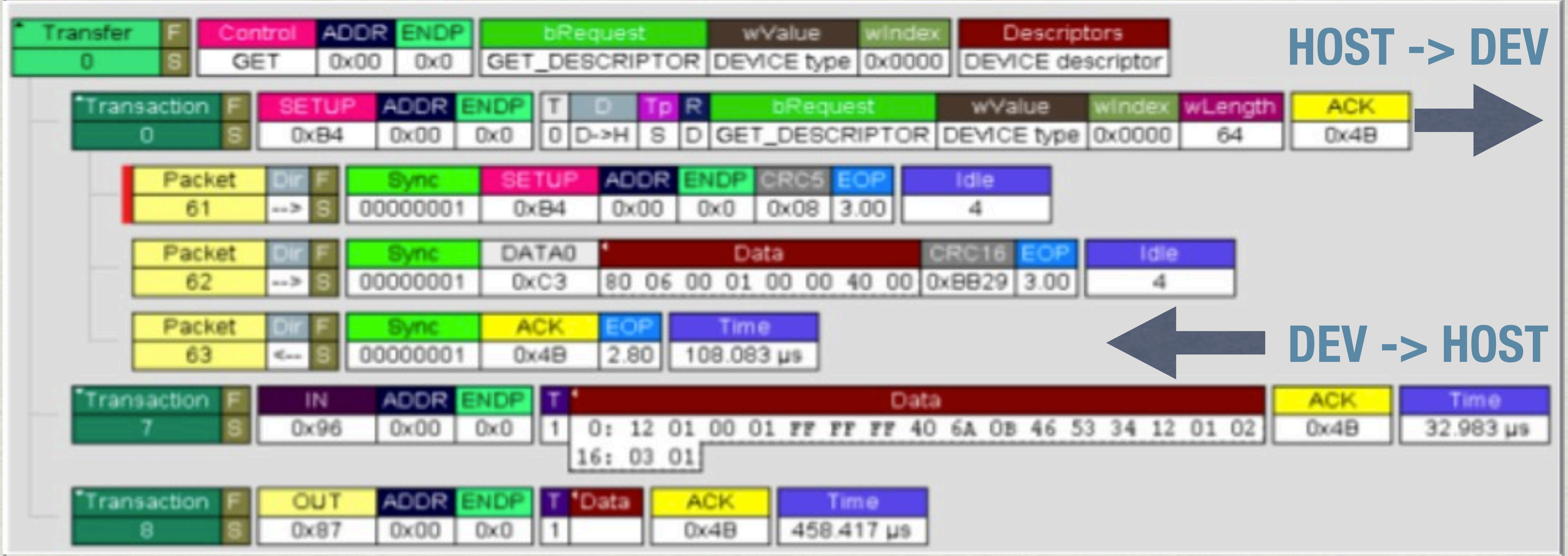
```
    Data: present (0)
    URB sec: 1339611306
    URB usec: 912260
    URB status: Operation now in progress (-EINPROGRESS) (-115)
    URB length [bytes]: 346
    Data length [bytes]: 346
    [Response in: 190]
    [bInterfaceClass: Unknown (0xffff)]
```

```
0080  84 00 00 00 00 07 00 00  00 00 17 70 72 6f 20 5b   ........ ...pro [
0090  30 65 3a 32 62 3a 38 39  3a 62 61 3a 61 34 3a 30   0e:2b:89 :ba:a4:0
00a0  61 5d 0c 5f 77 6f 72 6b  73 74 61 74 69 6f 6e 04   a]._work station.
00b0  5f 74 63 70 05 6c 6f 63  61 6c 00 00 10 80 01 00   _tcp.loc al......
00c0  00 11 94 00 01 00 03 70  72 6f c0 36 00 1c 80 01   .......p ro.6....
00d0  00 00 00 78 00 10 fe 80  00 00 00 00 00 00 0c 2b   ...x.... .......+
```

USB glossary

- Ports are called ***Endpoints***. EP0 or the SETUP endpoint is for auto-configuration (think a "broadcast address" for setup)
- Unconfigured devices respond to "broadcasts", send their **Descriptors**
- This setup exchange is called ***Enumeration***
- Host assigns device number (~address on the bus)

On the wire with MAX3420



Transfer	F	Control	ADDR	ENDP	bRequest	wValue	wIndex	Descriptors
0	S	GET	0x00	0x0	GET_DESCRIPTOR	DEVICE type	0x0000	DEVICE descriptor

Transaction	F	SETUP	ADDR	ENDP	T	D	Tp	R	bRequest	wValue	wIndex	wLength	ACK
0	S	0xB4	0x00	0x0	0	D->H	S	D	GET_DESCRIPTOR	DEVICE type	0x0000	64	0x4B

Packet	Dir	F	Sync	SETUP	ADDR	ENDP	CRC5	EOP	Idle
61	-->	S	00000001	0xB4	0x00	0x0	0x08	3.00	4

Packet	Dir	F	Sync	DATA0	Data	CRC16	EOP	Idle
62	-->	S	00000001	0xC3	80 06 00 01 00 00 40 00	0xBB29	3.00	4

Packet	Dir	F	Sync	ACK	EOP	Time
63	<--	S	00000001	0x4B	2.80	108.083 µs

Transaction	F	IN	ADDR	ENDP	T	Data	ACK	Time
7	S	0x96	0x00	0x0	1	0: 12 01 00 01 FF FF FF 40 6A 0B 46 53 34 12 01 02 16: 03 01	0x4B	32.983 µs

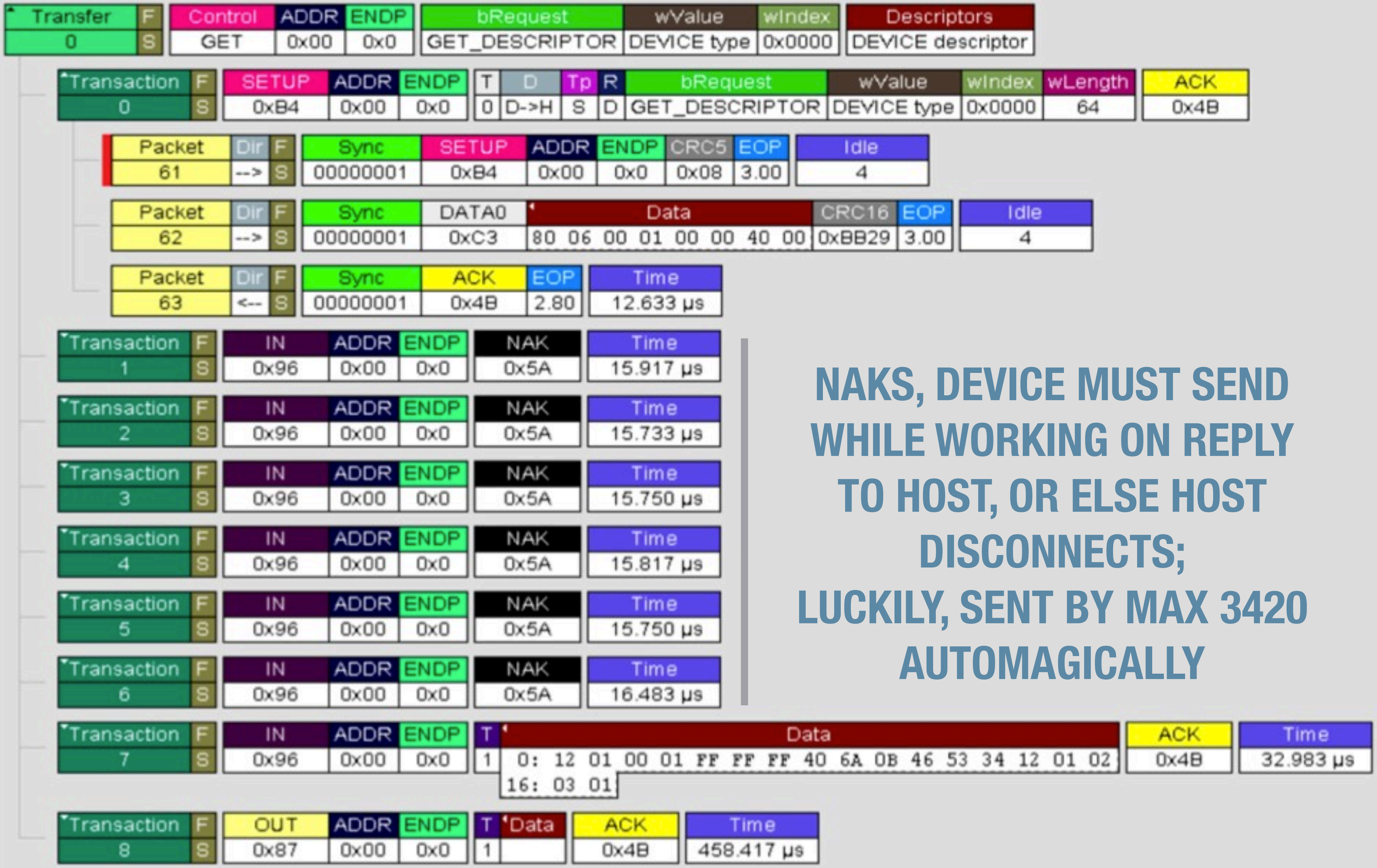
Transaction	F	OUT	ADDR	ENDP	T	Data	ACK	Time
8	S	0x87	0x00	0x0	1		0x4B	458.417 µs

HOST -> DEV

DEV -> HOST

- USB host acquires **device descriptors (tables)**
- Looks up driver by device/vendor numbers
- Sets up kernel “routing” through the stack layers

On the wire with MAX3420



Transfer	F	Control	ADDR	ENDP	bRequest	wValue	wIndex	Descriptors
0	S	GET	0x00	0x0	GET_DESCRIPTOR	DEVICE type	0x0000	DEVICE descriptor

Transaction	F	SETUP	ADDR	ENDP	T	D	Tp	R	bRequest	wValue	wIndex	wLength	ACK
0	S	0xB4	0x00	0x0	0	D->H	S	D	GET_DESCRIPTOR	DEVICE type	0x0000	64	0x4B

Packet	Dir	F	Sync	SETUP	ADDR	ENDP	CRC5	EOP	Idle
61	-->	S	00000001	0xB4	0x00	0x0	0x08	3.00	4

Packet	Dir	F	Sync	DATA0	Data	CRC16	EOP	Idle
62	-->	S	00000001	0xC3	80 06 00 01 00 00 40 00	0xBB29	3.00	4

Packet	Dir	F	Sync	ACK	EOP	Time
63	<--	S	00000001	0x4B	2.80	12.633 µs

Transaction	F	IN	ADDR	ENDP	NAK	Time
1	S	0x96	0x00	0x0	0x5A	15.917 µs
2	S	0x96	0x00	0x0	0x5A	15.733 µs
3	S	0x96	0x00	0x0	0x5A	15.750 µs
4	S	0x96	0x00	0x0	0x5A	15.817 µs
5	S	0x96	0x00	0x0	0x5A	15.750 µs
6	S	0x96	0x00	0x0	0x5A	16.483 µs

Transaction	F	IN	ADDR	ENDP	T	Data	ACK	Time
7	S	0x96	0x00	0x0	1	0: 12 01 00 01 FF FF FF 40 6A 0B 46 53 34 12 01 02 16: 03 01	0x4B	32.983 µs

Transaction	F	OUT	ADDR	ENDP	T	Data	ACK	Time
8	S	0x87	0x00	0x0	1		0x4B	458.417 µs

NAKS, DEVICE MUST SEND WHILE WORKING ON REPLY TO HOST, OR ELSE HOST DISCONNECTS;
LUCKILY, SENT BY MAX 3420 AUTOMAGICALLY

USB devices, in Python

- **Class types** are standardized. (**HID**, **Mass Storage**) **Vendor types** are not (e.g., FTDI, **Wi-Fi**).
- Descriptors have structs unique to each **device class**
- Fairly complex: nested lengths, offsets
  => parser bugs
- Be the host's worst driver nightmare - in Python: http://goodfet.sf.net/

Facedancer

"If you can write a webserver,
you can write a disk"

http://goodfet.sf.net/

“The Dark Side of ~~Socks~~ OS Code”

- Descriptor structs are unique to each device class: **Nested lengths**, in-struct **offsets** = trouble

Exploiting enumeration

- Host requests the first few bytes of the descriptor.
- Host mallocs that many bytes.
- Host reads the entire descriptor into a temporary buffer.
- Host memcpy() the descriptor into the malloced buffer.
- ***PSGroove*** exploits this on the Playstation 3!

Exploit Dev Cycle
Before & After

1. Change your code.
2. Plug the dongle into your workstation.
3. Reflash it.
4. Move the dongle to your target.
5. Try it.

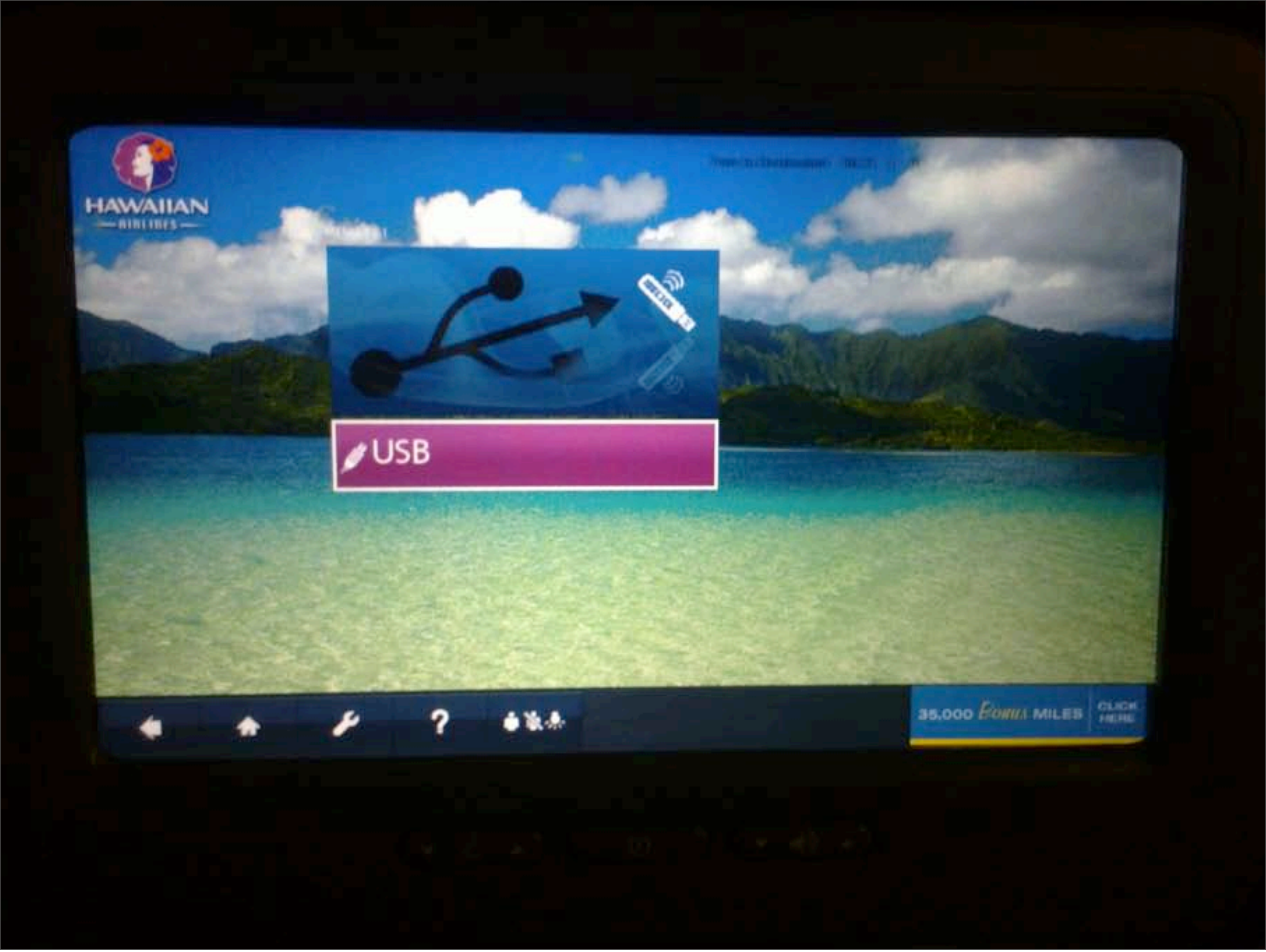
1. Change your code
2. Try it

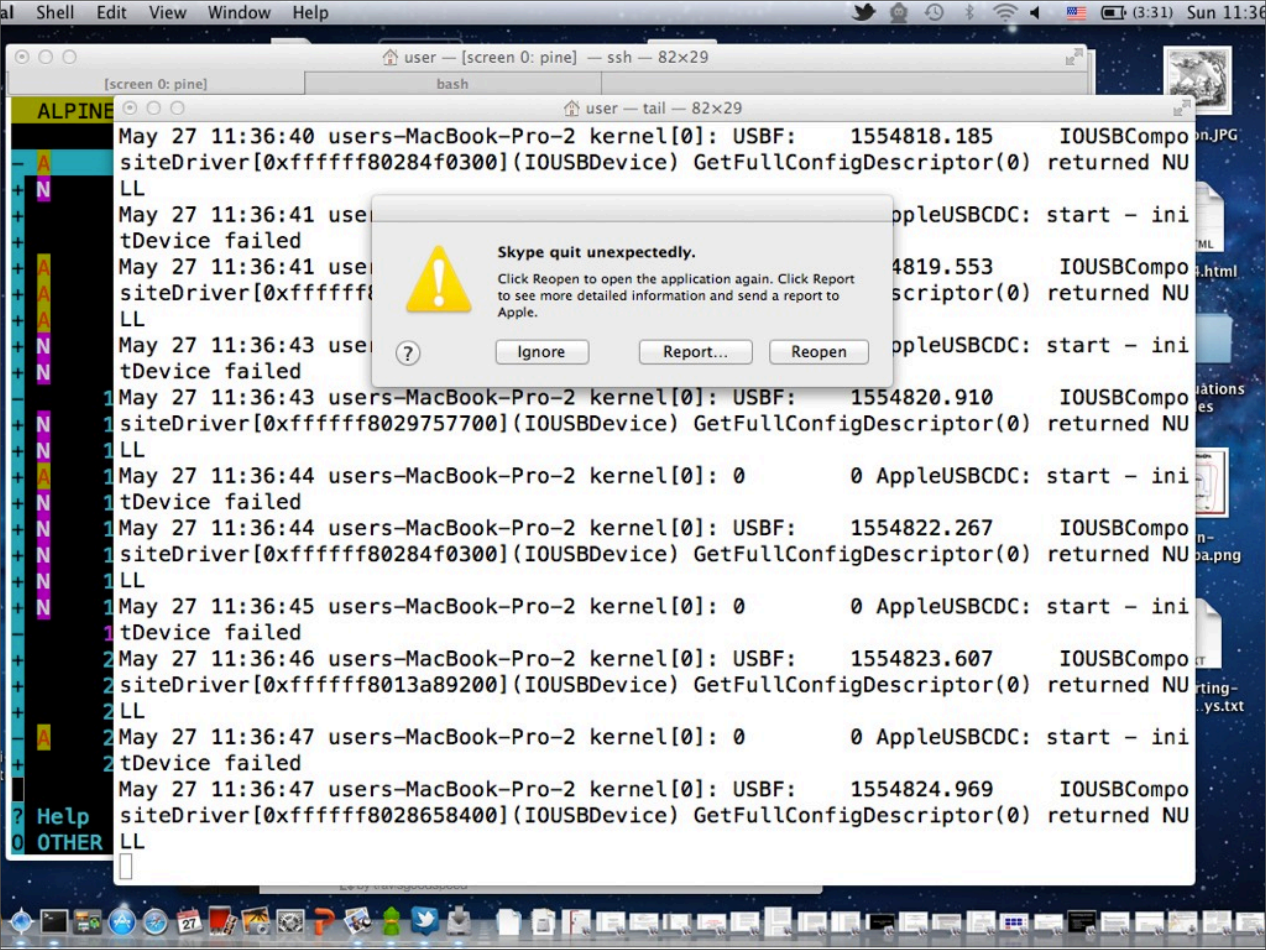
HID Emulation

- **python goodfet.maxusbhid**
- Easiest to implement.
- Lots of prior examples,
  - Social Engineering Toolkit
  - **Teensy**, AVR USB Key, vendor examples
- Embarrassing bugs remain!

HID Format String

- Ubuntu 12.04, Xorg
- Manufacturer String:
  “%n%s%n%s%n%s”
- Device String:
  “%n%s%n%s%n%s”
- Thanks to the ChromeOS team!


HAWAIIAN
AIRLINES
USB
35,000 Bonus MILES
CLICK HERE



```
May 27 11:36:40 users-MacBook-Pro-2 kernel[0]: USBF:     1554818.185    IOUSBCompo
siteDriver[0xffffff80284f0300](IOUSBDevice) GetFullConfigDescriptor(0) returned NU
LL
May 27 11:36:41 use                                      ppleUSBCDC: start - ini
tDevice failed
May 27 11:36:41 use                                      4819.553    IOUSBCompo
siteDriver[0xffffff                                      scriptor(0) returned NU
LL
May 27 11:36:43 use                                      ppleUSBCDC: start - ini
tDevice failed
May 27 11:36:43 users-MacBook-Pro-2 kernel[0]: USBF:     1554820.910    IOUSBCompo
siteDriver[0xffffff8029757700](IOUSBDevice) GetFullConfigDescriptor(0) returned NU
LL
May 27 11:36:44 users-MacBook-Pro-2 kernel[0]: 0         0 AppleUSBCDC: start - ini
tDevice failed
May 27 11:36:44 users-MacBook-Pro-2 kernel[0]: USBF:     1554822.267    IOUSBCompo
siteDriver[0xffffff80284f0300](IOUSBDevice) GetFullConfigDescriptor(0) returned NU
LL
May 27 11:36:45 users-MacBook-Pro-2 kernel[0]: 0         0 AppleUSBCDC: start - ini
tDevice failed
May 27 11:36:46 users-MacBook-Pro-2 kernel[0]: USBF:     1554823.607    IOUSBCompo
siteDriver[0xffffff8013a89200](IOUSBDevice) GetFullConfigDescriptor(0) returned NU
LL
May 27 11:36:47 users-MacBook-Pro-2 kernel[0]: 0         0 AppleUSBCDC: start - ini
tDevice failed
May 27 11:36:47 users-MacBook-Pro-2 kernel[0]: USBF:     1554824.969    IOUSBCompo
siteDriver[0xffffff8028658400](IOUSBDevice) GetFullConfigDescriptor(0) returned NU
LL
```

Skype quit unexpectedly.

Click Reopen to open the application again. Click Report to see more detailed information and send a report to Apple.

Host Mode Emulation

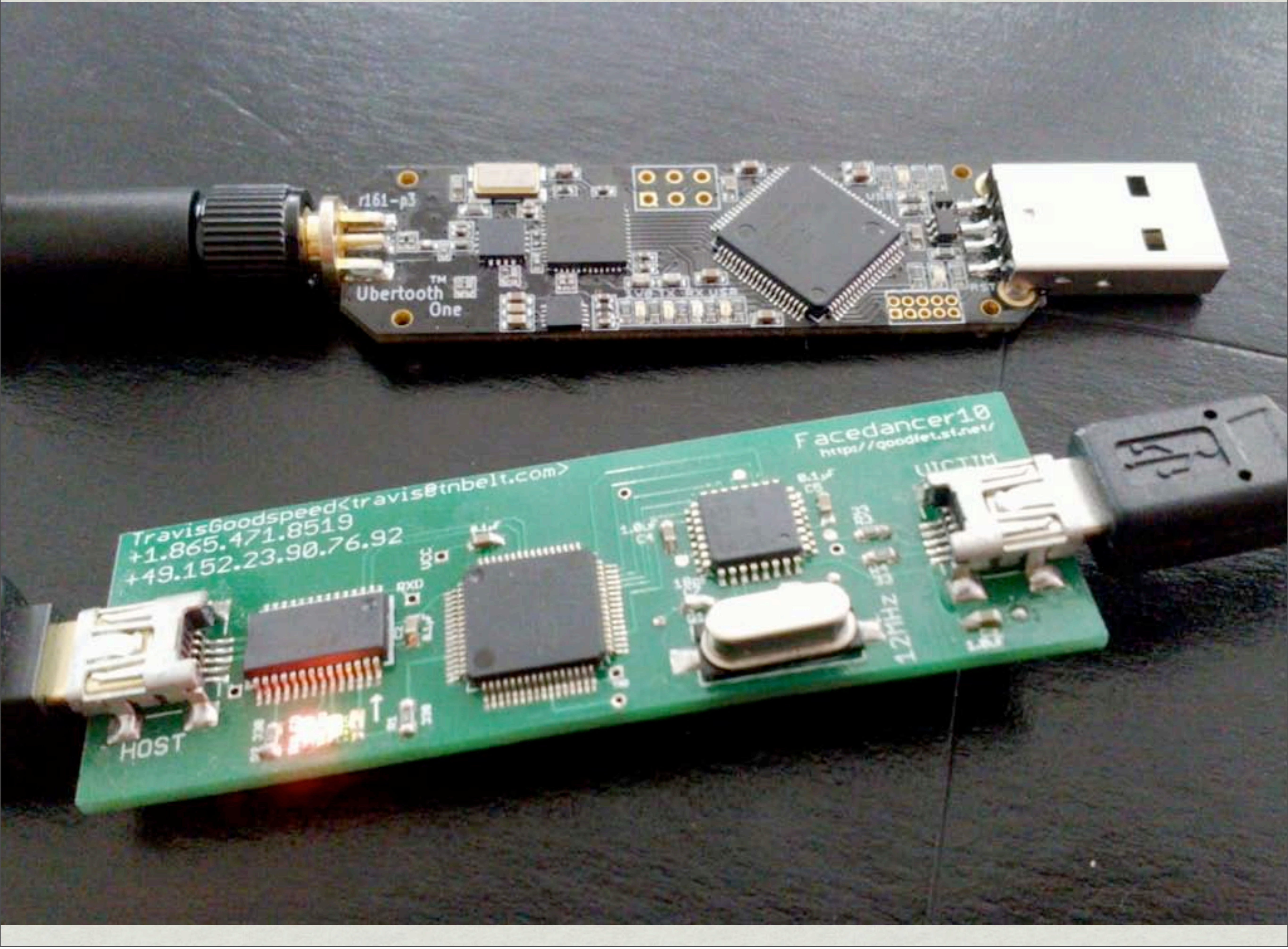
- Roundtrip time becomes an issue. (Only on OS X)
- Code is already in SVN, hardware coming in FD20.
- Firmware security is even worse than in drivers!
- Most exploits can use **libusb** instead of a Facedancer.

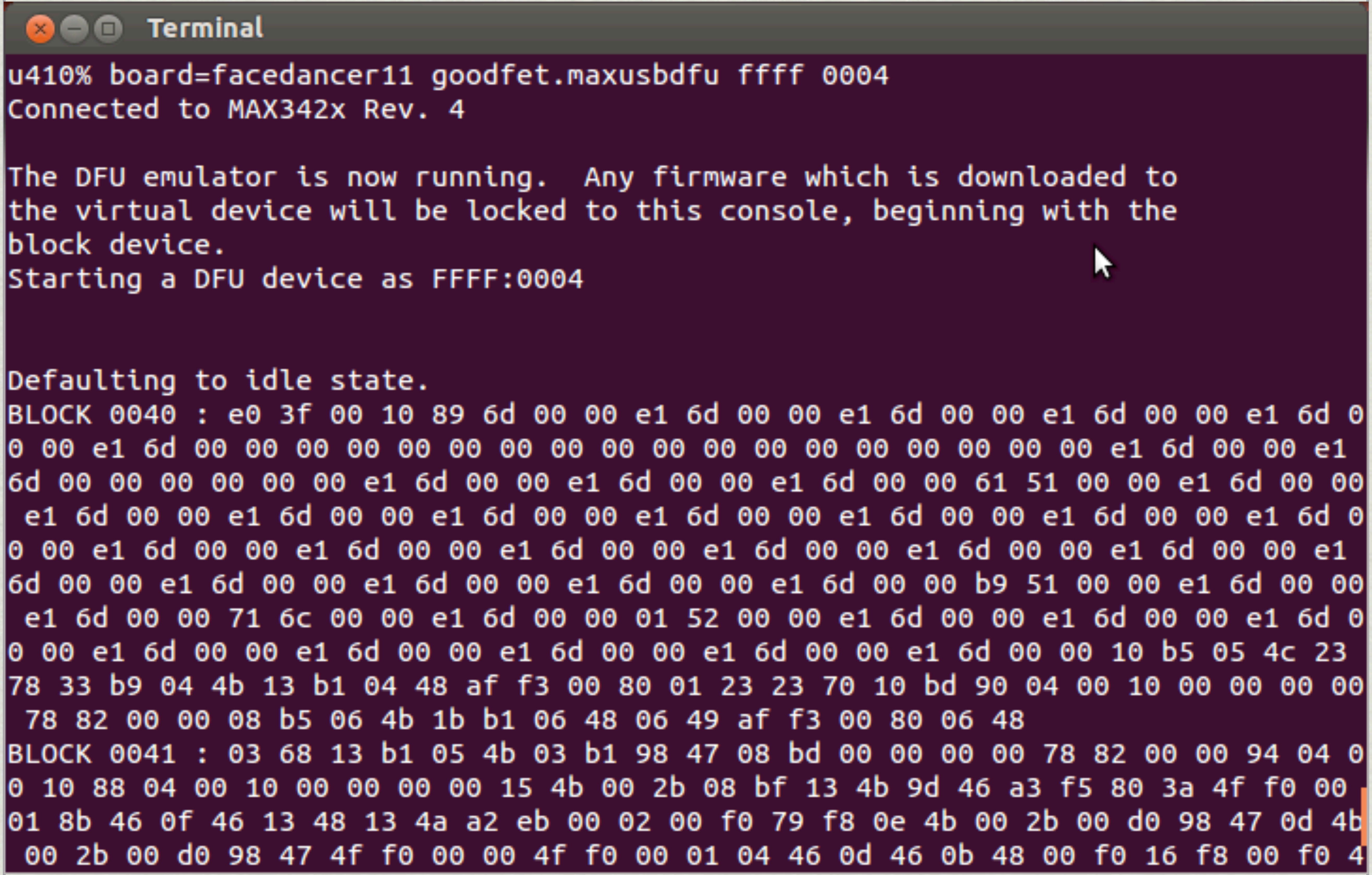
Device Bugs

- Memory exposed by reads past the end of the Strings table.
- Integer overflows, stack smashing, etc.
- Never any ASLR; any DEP is accidental.

Device Firmware Update (DFU)

- Device Firmware Update Protocol
  - iPhone, iPod, and other MP3 players.
  - Handy attack target.
- Facedancer supported.


r161-p3
TM
Ubertooth One
Facedancer10
http://goodfet.sf.net/
VICTIM
TravisGoodspeed<travis@tnbelt.com>
+1.865.471.8519
+49.152.23.90.76.92
VCC
RXD
12MHz
HOST

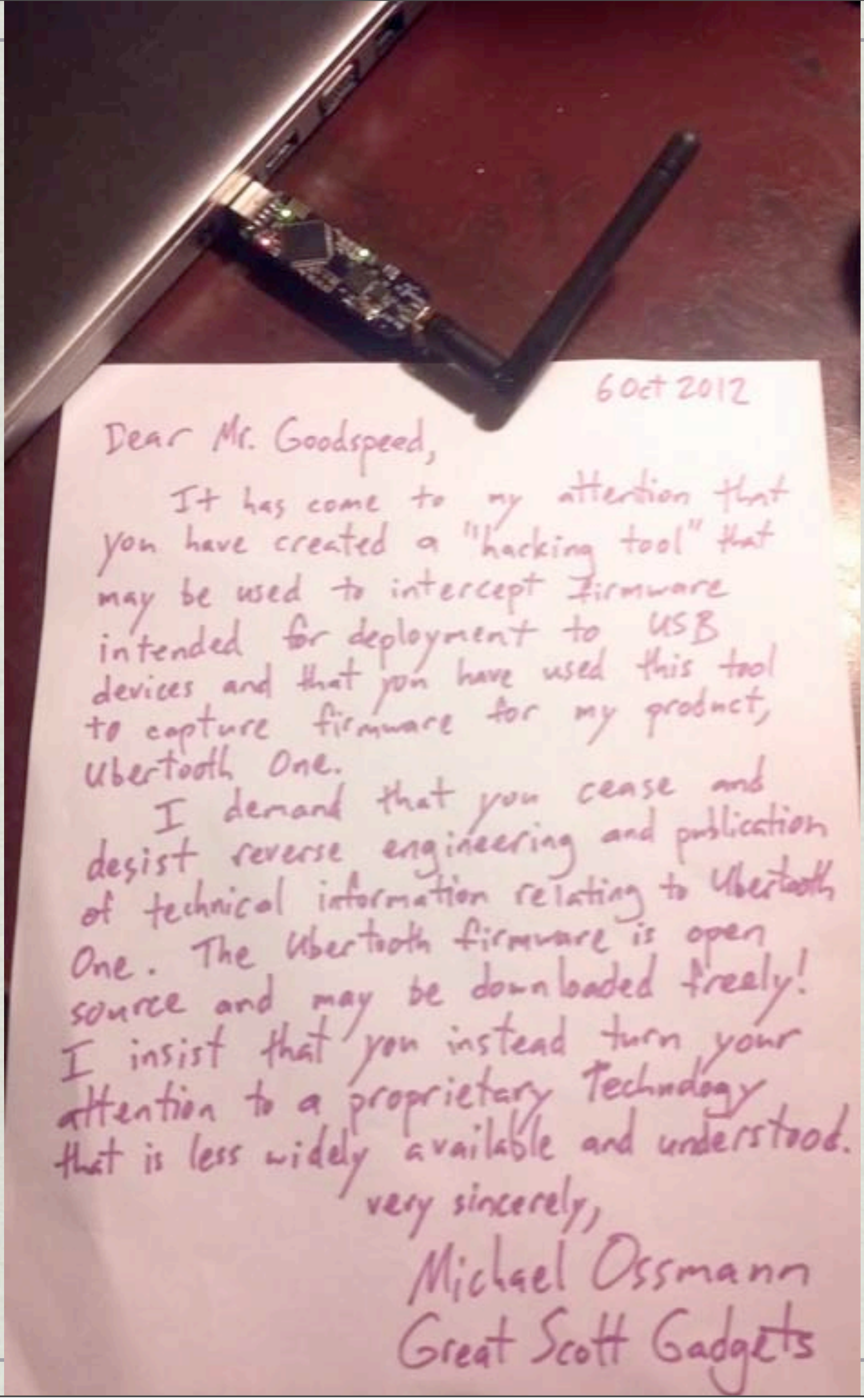


```
u410% board=facedancer11 goodfet.maxusbdfu ffff 0004
Connected to MAX342x Rev. 4

The DFU emulator is now running.  Any firmware which is downloaded to
the virtual device will be locked to this console, beginning with the
block device.
Starting a DFU device as FFFF:0004



Defaulting to idle state.
BLOCK 0040 : e0 3f 00 10 89 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 0
0 00 e1 6d 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e1 6d 00 00 e1
6d 00 00 00 00 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 61 51 00 00 e1 6d 00 00
 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 0
0 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1
6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 b9 51 00 00 e1 6d 00 00
 e1 6d 00 00 71 6c 00 00 e1 6d 00 00 01 52 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 0
0 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 e1 6d 00 00 10 b5 05 4c 23
78 33 b9 04 4b 13 b1 04 48 af f3 00 80 01 23 23 70 10 bd 90 04 00 10 00 00 00 00
 78 82 00 00 08 b5 06 4b 1b b1 06 48 06 49 af f3 00 80 06 48
BLOCK 0041 : 03 68 13 b1 05 4b 03 b1 98 47 08 bd 00 00 00 00 78 82 00 00 94 04 0
0 10 88 04 00 10 00 00 00 00 15 4b 00 2b 08 bf 13 4b 9d 46 a3 f5 80 3a 4f f0 00
01 8b 46 0f 46 13 48 13 4a a2 eb 00 02 00 f0 79 f8 0e 4b 00 2b 00 d0 98 47 0d 4b
 00 2b 00 d0 98 47 4f f0 00 00 4f f0 00 01 04 46 0d 46 0b 48 00 f0 16 f8 00 f0 4
```



6 Oct 2012

Dear Mr. Goodspeed,

It has come to my attention that you have created a "hacking tool" that may be used to intercept firmware intended for deployment to USB devices and that you have used this tool to capture firmware for my product, Ubertooth One.

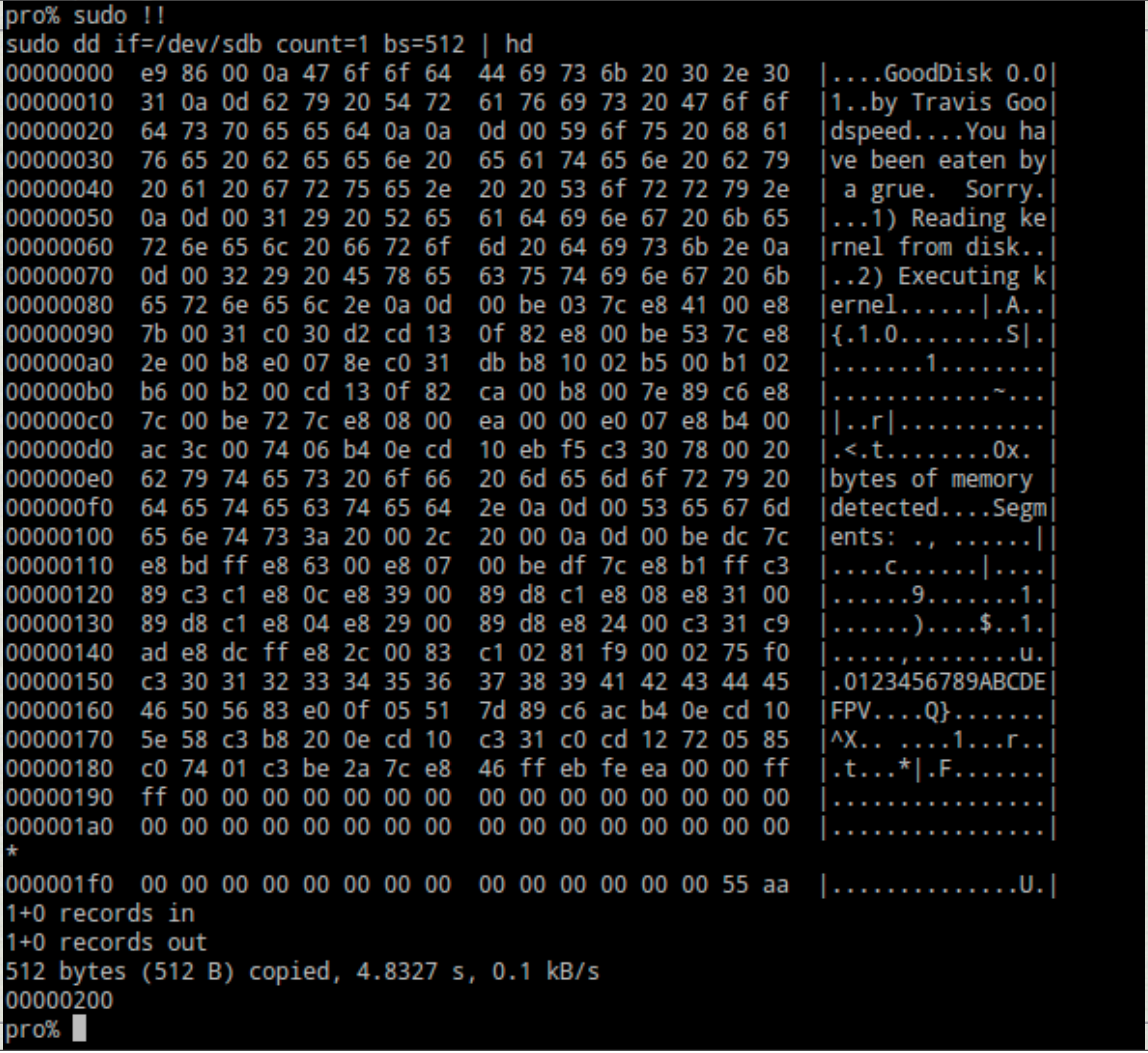
I demand that you cease and desist reverse engineering and publication of technical information relating to Ubertooth One. The Ubertooth firmware is open source and may be downloaded freely! I insist that you instead turn your attention to a proprietary technology that is less widely available and understood.

very sincerely,

Michael Ossmann
Great Scott Gadgets

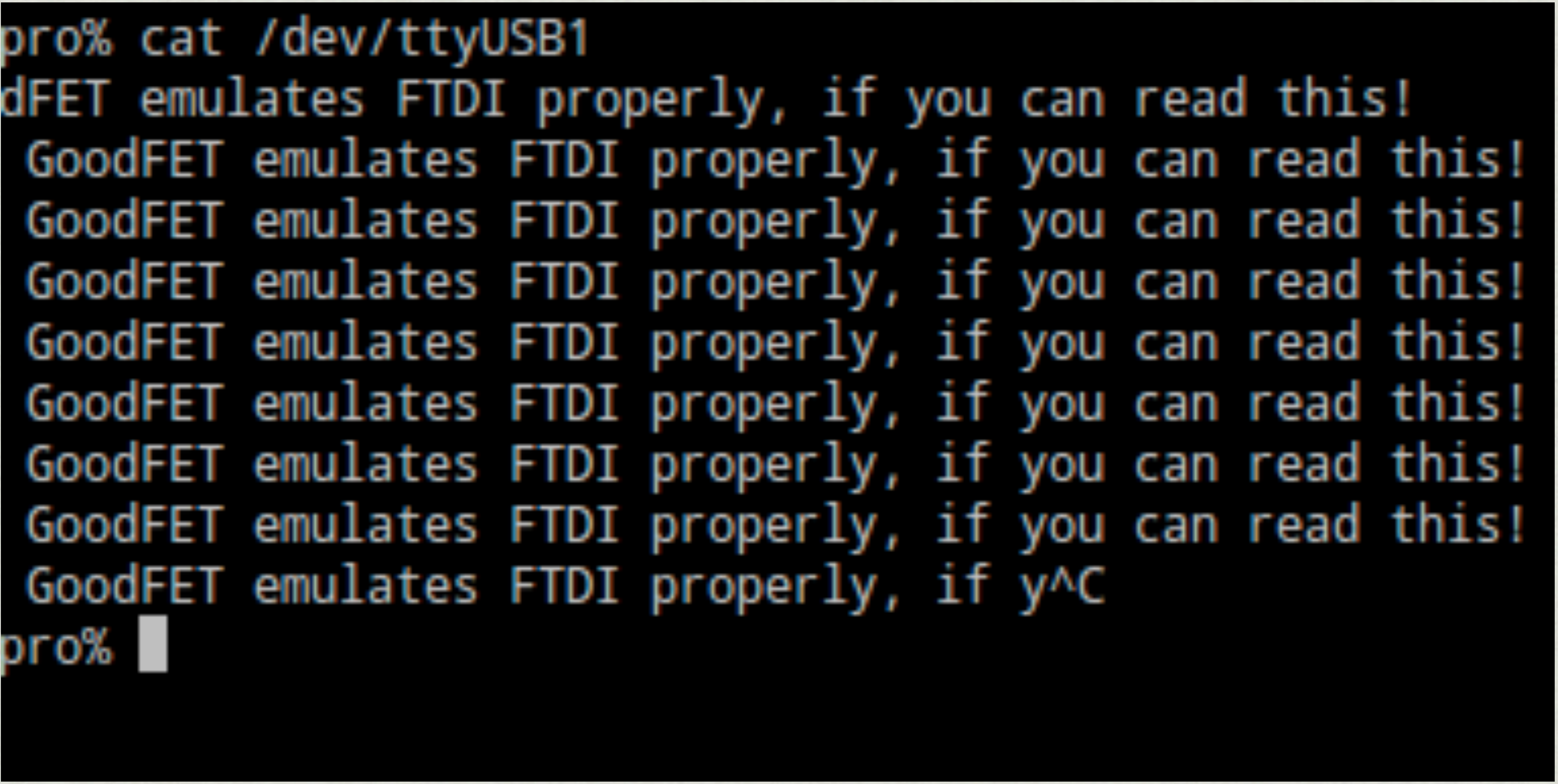
Mass Storage

- TOCTTOU Exploits
  - See Collin Mulliner’s at WOOT ’12.
- Active Antiforensics
  - Disk erases itself if forensically analyzed.



```
pro% sudo !!
sudo dd if=/dev/sdb count=1 bs=512 | hd
00000000  e9 86 00 0a 47 6f 6f 64  44 69 73 6b 20 30 2e 30  |....GoodDisk 0.0|
00000010  31 0a 0d 62 79 20 54 72  61 76 69 73 20 47 6f 6f  |1..by Travis Goo|
00000020  64 73 70 65 65 64 0a 0a  0d 00 59 6f 75 20 68 61  |dspeed....You ha|
00000030  76 65 20 62 65 65 6e 20  65 61 74 65 6e 20 62 79  |ve been eaten by|
00000040  20 61 20 67 72 75 65 2e  20 20 53 6f 72 72 79 2e  | a grue.  Sorry.|
00000050  0a 0d 00 31 29 20 52 65  61 64 69 6e 67 20 6b 65  |...1) Reading ke|
00000060  72 6e 65 6c 20 66 72 6f  6d 20 64 69 73 6b 2e 0a  |rnel from disk..|
00000070  0d 00 32 29 20 45 78 65  63 75 74 69 6e 67 20 6b  |..2) Executing k|
00000080  65 72 6e 65 6c 2e 0a 0d  00 be 03 7c e8 41 00 e8  |ernel......|.A..|
00000090  7b 00 31 c0 30 d2 cd 13  0f 82 e8 00 be 53 7c e8  |{.1.0........S|.|
000000a0  2e 00 b8 e0 07 8e c0 31  db b8 10 02 b5 00 b1 02  |.......1........|
000000b0  b6 00 b2 00 cd 13 0f 82  ca 00 b8 00 7e 89 c6 e8  |............~...|
000000c0  7c 00 be 72 7c e8 08 00  ea 00 00 e0 07 e8 b4 00  ||..r|...........|
000000d0  ac 3c 00 74 06 b4 0e cd  10 eb f5 c3 30 78 00 20  |.<.t........0x. |
000000e0  62 79 74 65 73 20 6f 66  20 6d 65 6d 6f 72 79 20  |bytes of memory |
000000f0  64 65 74 65 63 74 65 64  2e 0a 0d 00 53 65 67 6d  |detected....Segm|
00000100  65 6e 74 73 3a 20 00 2c  20 00 0a 0d 00 be dc 7c  |ents: ., ......||
00000110  e8 bd ff e8 63 00 e8 07  00 be df 7c e8 b1 ff c3  |....c......|....|
00000120  89 c3 c1 e8 0c e8 39 00  89 d8 c1 e8 08 e8 31 00  |......9.......1.|
00000130  89 d8 c1 e8 04 e8 29 00  89 d8 e8 24 00 c3 31 c9  |......)....$..1.|
00000140  ad e8 dc ff e8 2c 00 83  c1 02 81 f9 00 02 75 f0  |.....,........u.|
00000150  c3 30 31 32 33 34 35 36  37 38 39 41 42 43 44 45  |.0123456789ABCDE|
00000160  46 50 56 83 e0 0f 05 51  7d 89 c6 ac b4 0e cd 10  |FPV....Q}.......|
00000170  5e 58 c3 b8 20 0e cd 10  c3 31 c0 cd 12 72 05 85  |^X.. ....1...r..|
00000180  c0 74 01 c3 be 2a 7c e8  46 ff eb fe ea 00 00 ff  |.t...*|.F.......|
00000190  ff 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000001a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
000001f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 55 aa  |..............U.|
1+0 records in
1+0 records out
512 bytes (512 B) copied, 4.8327 s, 0.1 kB/s
00000200
pro% 
```

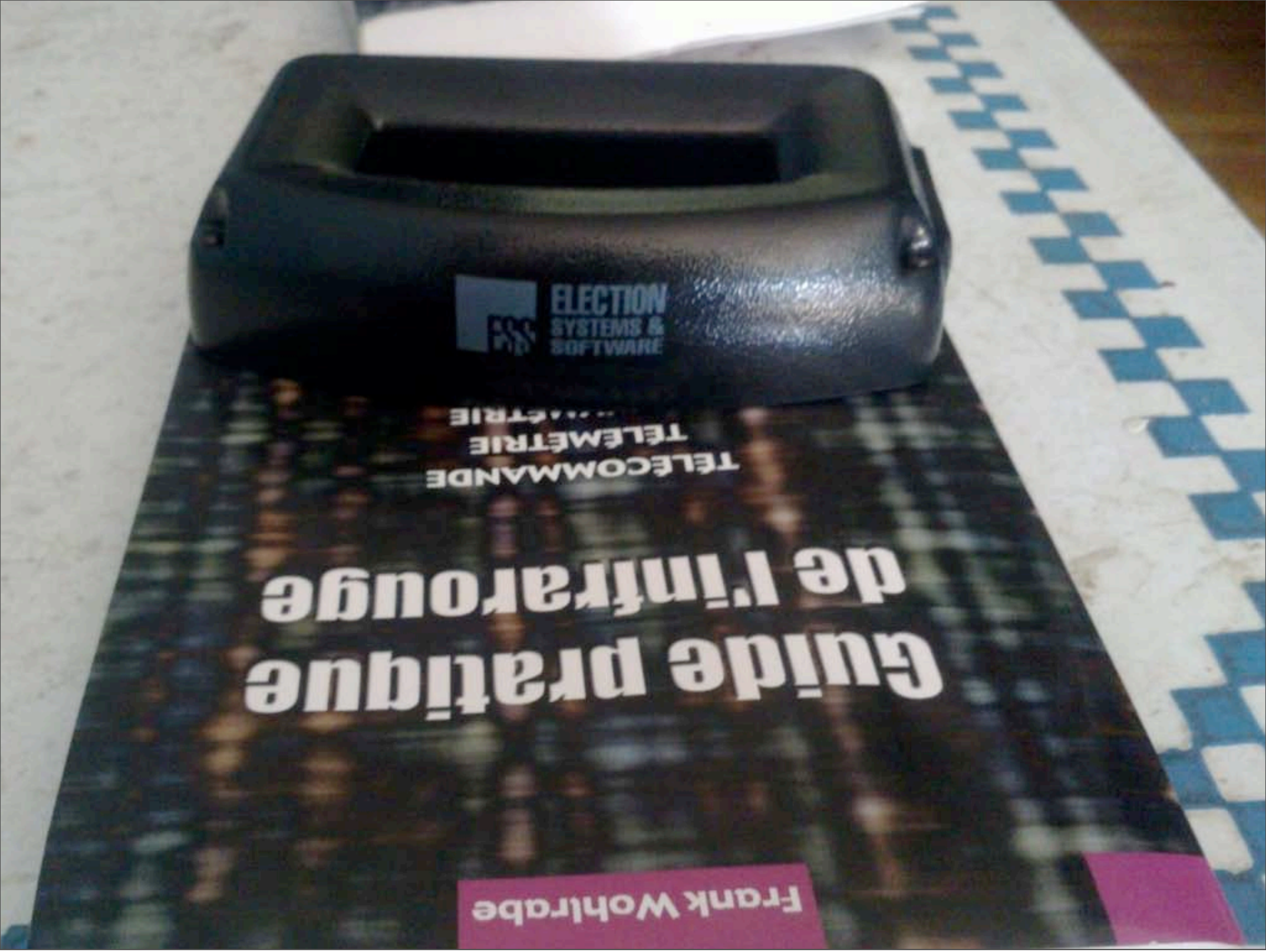
USB Serial Emulation



```
pro% cat /dev/ttyUSB1
dFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if you can read this!
 GoodFET emulates FTDI properly, if y^C
pro% 
```

USB Serial Emulation

- All sorts of things appear as a serial port.
  - Uninterruptible Power Supplies
  - Modems, Phones, Radios
  - Facedancer!



ELECTION
SYSTEMS &
SOFTWARE
TÉLÉCOMMANDE
TÉLÉMÉTRIE
...MÉTRIE
Guide pratique
de l'infrarouge
Frank Wohlrabe

Targets in Windows

- Unmaintained drivers are gold.
- Auto-installation approximates Linux variety.
  - **Variety**, but not speed.
- Windows 8 disables misbehaving USB ports.

Targets in Linux

- All drivers by default!
  - No loading delays!
- Massive attack surface.

Targets in Mac

- Holy crap the stack's performance is bad.
- Can't emulate HID on localhost!
- Lack of driver variety can limit attack surface.

Targets in FreeBSD

- Complex drivers not included by default.
  - Wifi, etc.
- Pay attention to **usbpf**.
- See our paper from **WESS 2012**.
  - Instrumentation with dtrace.

Conclusions

- USB opens a massive attack surface to inputs.
- Network stack exploration methods also work for USB stacks – similar “routing” structure to be exploited.
- We’ve begun to build tools to exploit this structure
- “Magical” abstractions lead to unrealistic validity assumptions ⇒ bugs, likely exploitable.
- Other buses: you are next!
  (If Daisho doesn’t beat us to it)

“Layers of abstraction become boundaries of competence”

← “Fast path”, cross-layer design

WTF 1.0, reference implementation →