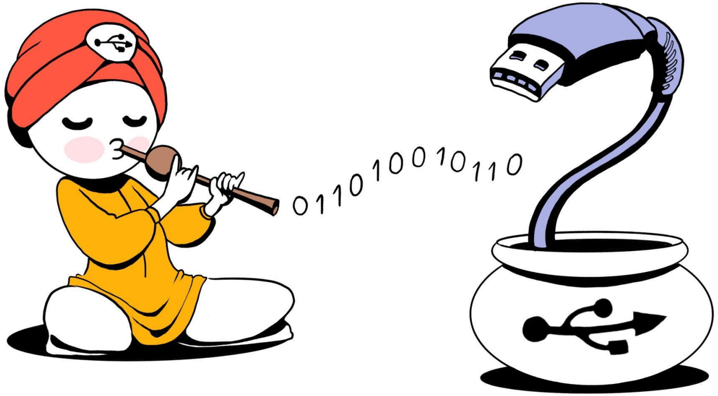


Introduction to USB Hacking

Andrey Konovalov <andreyknvl@gmail.com>

April 19th 2020

whoami

- Andrey Konovalov <andreyknvl@gmail.com>
- Software Engineer at Google
- Working on bug-finding tools for the Linux kernel (syzkaller, KASAN, …)
- More: xairy.github.io

- Why am talking about USB?

Materials

github.com/xairy/hardware-village => usb

Agenda

- Part 1: USB 101
- Part 2: USB Attack Surface
- Part 3: Linux USB subsystem
- Part 4: BadUSB
- Part 5: Facedancer
- Part 6: Linux USB Gadget Subsystem
- Part 7: USB Fuzzing
- Part 8: USB Sniffing

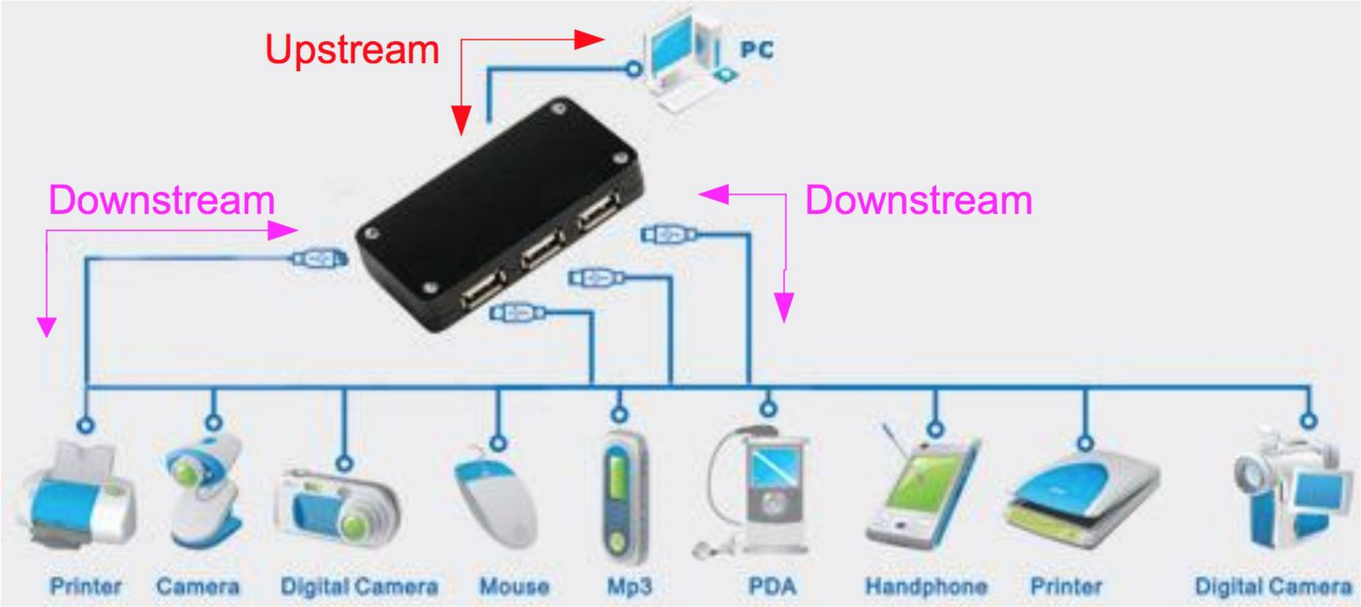
Part 1: USB 101

USB 101

- Based on USB 101: An Introduction to Universal Serial Bus 2.0

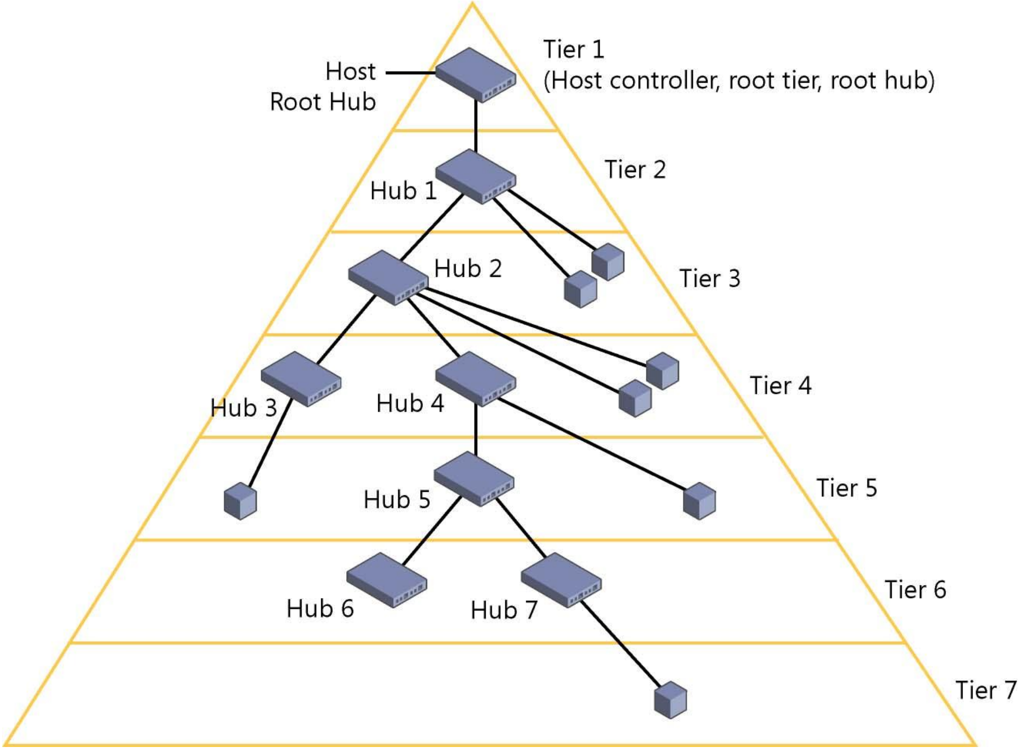
- USB Host
- USB Device (aka Gadget)
- USB Cable

USB Topology



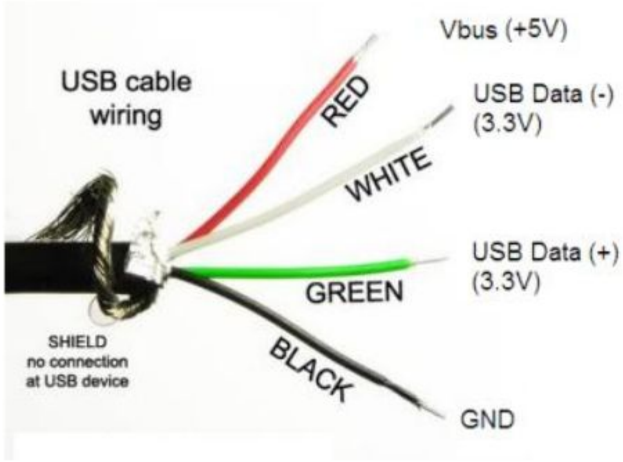
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

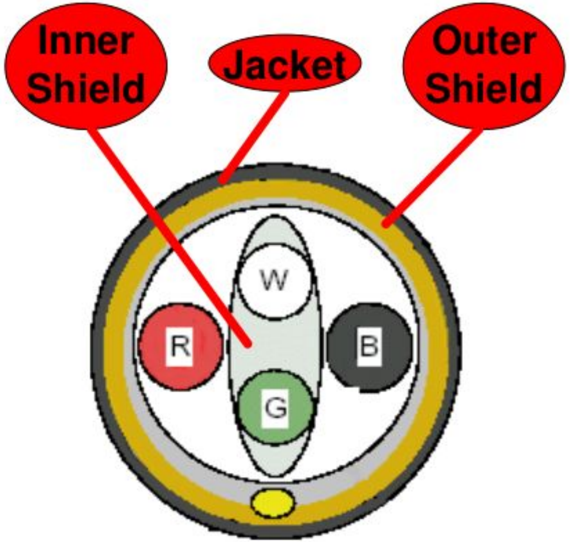
USB Hubs

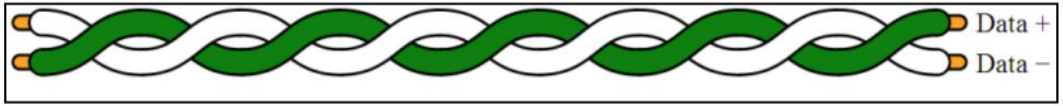


"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Cable

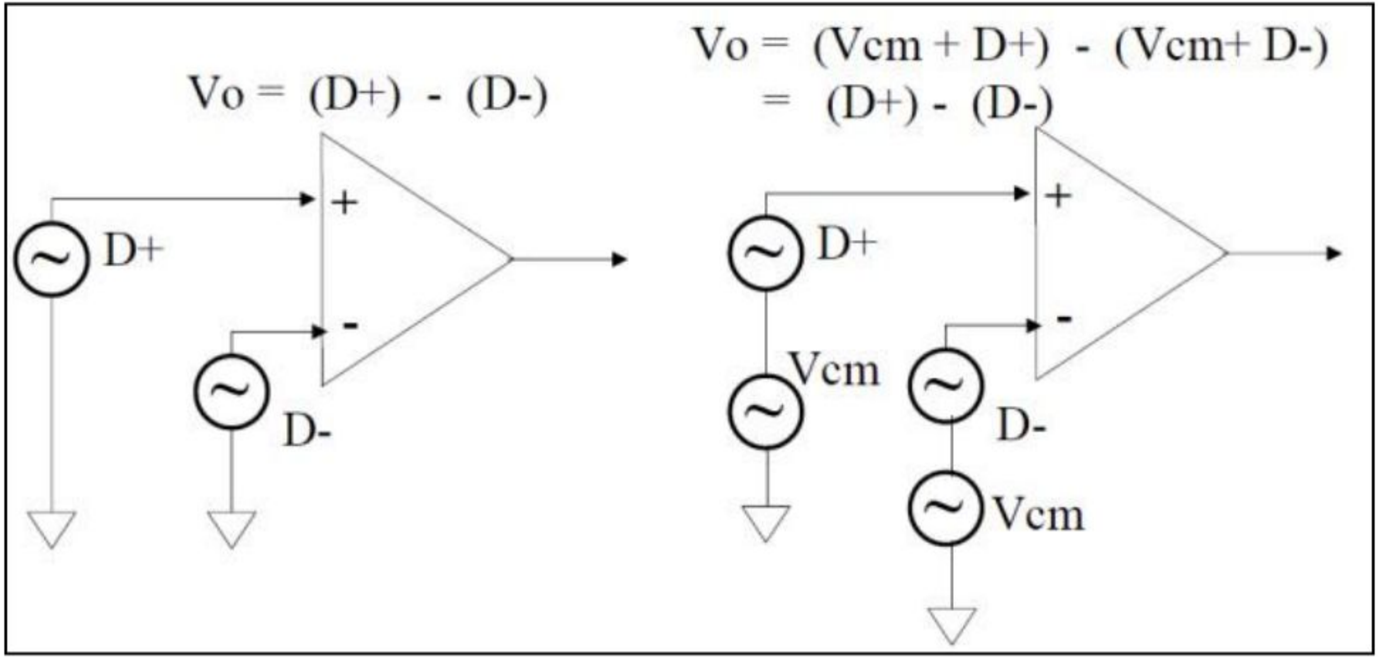






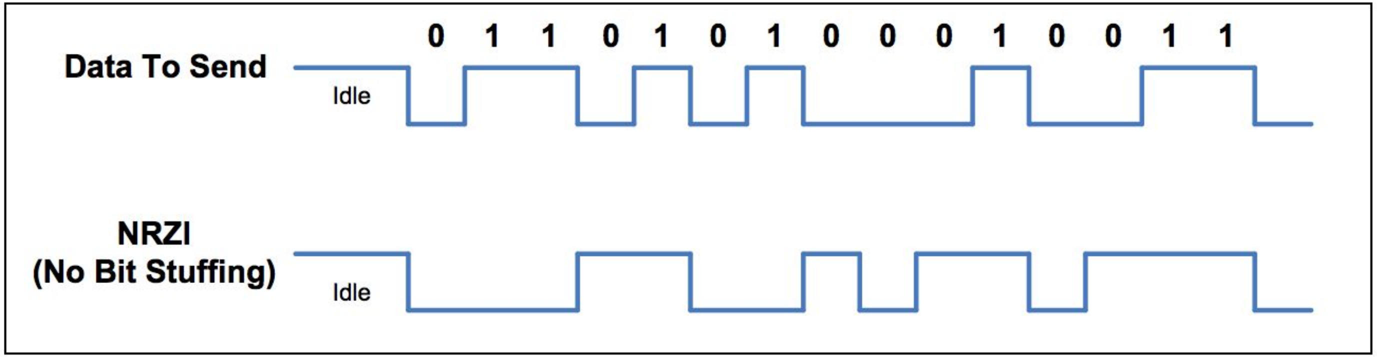
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

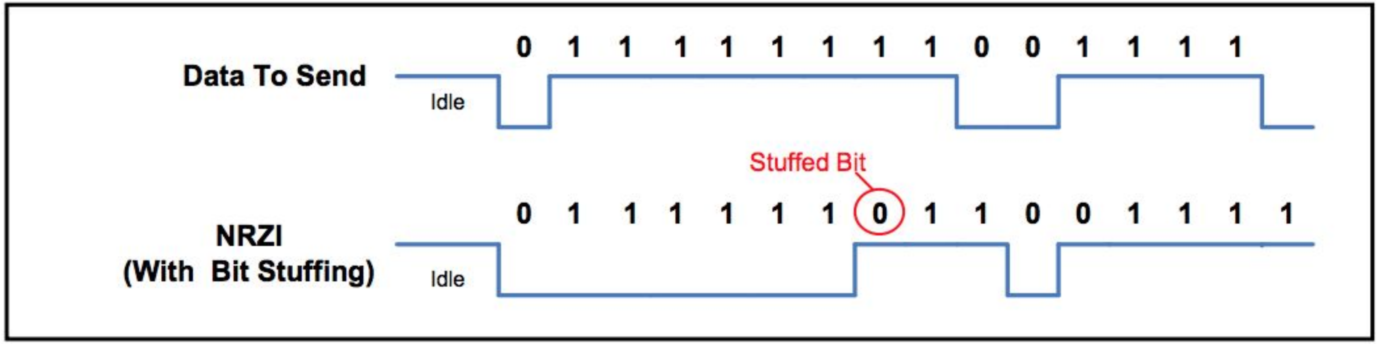
USB Differential Amplifier



"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

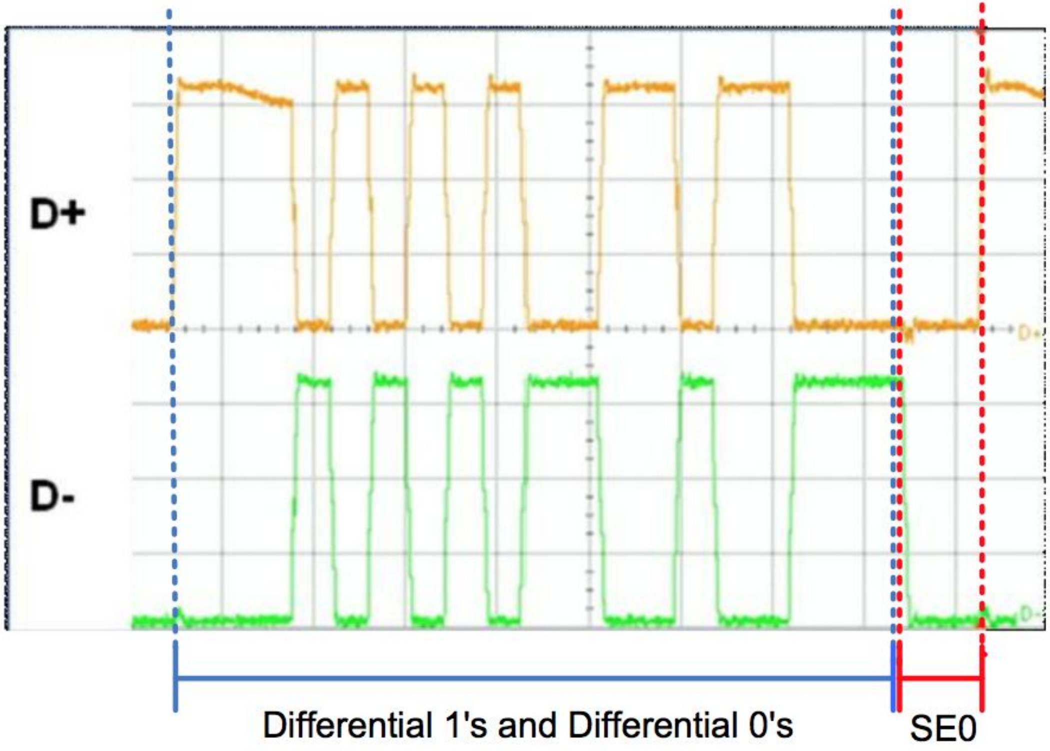
NRZI Encoding with Bit Stuffing





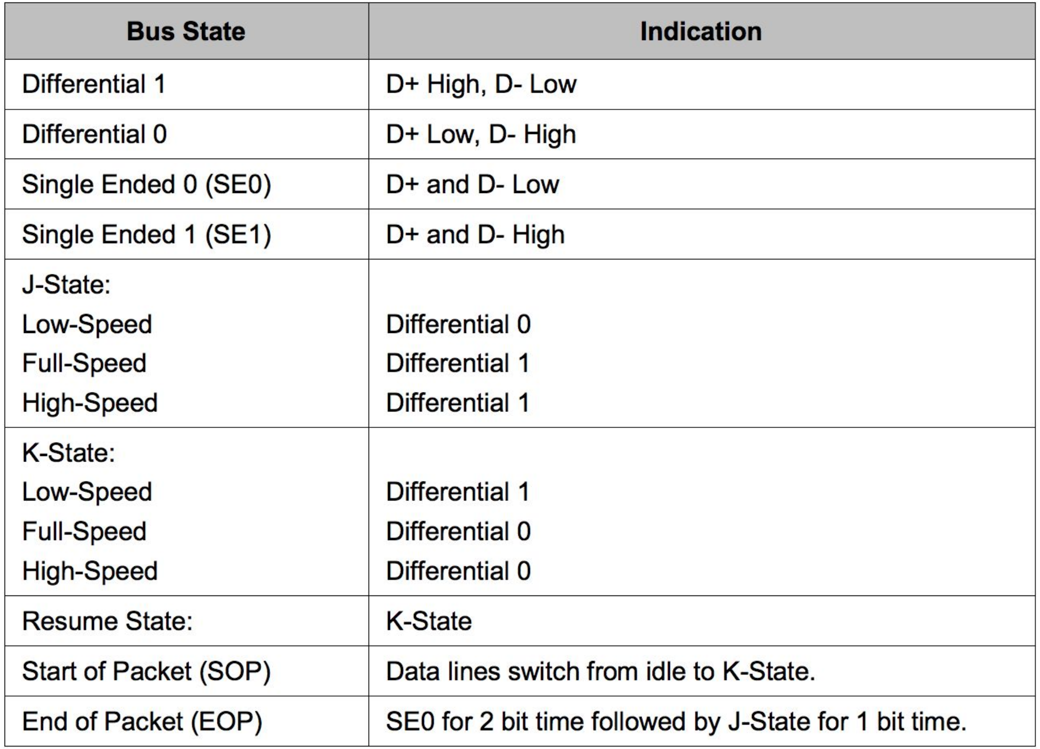
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB D+ and D- Communication



"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Communication States

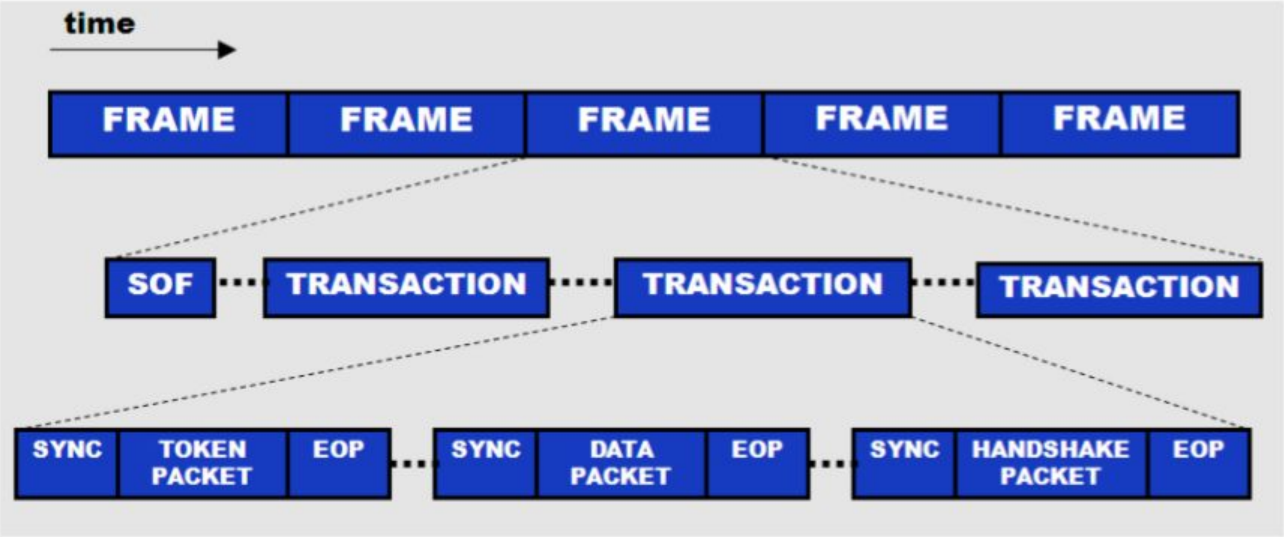


Bus State	Indication
Differential 1	D+ High, D- Low
Differential 0	D+ Low, D- High
Single Ended 0 (SE0)	D+ and D- Low
Single Ended 1 (SE1)	D+ and D- High
J-State: Low-Speed Full-Speed High-Speed	 Differential 0 Differential 1 Differential 1
K-State: Low-Speed Full-Speed High-Speed	 Differential 1 Differential 0 Differential 0
Resume State:	K-State
Start of Packet (SOP)	Data lines switch from idle to K-State.
End of Packet (EOP)	SE0 for 2 bit time followed by J-State for 1 bit time.

"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

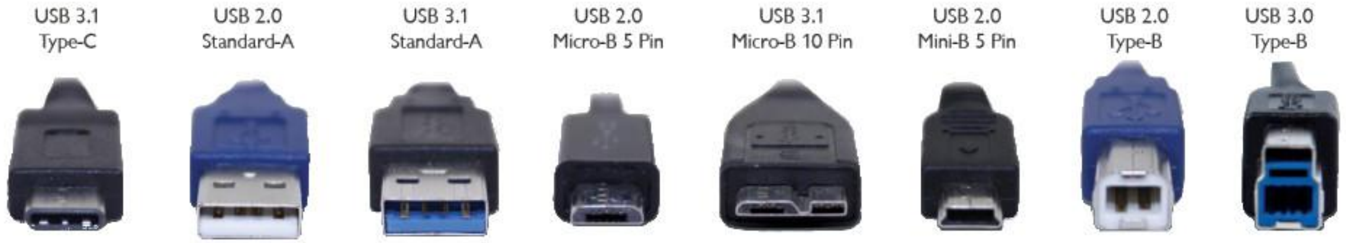
Demo: Sniffing USB with a Logic Analyzer

USB Protocol



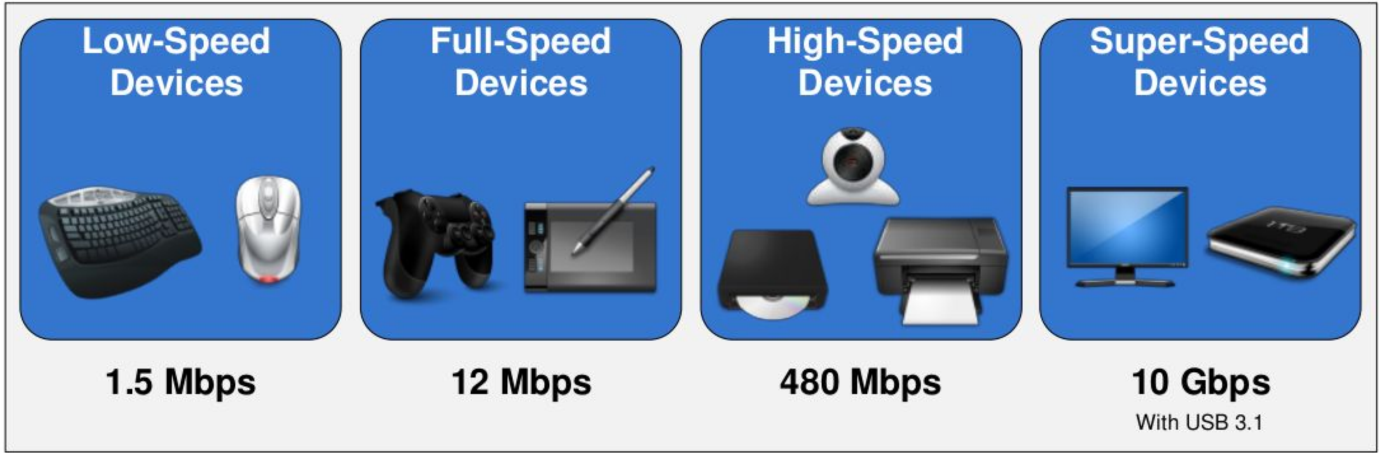
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Connectors



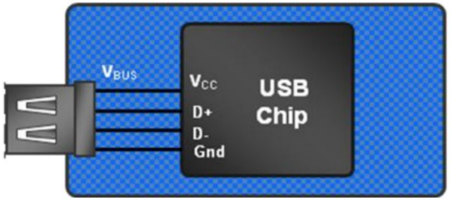
https://stroom.live/blog/index.php/tag/micro-usb/

USB Transfer Speeds

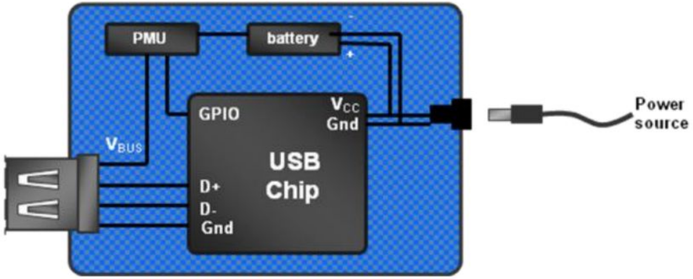


"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

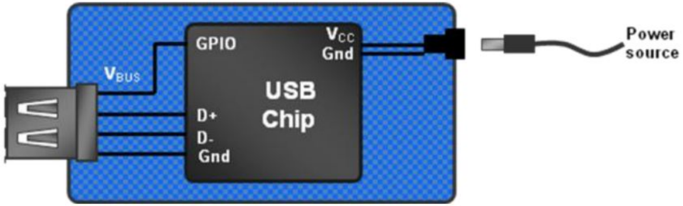
USB Power



bus-powered



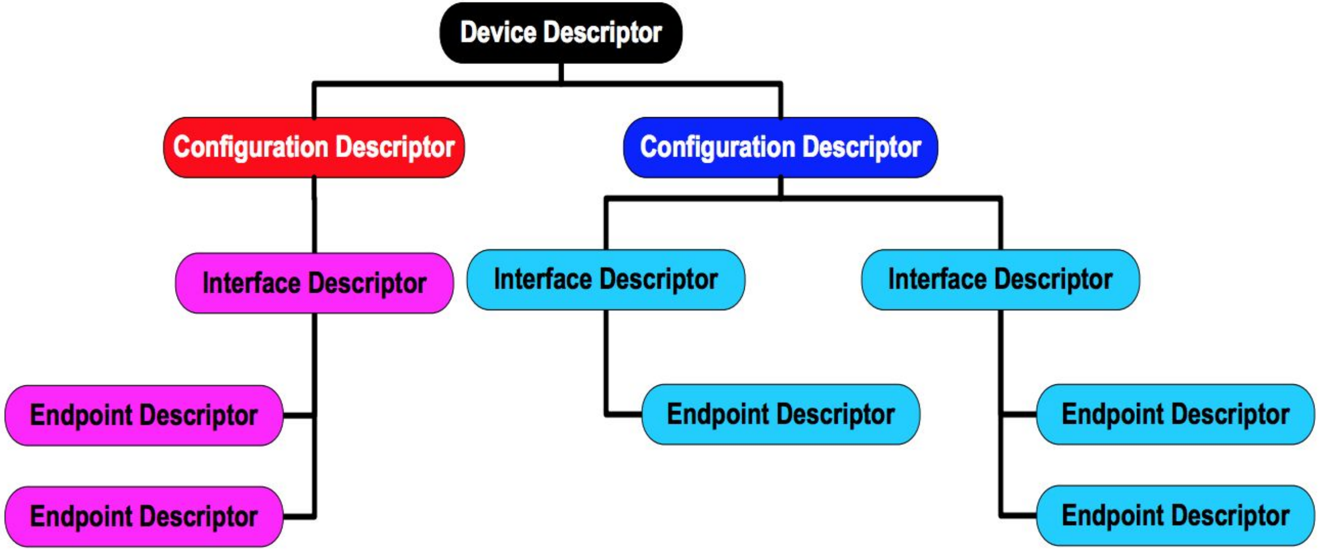
hybrid powered



self-powered

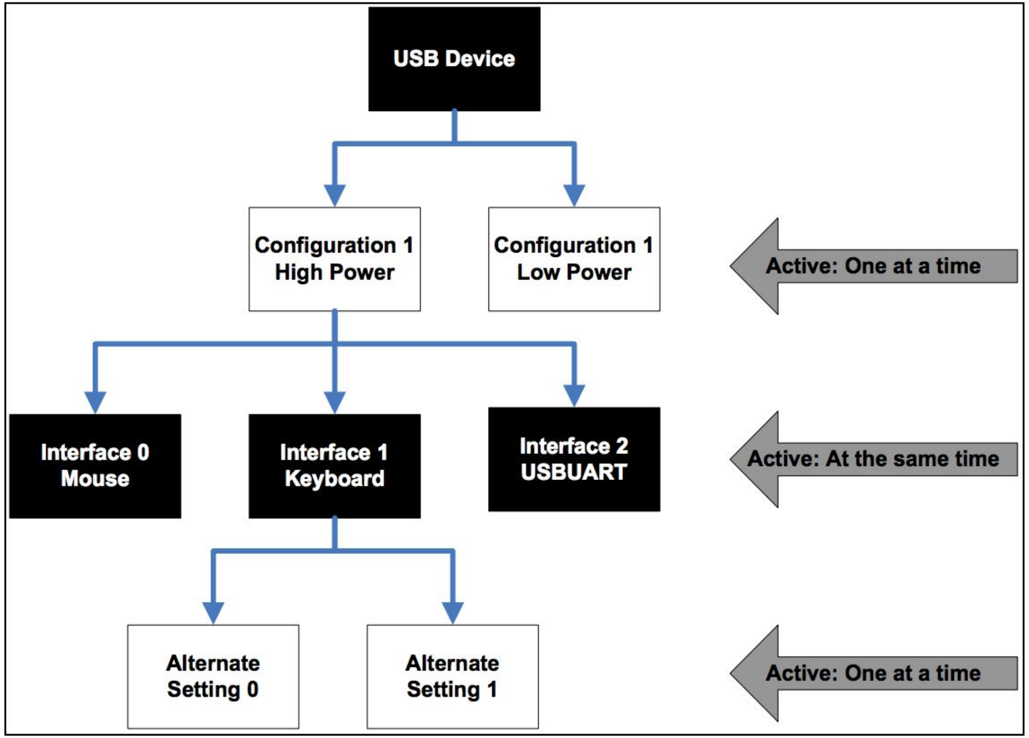
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Device Descriptor



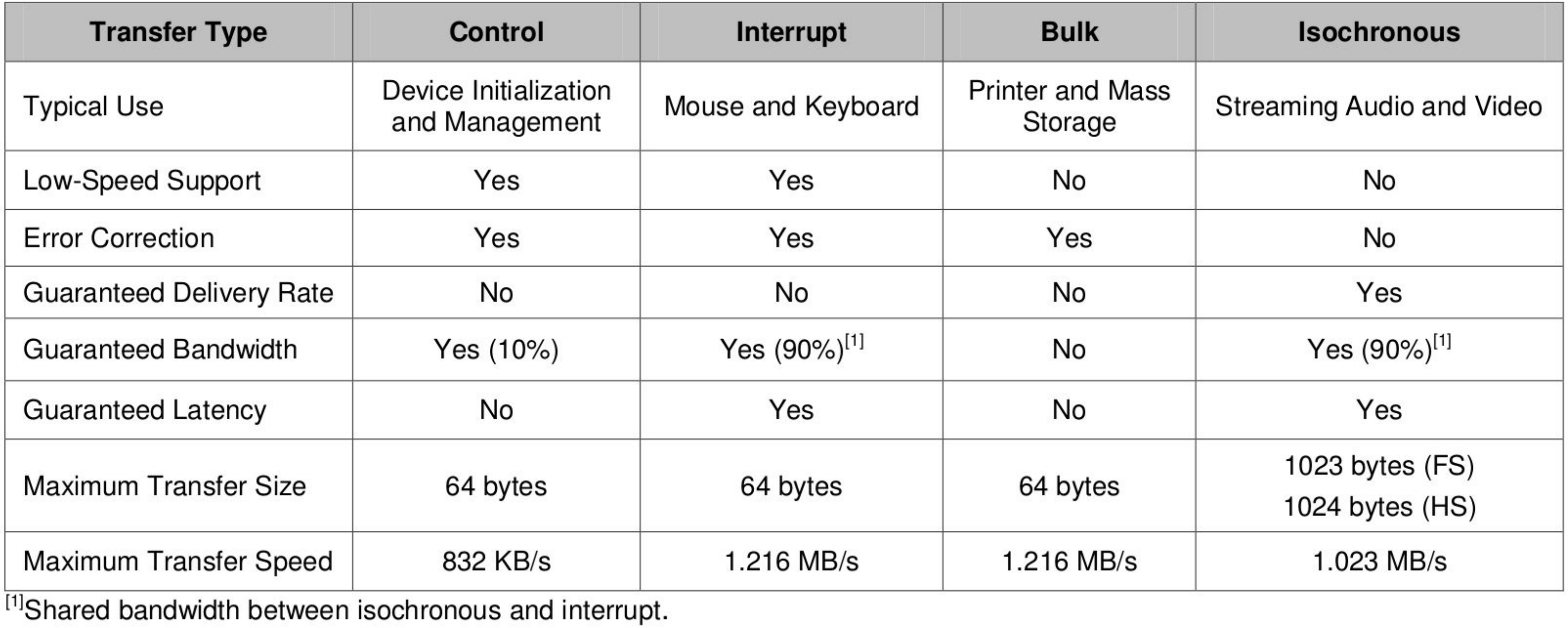
"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Device Descriptor: Example



"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Endpoint Types

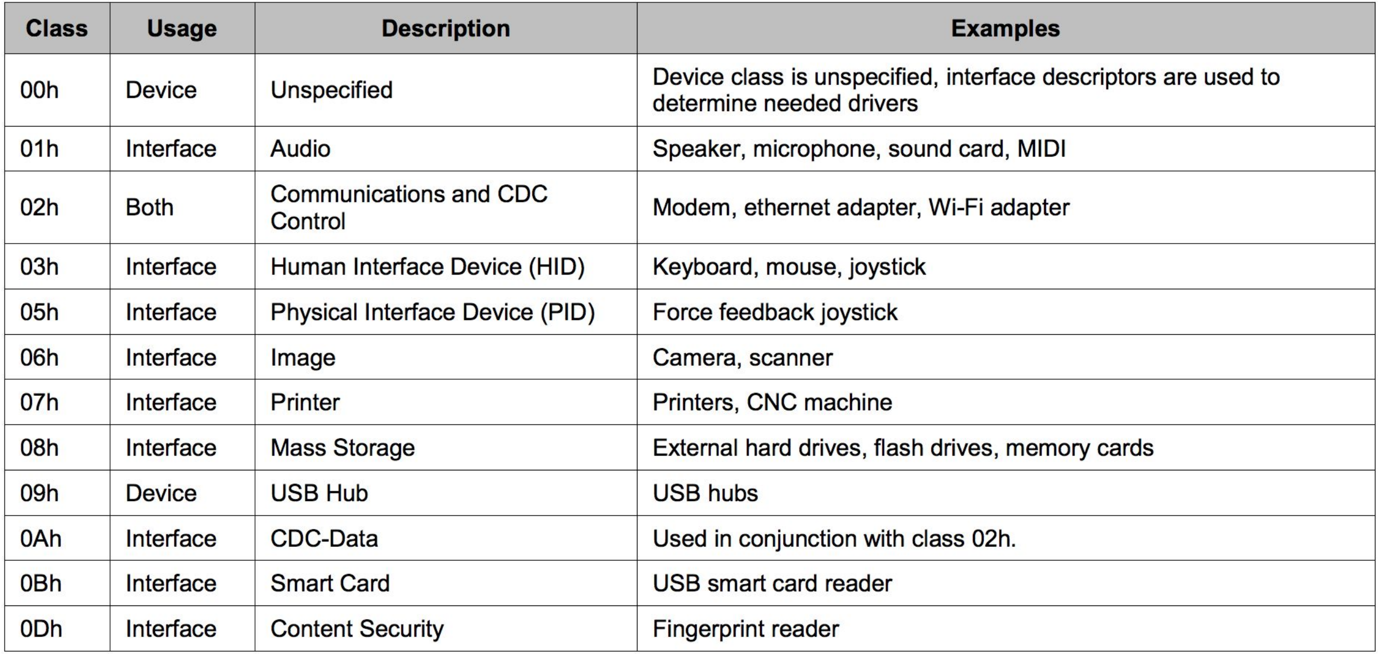


Transfer Type	Control	Interrupt	Bulk	Isochronous
Typical Use	Device Initialization and Management	Mouse and Keyboard	Printer and Mass Storage	Streaming Audio and Video
Low-Speed Support	Yes	Yes	No	No
Error Correction	Yes	Yes	Yes	No
Guaranteed Delivery Rate	No	No	No	Yes
Guaranteed Bandwidth	Yes (10%)	Yes (90%)[1]	No	Yes (90%)[1]
Guaranteed Latency	No	Yes	No	Yes
Maximum Transfer Size	64 bytes	64 bytes	64 bytes	1023 bytes (FS) 1024 bytes (HS)
Maximum Transfer Speed	832 KB/s	1.216 MB/s	1.216 MB/s	1.023 MB/s

[1]Shared bandwidth between isochronous and interrupt.

"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Class Codes



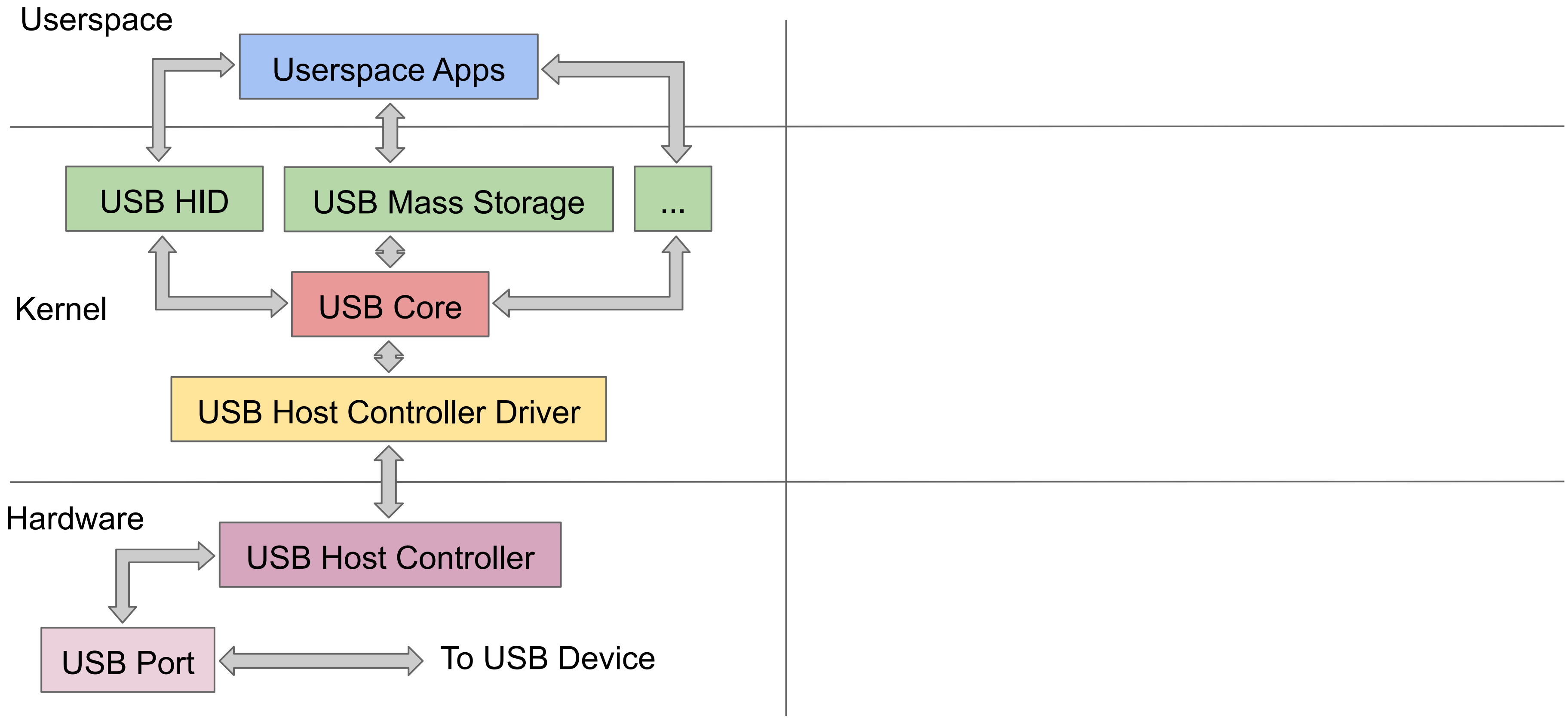
Class	Usage	Description	Examples
00h	Device	Unspecified	Device class is unspecified, interface descriptors are used to determine needed drivers
01h	Interface	Audio	Speaker, microphone, sound card, MIDI
02h	Both	Communications and CDC Control	Modem, ethernet adapter, Wi-Fi adapter
03h	Interface	Human Interface Device (HID)	Keyboard, mouse, joystick
05h	Interface	Physical Interface Device (PID)	Force feedback joystick
06h	Interface	Image	Camera, scanner
07h	Interface	Printer	Printers, CNC machine
08h	Interface	Mass Storage	External hard drives, flash drives, memory cards
09h	Device	USB Hub	USB hubs
0Ah	Interface	CDC-Data	Used in conjunction with class 02h.
0Bh	Interface	Smart Card	USB smart card reader
0Dh	Interface	Content Security	Fingerprint reader

and even more ...

"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

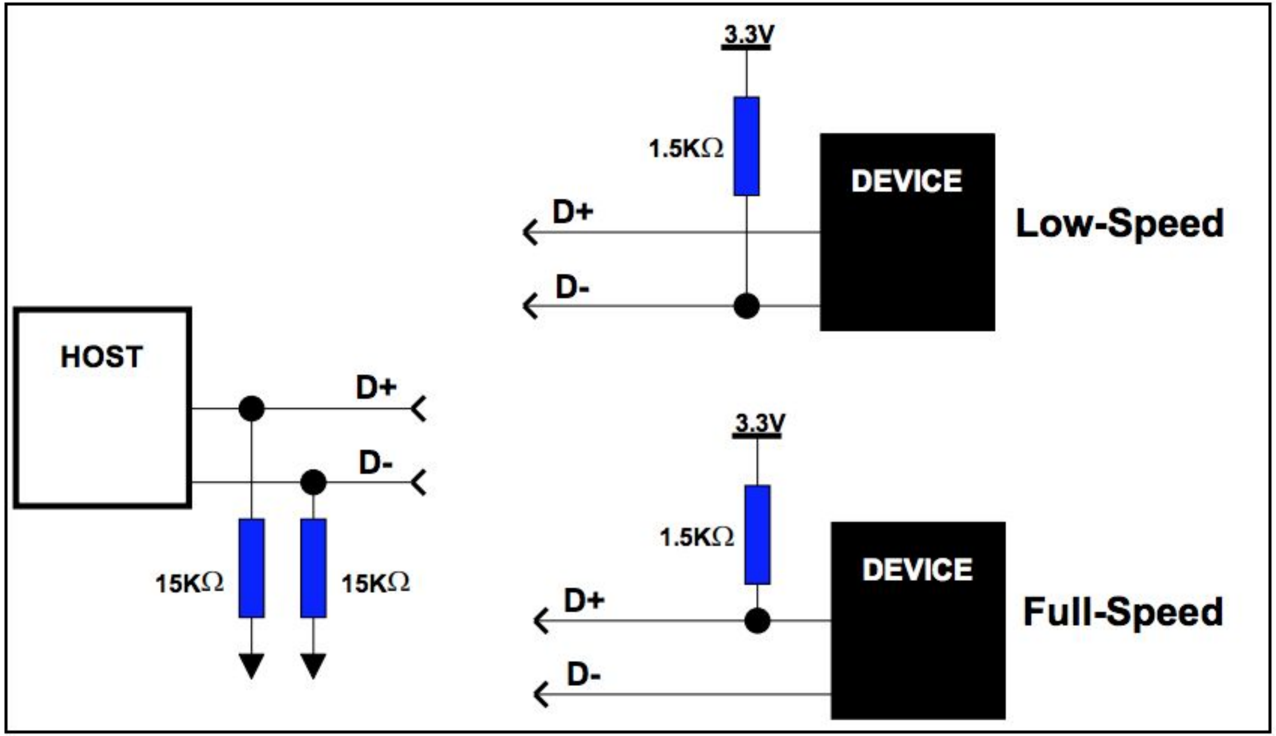
Demo: lsusb and syslog

USB Host



Userspace
Userspace Apps
USB HID
USB Mass Storage
...
Kernel
USB Core
USB Host Controller Driver
Hardware
USB Host Controller
USB Port
To USB Device

USB Speed Detection



"USB 101: An Introduction to Universal Serial Bus 2.0" by Robert Murphy

USB Enumeration (simplified)

1. The device is plugged into a USB port
2. The host requests device descriptors
3. The host loads the appropriate driver
4. The host sets a specific device configuration
5. Done

USB Communication

- Host communicates with the device through endpoints
- Enumeration happens through bidirectional control endpoint #0
- Data requests typically go through unidirectional endpoints #1, #2, ...
- All communication is initiated by host

Demo: Sniffing USB with usbmon and Wireshark

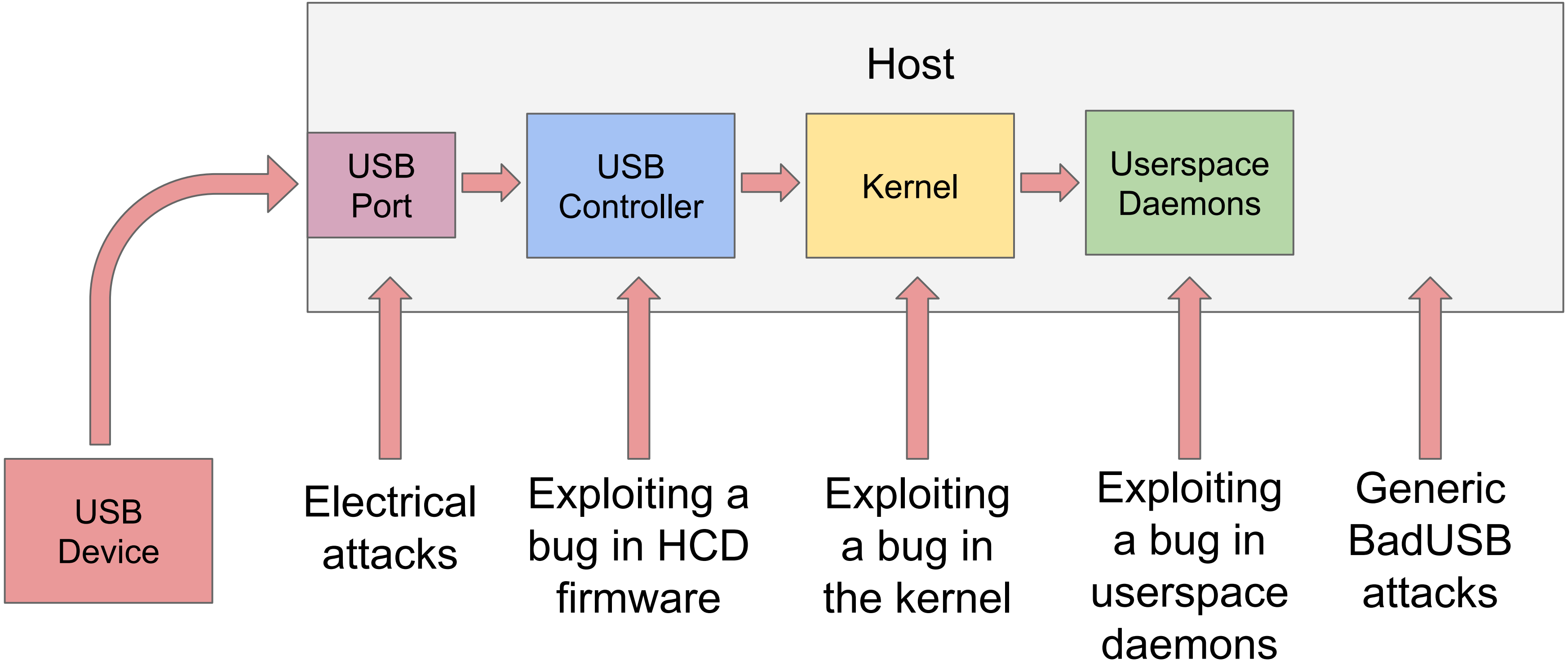
Demo: Turning off LED on a Logitech Web Camera

Part 2: USB Attack Surface

USB Attack Surface

- Simple scenarios:
  - Rogue device attacks host (Device => Host)
  - Rogue host attacks device (Host => Device)

Attack Surface: Device => Host



Host

USB Port

USB Controller

Kernel

Userspace Daemons

USB Device

Electrical attacks

Exploiting a bug in HCD firmware

Exploiting a bug in the kernel

Exploiting a bug in userspace daemons

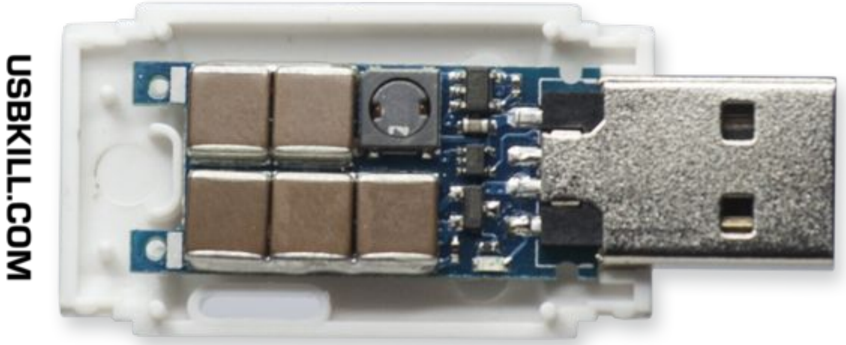
Generic BadUSB attacks

Device => Host: Examples

- Electrical attacks: USB Killer, see the following slides
- Exploiting a bug in UDC firmware: no known to me examples
- Exploiting a bug in the kernel: no known to me non-DOS examples
    - Related: HubCap: pwning the ChromeCast
- Exploiting a bug in userspace daemons
    - OATmeal on the Universal Cereal Bus: Exploiting Android phones over USB
- Generic BadUSB attacks: see the following slides

Electrical Attacks: USB Killer

- "When plugged into a device, the USB Killer rapidly charges its capacitors from the USB power lines. When the device is charged, -200VDC is discharged over the data lines of the host device"
- Available for 65$ at https://usbkill.com/products/usb-killer-v3



https://www.computerworld.com.sg/print-article/102264/

Attack Surface: Host => Device

- Reprogramming/updating device firmware by sending specific USB requests
  - Example: iSeeYou: Disabling the MacBook Webcam Indicator LED

- Exploiting a [memory corruption] bug in device firmware/software
  - Example: iPhone bootrom checkm8 exploit

- Attack surface depends on how complicated the device is

Host => Device => Host

- Hosts exploit a legitimate USB device and turns it into a malicious one
- The original BadUSB research by Karsten Nohl and Jakob Lell

- Why?
  - Spreading the attack
  - Escalating privileges
  - Breaking out of virtual machines

BadUSB

- Today the term BadUSB is used to refer to any kind of malicious USB device

- Examples:
  - BadUSB keyboard that looks like a flash drive
  - BadUSB Ethernet adapter that looks like flash drive

- Lots of consumer-ready BadUSB devices, lots of ways to make your own

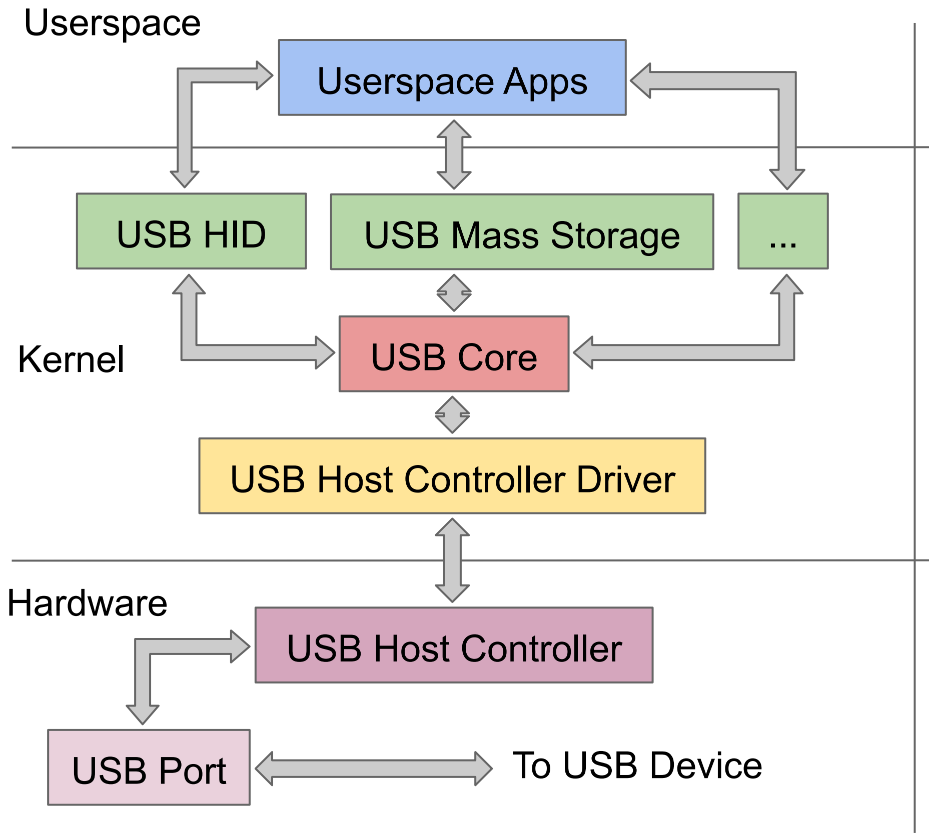
Remote USB Attacks

- USB/IP - USB device sharing system over IP network
- WebUSB - exposes USB device services to the web

- USBAnywhere - bugs that allow to remotely connect USB devices to Supermicro X9, X10 and X11 BMCs

Part 3: Linux USB Subsystem

Linux USB Subsystem



Communicating with USB Devices

- Linux kernel interface: usbfs
- C library: libusb
- Python wrapper: pyusb

Demo: Sending USB control requests with pyusb

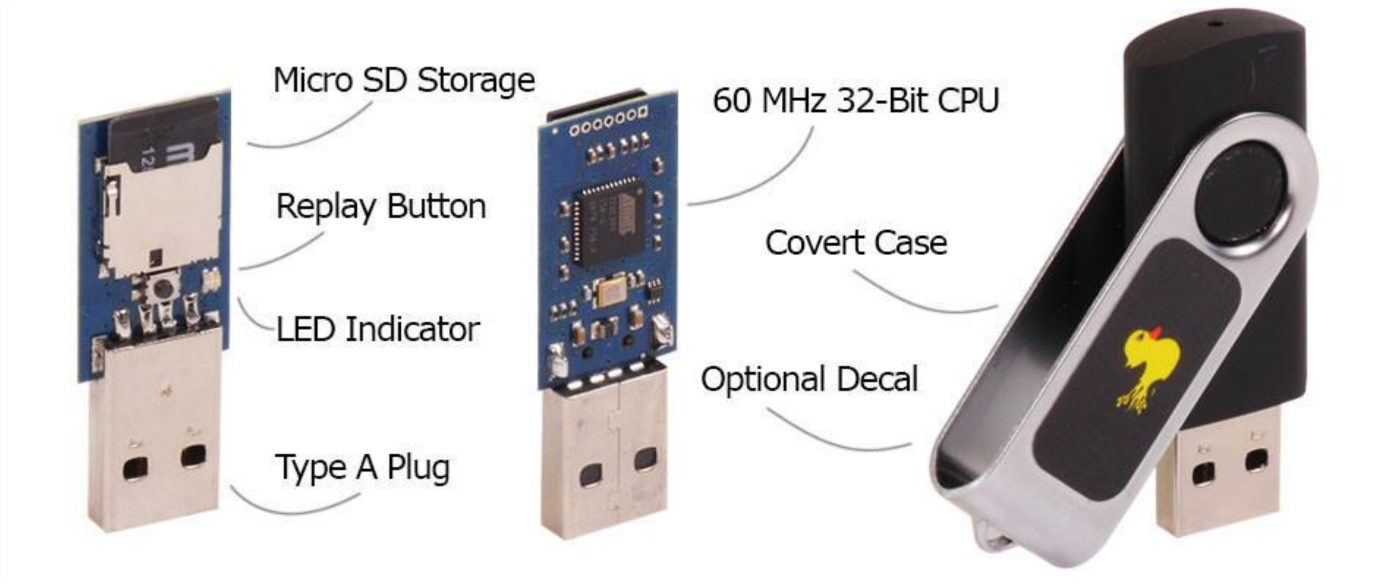
Part 4: BadUSB

Hardware

- Consumer-ready:
  - Rubber Ducky
  - Bash Bunny
  - Lan Turtle
- Microcontroller-based:
  - Teensy 3.2
  - ATtiny55 board
  - CJMCU BadUSB
  - Cactus WHID

Rubber Ducky

- "The USB Rubber Ducky is a keystroke injection tool disguised as a generic flash drive"



https://hakshop.com/products/usb-rubber-ducky-deluxe

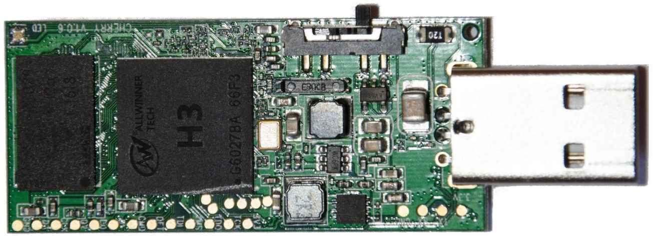
Rubber Ducky

- AT32UC3B1256 (32 bit AVR), more hardware specs)
- Uses its own language to describe keystroke payloads called Duckyscript
- Payloads are stored on a microSD card
- A lot of available payloads
- Price: 50$ (vs 1.3$ for ATtiny45 or 6$ for CJMCU BadUSB)

Demo: Rubber Ducky

Bash Bunny

- Bash Bunny, 60$, Quad-core ARM Cortex A7, more hardware specs
- "The Bash Bunny by Hak5 is ... USB attack platform. It delivers ... by emulating ... gigabit Ethernet, serial, flash storage and keyboards"
- Can emulate: HID, Ethernet, Serial, Mass Storage



https://hakshop.com/products/bash-bunny

https://forums.hak5.org/topic/40208-bash-bunny-specs/?tab=comments#comment-286703

Demo: Bash Bunny

Lan Turtle

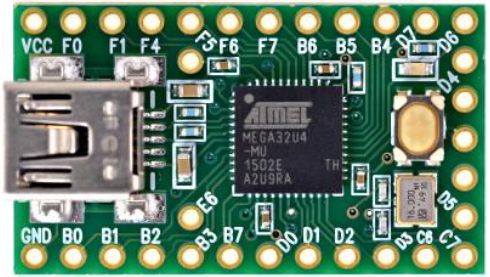
- Lan Turtle, 60$
- "USB Ethernet adapter with covert backdoors"
- There's an edition with a sim card, but never seen available for sale

https://www.pantomath.xyz/hak5-lan-turtle-just-plug-hack-within-5-seconds/

Teensy

- "The Teensy USB Development Board is a complete USB-based microcontroller development system"

Teensy 3.2


VCC F0 F1 F4 F5 F6 F7 B6 B5 B4 D7 D6
D4
ATMEL
MEGA32U4
-MU
1502E
A2U9RA
E6
D5
GND B0 B1 B2 B3 B7 D0 D1 D2 D3 C6 C7

Teensy 2.0

https://www.pjrc.com/store/teensy32.html https://www.pjrc.com/store/teensy.html

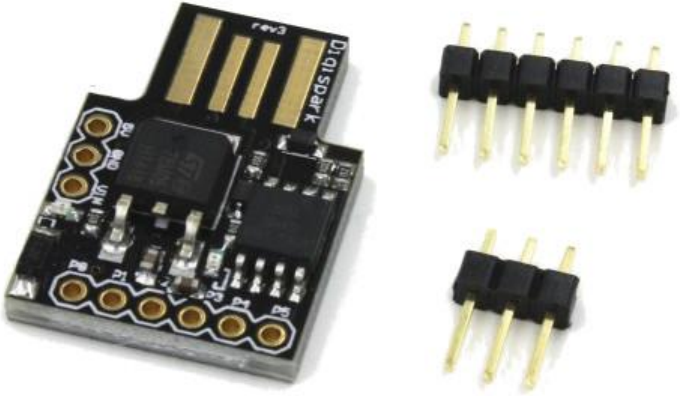
Teensy

- Teensy 2.0 based on 8 bit AVR 16 MHz (ATMEGA32U4), price: 16$
- Teensy 3.2 based on 32 bit ARM Cortex-M4 72 MHz (MK20DX256), price: 20$
- Can be programmed in C with Arduino Studio
- Has out-of-the-box support for emulating Serial, Keyboard, Mouse, Joystick, MIDI and Flight Sim USB devices
- Core libraries are open source

Demo: Teensy

ATTiny85 board

- ATTiny85 - keystroke injection tool, programmed with Arduino Studio
- Price: 1.3$

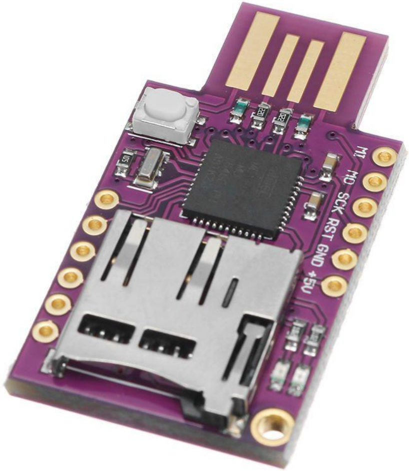


http://artofcircuits.com/product/digispark-attiny85-miniature-5v-usb-arduino-board

Demo: ATTiny85 board

CJMCU Virtual Keyboard

- CJMCU Virtual Keyboard - keystroke injection tool, programmed with Arduino Studio, executes Duckyscript from microSD card
- Price: 6$



https://www.pinterest.com/pin/751749362772345833/

Demo: CJMCU Virtual Keyboard

Cactus WHID

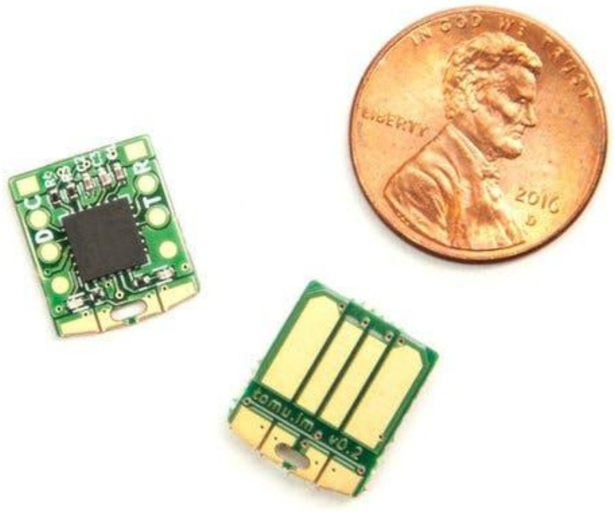
- Cactus WHID - keystroke injection tool, controlled over WiFi
- Price: 22$

https://www.aliexpress.com/item/Cactus-Micro-compatible-board-plus-WIFI-chip-esp8266-for-atmega32u4/32318391529.html

Demo: Cactus WHID

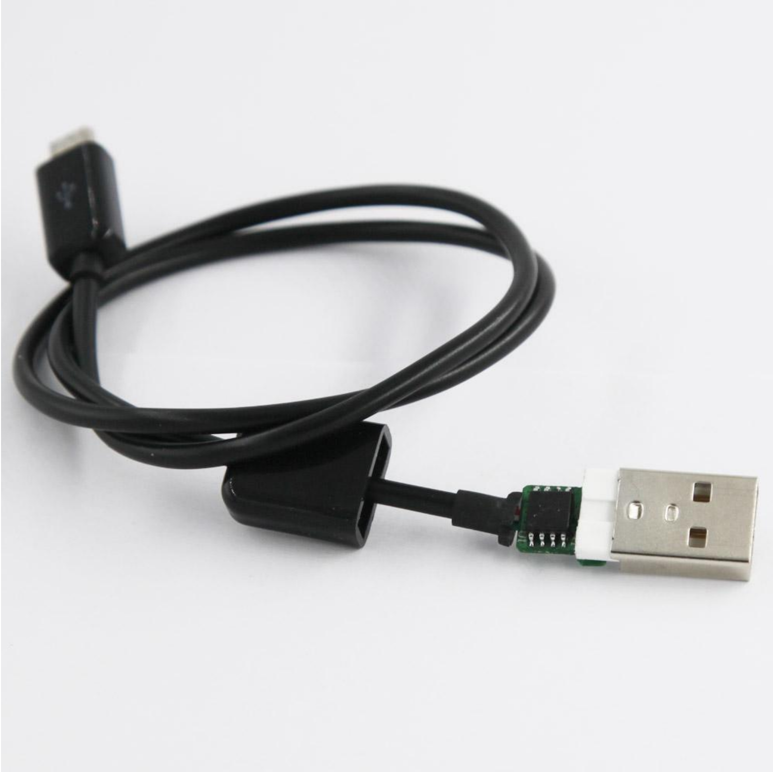
Tomu

- Tomu (12-30$) - "a group of tiny boards designed to fit inside your USB port"



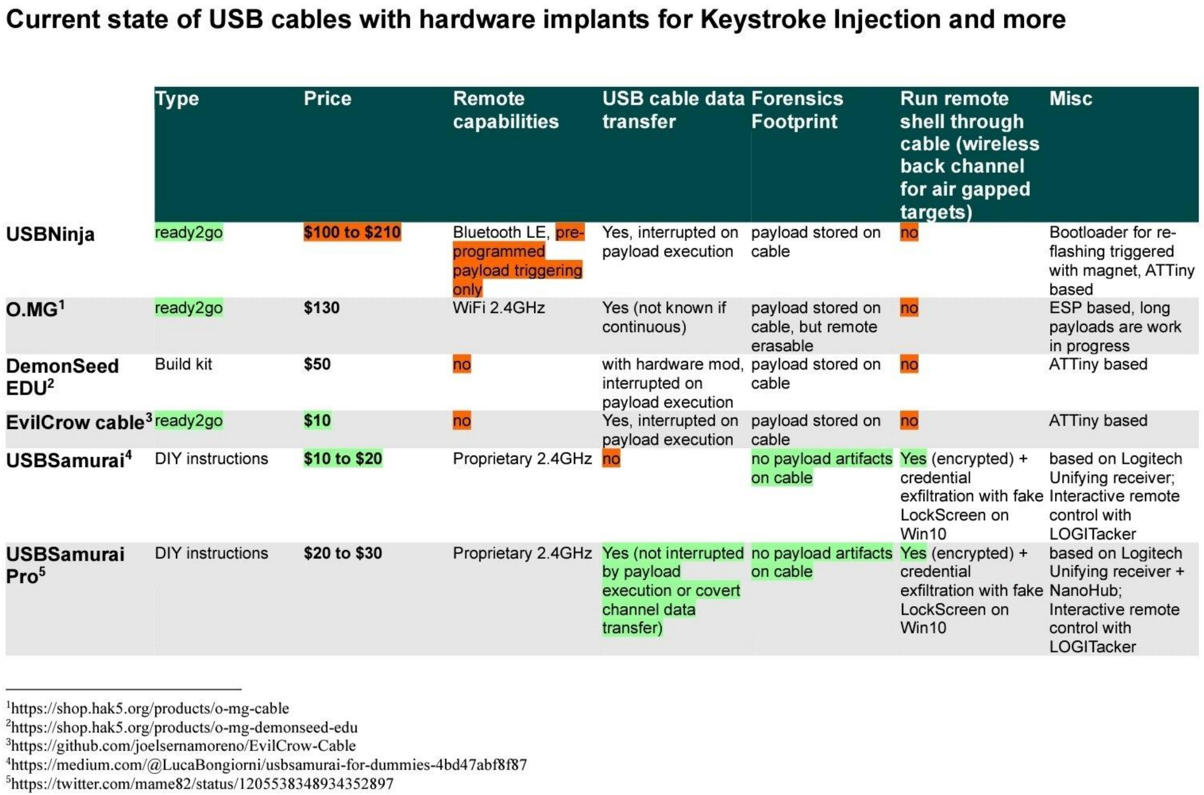
https://www.hackster.io/news/the-tomu-an-arm-microcontroller-that-fits-in-your-usb-port-31f60af97471

BadUSB Cables



https://hacker-gadgets.com/product/evil-crow-cable-badusb/

BadUSB Cables



Current state of USB cables with hardware implants for Keystroke Injection and more

	Type	Price	Remote capabilities	USB cable data transfer	Forensics Footprint	Run remote shell through cable (wireless back channel for air gapped targets)	Misc
USBNinja	ready2go	**$100 to $210**	Bluetooth LE, pre-programmed payload triggering only	Yes, interrupted on payload execution	payload stored on cable	no	Bootloader for re-flashing triggered with magnet, ATTiny based
O.MG[1]	ready2go	**$130**	WiFi 2.4GHz	Yes (not known if continuous)	payload stored on cable, but remote erasable	no	ESP based, long payloads are work in progress
DemonSeed EDU[2]	Build kit	**$50**	no	with hardware mod, interrupted on payload execution	payload stored on cable	no	ATTiny based
EvilCrow cable[3]	ready2go	**$10**	no	Yes, interrupted on payload execution	payload stored on cable	no	ATTiny based
USBSamurai[4]	DIY instructions	**$10 to $20**	Proprietary 2.4GHz	no	no payload artifacts on cable	Yes (encrypted) + credential exfiltration with fake LockScreen on Win10	based on Logitech Unifying receiver; Interactive remote control with LOGITacker
USBSamurai Pro[5]	DIY instructions	**$20 to $30**	Proprietary 2.4GHz	Yes (not interrupted by payload execution or covert channel data transfer)	no payload artifacts on cable	Yes (encrypted) + credential exfiltration with fake LockScreen on Win10	based on Logitech Unifying receiver + NanoHub; Interactive remote control with LOGITacker

[1] https://shop.hak5.org/products/o-mg-cable
[2] https://shop.hak5.org/products/o-mg-demonseed-edu
[3] https://github.com/joelsernamoreno/EvilCrow-Cable
[4] https://medium.com/@LucaBongiorni/usbsamurai-for-dummies-4bd47abf8f87
[5] https://twitter.com/mame82/status/1205538348934352897

https://mobile.twitter.com/mame82/status/1221093466463182849/photo/1

How Do I Make My Own BadUSB?

- Take and modify one of the shown BadUSB devices
- Use Facedancer (see the following part)
- Use a Linux-based board and USB Gadget (see the following part)

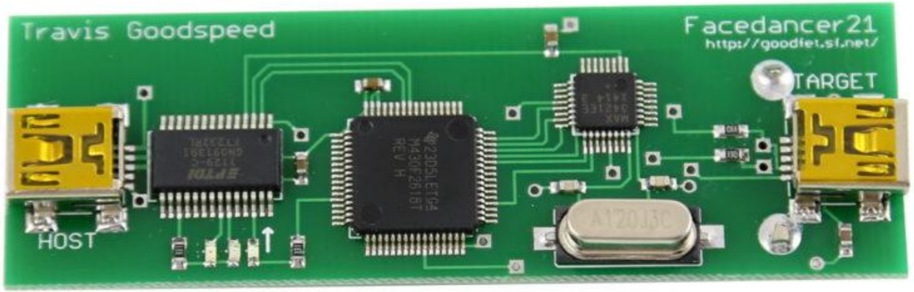
Part 5: Facedancer

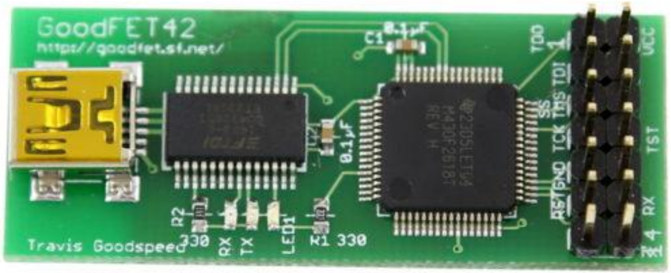
Facedancer

- “The purpose of this board is to allow USB devices to be written in host-side Python, so that one workstation can fuzz-test the USB device drivers of another host”
- Hardware:
  - Old: Facedancer21 (and GoodFET42)
  - New: GreatFET One
- Software:
  - Old: https://github.com/travisgoodspeed/goodfet
  - New: https://github.com/usb-tools/Facedancer

FaceDancer21 (and GoodFET42)

- Facedancer21, 85$, MSP430F2618TPM + MAX3421E + FT232RL



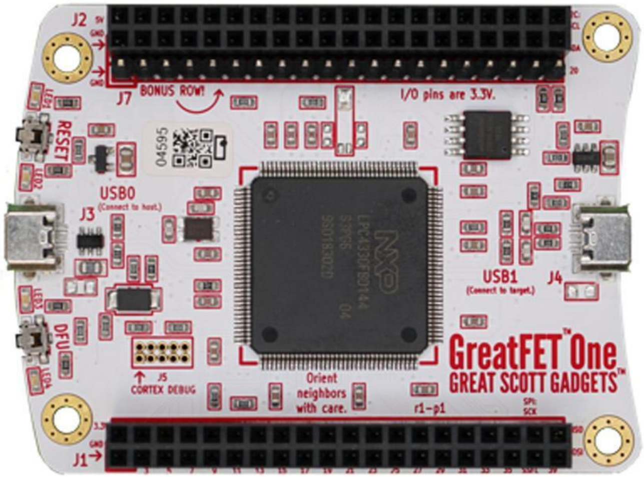


https://hackerwarehouse.com/product/facedancer21/

https://hackerwarehouse.com/product/goodfet42/

GreatFET One

- GreatFET One, 100$



https://shop.hak5.org/products/greatfet

Demo: Emulating USB Keyboard with Facedancer

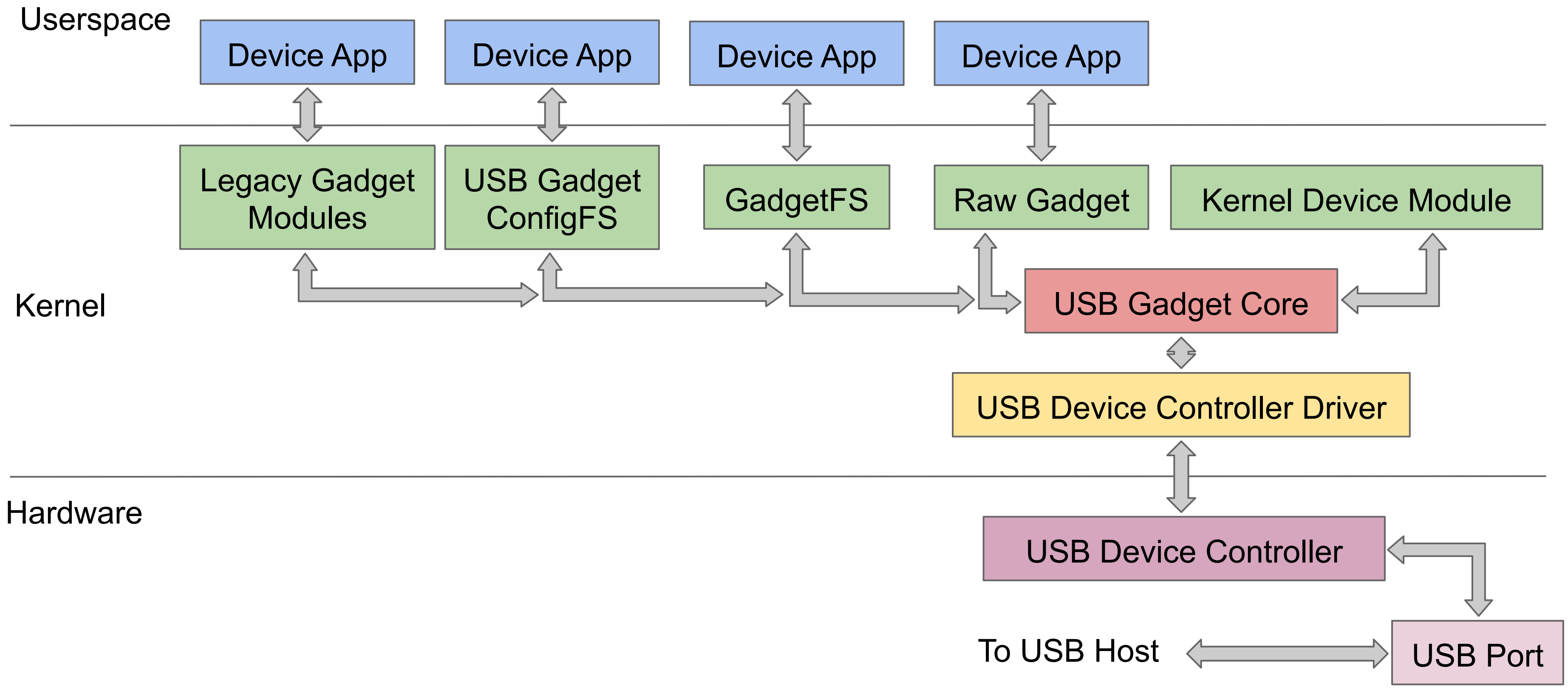
Demo: USB Reconnaissance with Facedancer

Part 6: Linux Gadget Subsystem

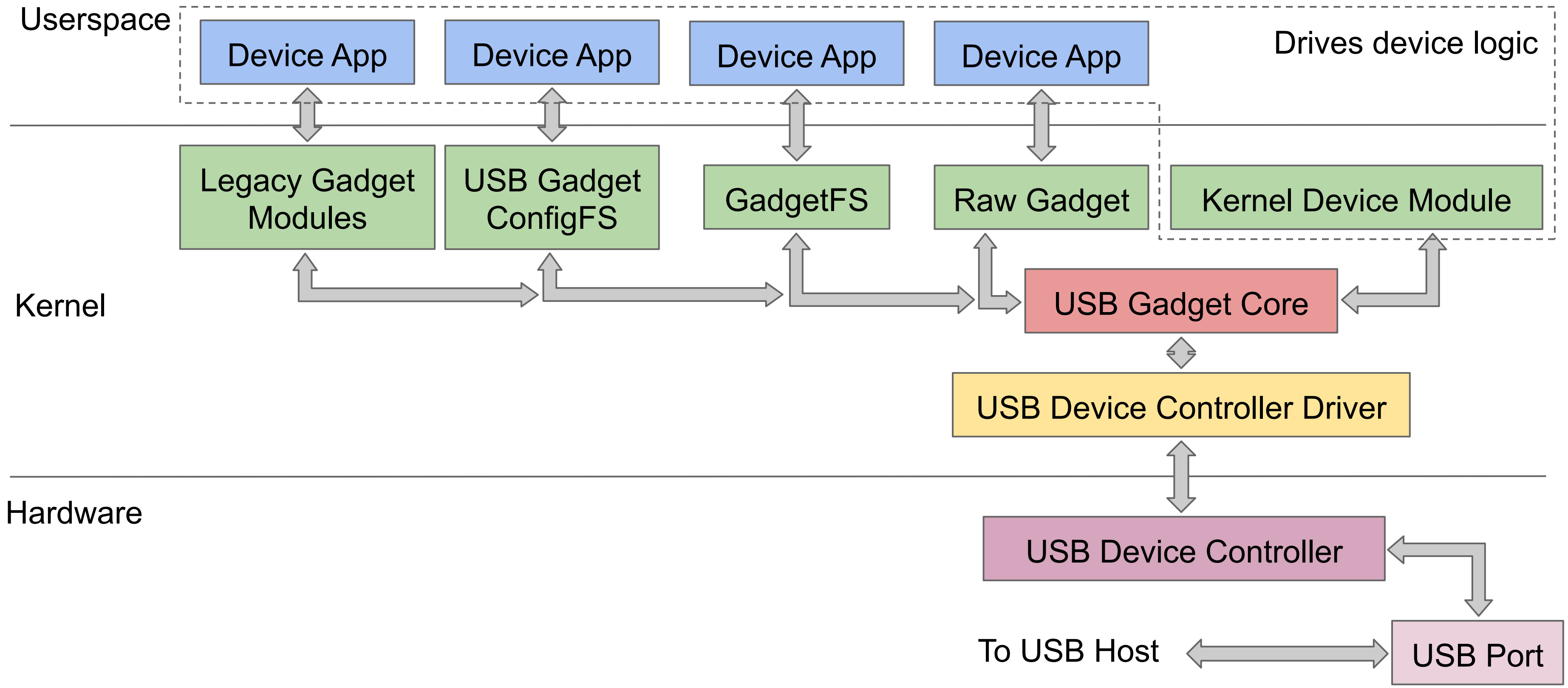
Linux USB Subsystem: Gadget

- Allows to turn a Linux-based device into a USB device
- Requires a USB Device Controller, USB OTG port and proper driver support
- Typically marketed as "USB OTG" in board description
- Linux kernel provides a few interfaces for the Gadget Subsystem

Linux USB Subsystem: Gadget



Linux USB Subsystem: Gadget

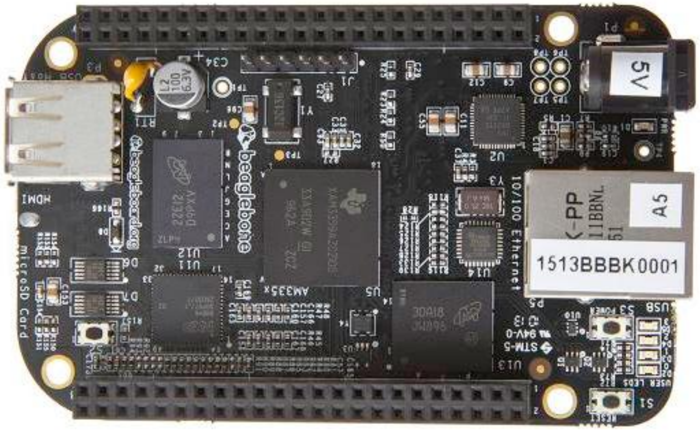


Linux Gadget Hardware

- Almost any Linux-based single-board-computer with a USB Device Controller, USB OTG port and proper driver support, examples:
  - Beagle boards (BeagleBone Black)
  - Odroid boards (ODROID-XU3, ODROID-C2)
  - USB Armory
  - Nexus/Pixel Android devices
  - Raspberry Pi Zero boards (v1.3, W)
- Connect USB Device Controller over PCI/PCIe to a PC
  - EC3380-AB

BeagleBone Black

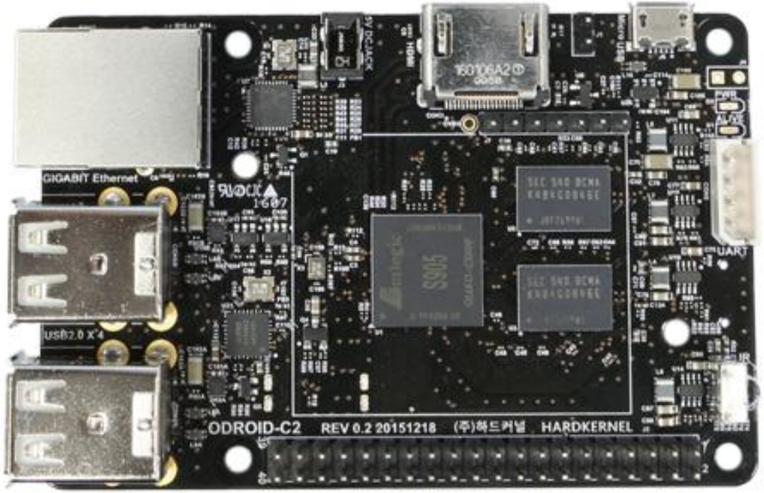
- BeagleBone Black, 50$
- Open Hardware, AM335x ARM Cortex-A8, 512MB RAM, USB OTG



https://beagleboard.org/static/ti/product_detail_black_lg.jpg

ODROIDs

- ODROID-XU3: discontinued, replaced by ODROID-XU4 without OTG
- ODROID-C2: 50$, USB OTG



https://www.hardkernel.com/main/_Files/prdt/2016/201602/C2BoardRevision2.jpg

USB Armory

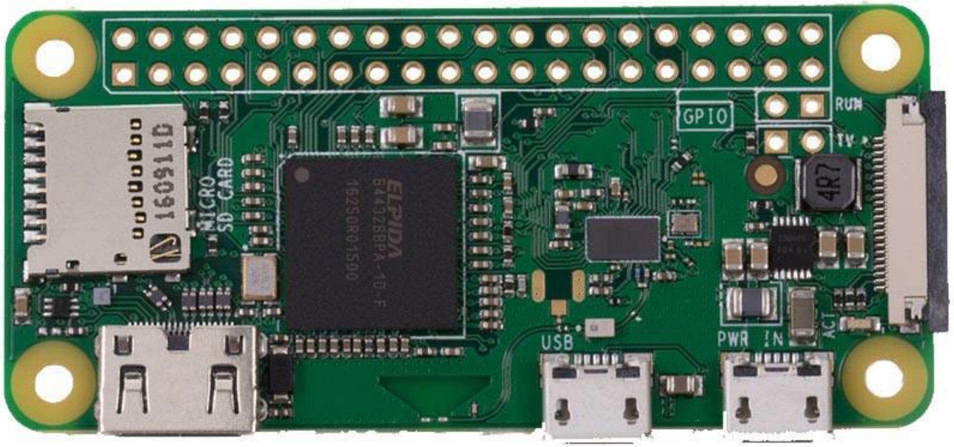
- USB Armory, 150$
- Open Hardware, NXP i.MX6UL(Z) ARM Cortex-A7, 512MB RAM, USB OTG

https://inversepath.com/usbarmory.html

Android

- Some Nexus/Pixel/… Android devices have OTG/Gadget support
- Kali Linux NetHunter supports some BadUSB attacks [1], [2]

Raspberry Pi Zero



https://thepihut.com/products/raspberry-pi-zero-w

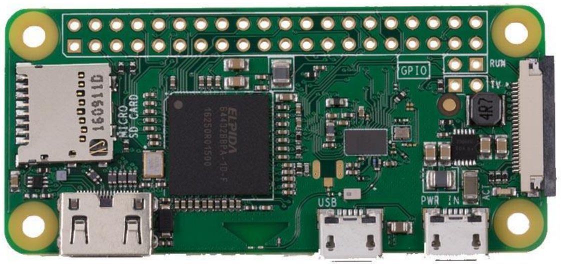
Raspberry Pi Zero

- Raspberry Pi Zero v1.3 (10$)
  - BCM2835, 512MB RAM, USB OTG

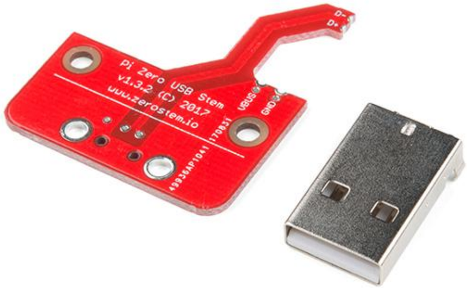
- Raspberry Pi Zero W (15$)
  - 802.11 b/g/n wireless LAN
  - Bluetooth 4.1, Low Energy (BLE)

- P4wnP1: USB attack platform for RPi Zero

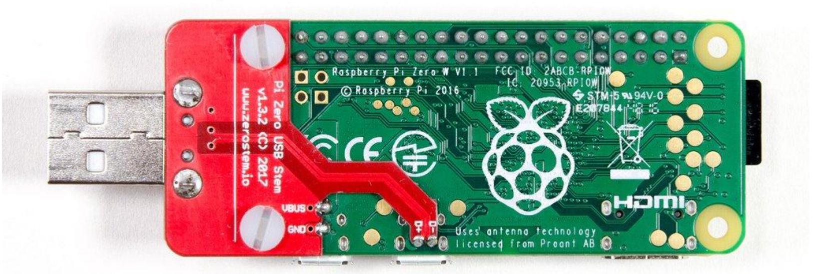
Raspberry Pi Zero W + Zero Stem



+



=



https://thepihut.com/products/raspberry-pi-zero-w

https://www.sparkfun.com/products/14526

https://shop.pimoroni.com/products/zero-stem-usb-otg-connector

EC3380-AB

- EC3380-AB (178$): ExpressCard/34 USB Device Controller
- Works with mainline net2280 driver, supports SuperSpeed
- Requires adapters to connect to modern PCs (e.g. over Thunderbolt)

http://www.hwtools.net/Adapter/EC3380-AB.html

Sonnet Echo ExpressCard Pro

- Sonnet Echo ExpressCard Pro (170$): Thunderbolt 2 => ExpressCard/34

https://www.bpm-media.de/produkte/post-it/zubehoer/adapter/sonnet-echo-expresscard/34-thunderbolt-adapter/

Apple T3 to T2 Adapter

- Apple T3 to T2 Adapter (61$): Thunderbolt 3 => Thunderbolt 2

https://www.apple.com/de/shop/product/MMEL2ZM/A/thunderbolt-3-usb%E2%80%91c-auf-thunderbolt-2-adapter

Legacy Gadget Modules

- Loadable kernel modules that emulate USB devices of particular classes
- Nowadays these modules are based on the Composite Gadget Framework
- Examples:
  - `g_hid.ko` - HID
  - `g_mass_storage.ko` - Mass Storage
  - `g_ether.ko` - Ethernet
  - ...
  - `g_multi.ko` - combines multiple legacy modules
  - `g_ffs.ko` - FunctionFS

USB Gadget ConfigFS

- Allows to compose multiple USB functions (USB classes) into a single USB device
- Basically a more convenient replacement for Legacy Gadget Modules
- Filesystem based interface
- libusbg - USB Gadget ConfigFS wrapper library

GadgetFS

- GadgetFS - allows to implement USB device logic in a userspace app
- Allows to emulate (almost) arbitrary USB devices
- Filesystem based interface
- The interface allows to receive USB messages sent to the device and reply when necessary

Raw Gadget

- Similar to GadgetFS, but
  - All USB requests forwarded to userspace
  - No sanity checking on USB descriptors
  - See more differences here
- Just merged into mainline in 5.7-rc1

- github.com/xairy/raw-gadget

Custom Kernel Module

- The kernel provides internal API for creating USB gadgets
- Instead of using some pass-through interface from the userspace (GadgetFS, …) we can implement a custom kernel modules that uses this API
- Allows a very low level control of the content of USB messages (invalid descriptors, etc.)

Demo: Emulating USB Devices via USB Gadget Interfaces

Demo: Emulating USB Devices via EC3380-AB

Part 7: USB Fuzzing

USB Fuzzing Approaches

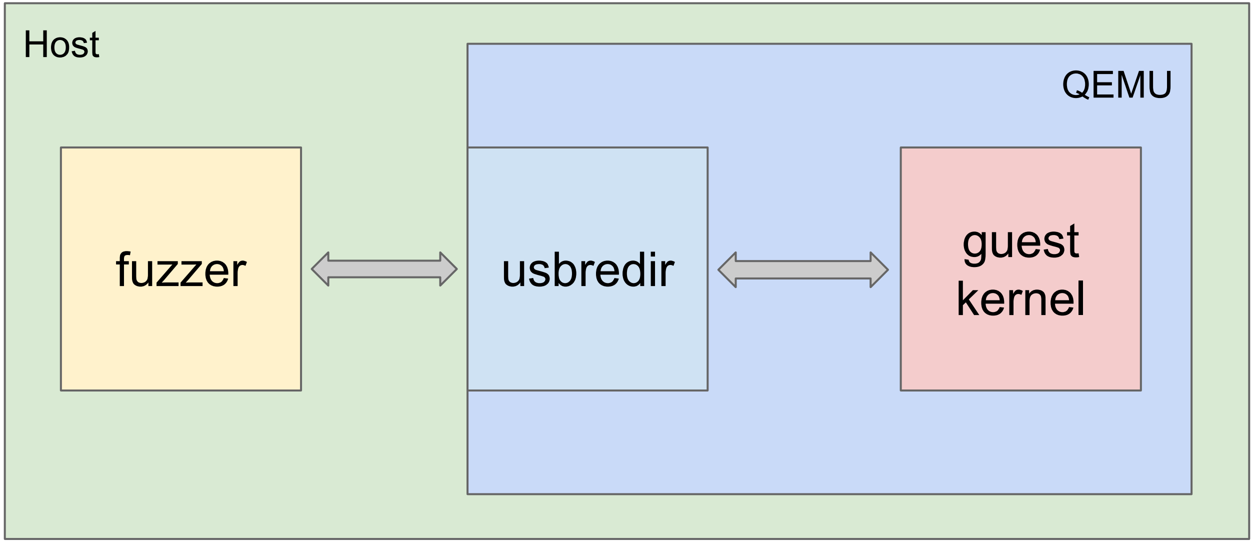
- Emulate USB devices via hardware
  - Plug in Facedancer into a USB host and use umap, umap2 or nu-map

- Emulate USB devices through a hypervisor
  - vUSBf fuzzes the guest kernel running in QEMU by connecting USB devices via usbredir protocol

- Emulate USB devices in the kernel
  - syzkaller can fuzz Linux USB stack

vUSBf

- Virtual USB Fuzzer - KVM/QEMU based USB-fuzzing framework
- Fuzzing the kernel running in QEMU via usbredir
- github.com/schumilo/vUSBf



CVE-2016-2384

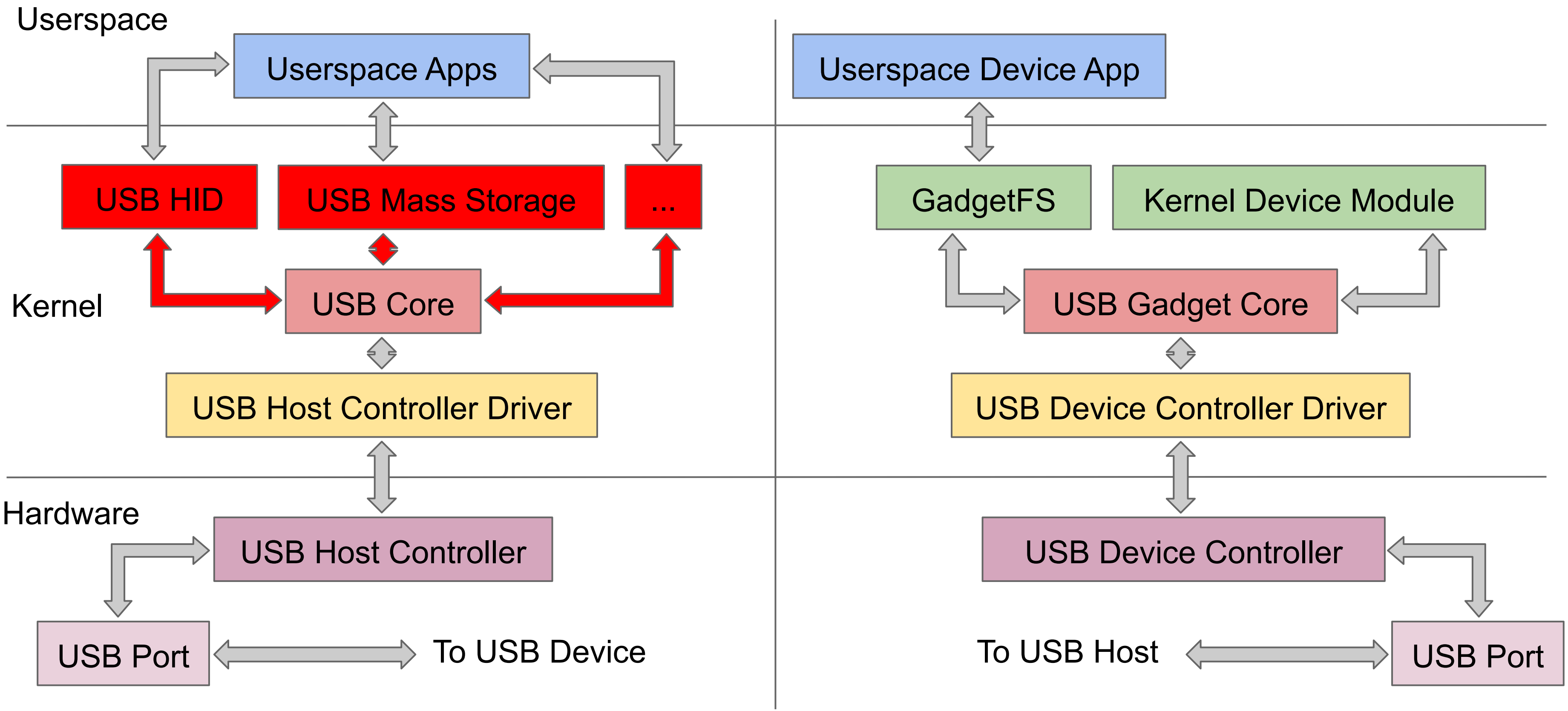
- Double-free in USB-MIDI Linux kernel driver
- Found with vUSBf
- Confirmed and exploited with Facedancer21
- https://xairy.github.io/blog/2016/cve-2016-2384

Syzkaller

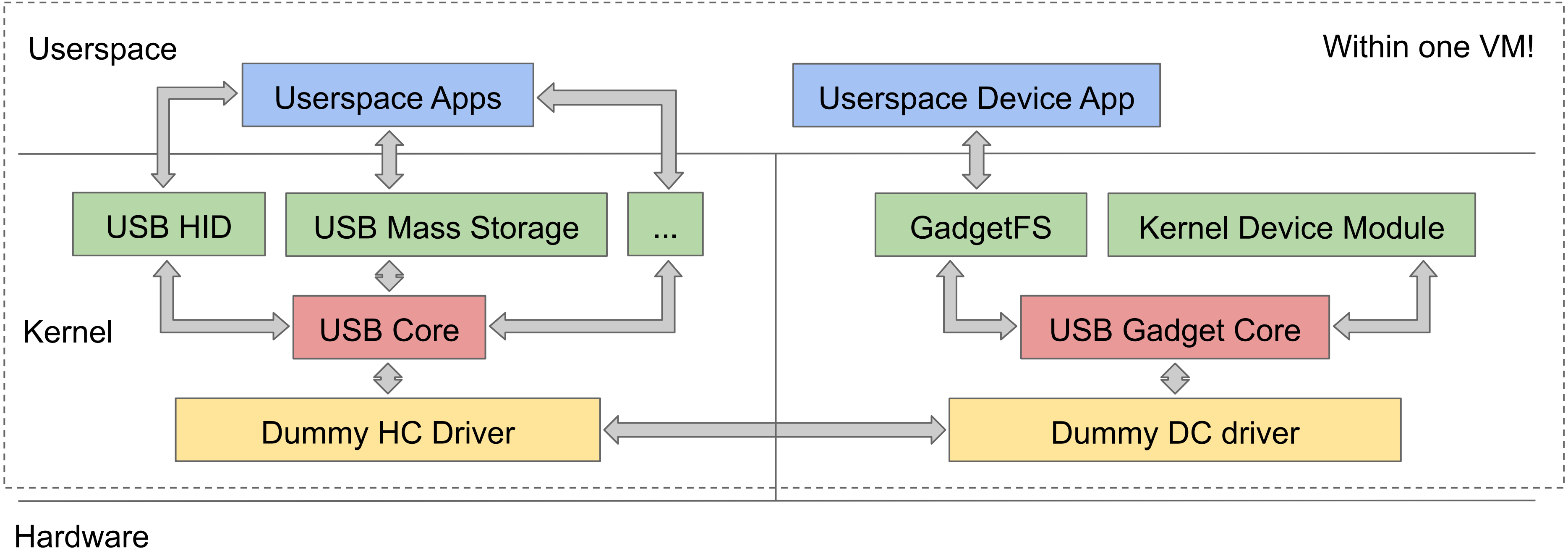
- Unsupervised coverage-guided grammar-based kernel fuzzer
- Mainly targets Linux kernel, but supports other OSes
- Found over 2500 bugs: syzkaller.appspot.com

- Supports "external" fuzzing of the Linux kernel USB subsystem
- 200+ bugs in the Linux USB subsystem

Linux USB Subsystem



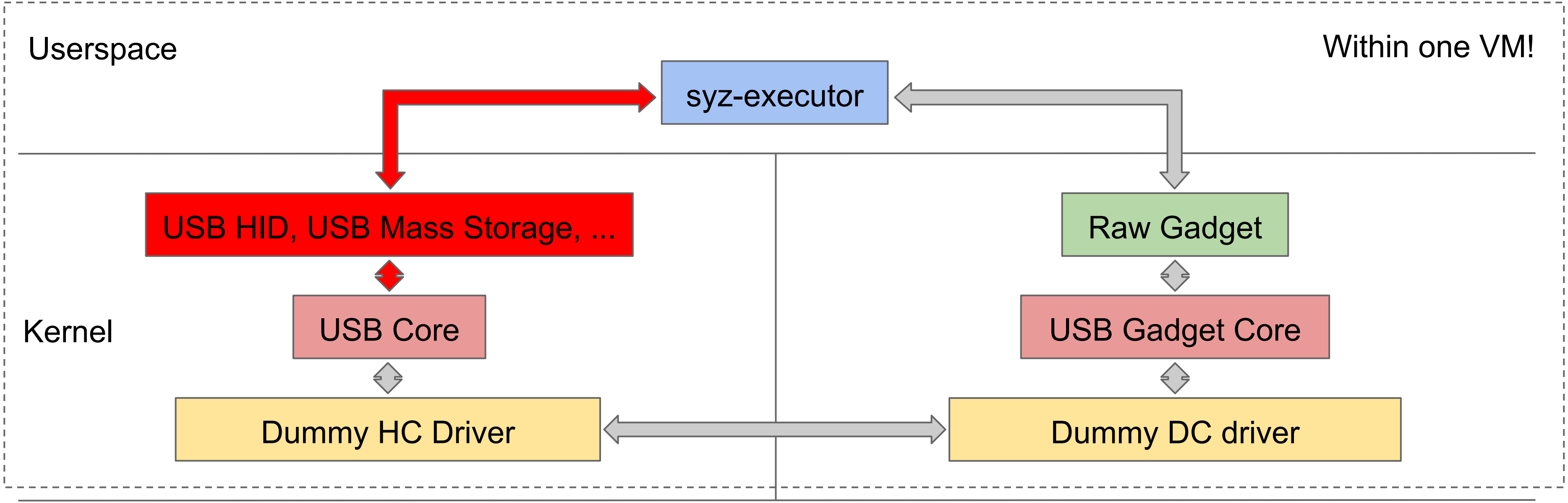
CONFIG_USB_DUMMY_HCD



No hardware (or hypervisors) required!

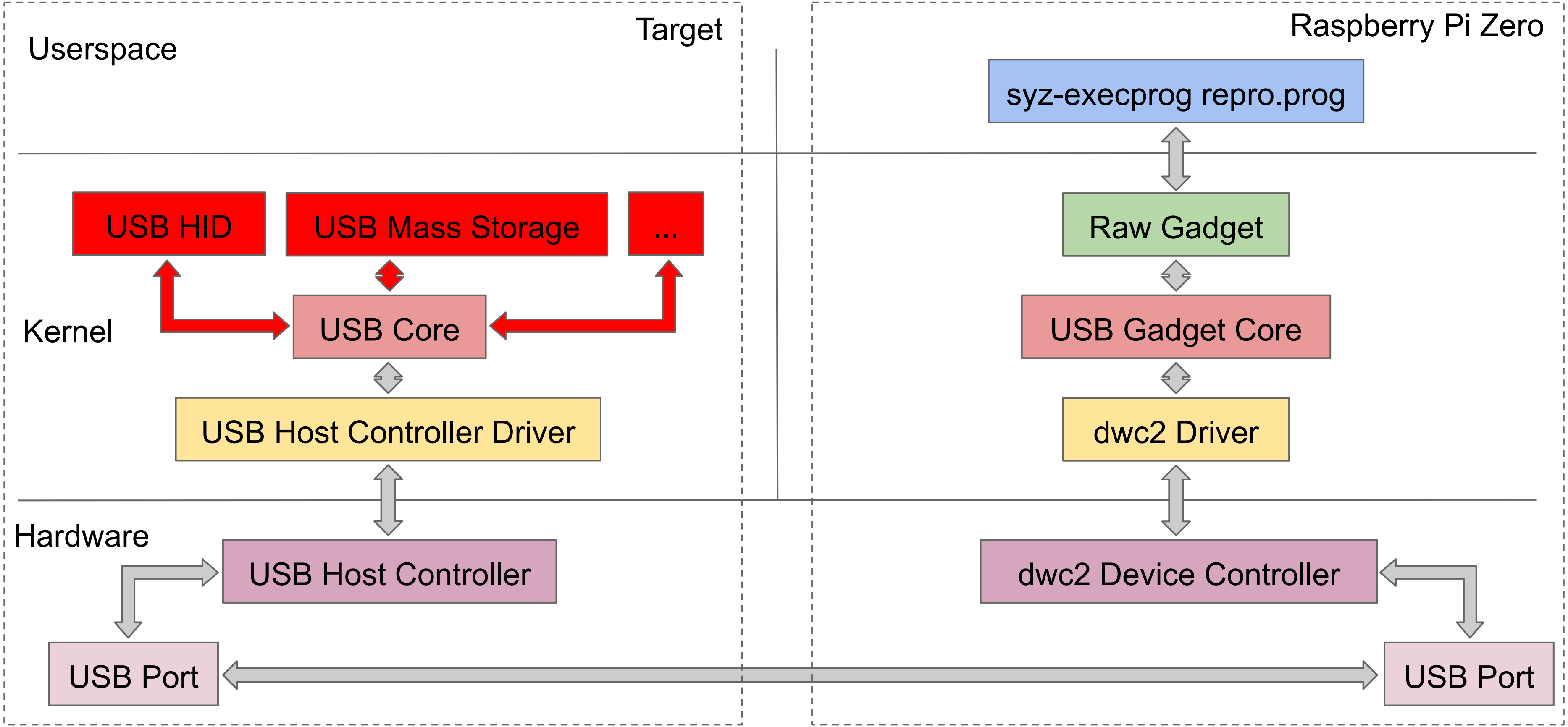
Demo: Dummy HCD/UDC

Syzkaller USB Fuzzing Approach



No hardware (or hypervisors) required!

Running Reproducers via Raspberry Pi Zero



Demo: Crashing Linux Over USB

Demo: Crashing Windows Over USB

Part 8: USB Sniffing

USB Sniffing

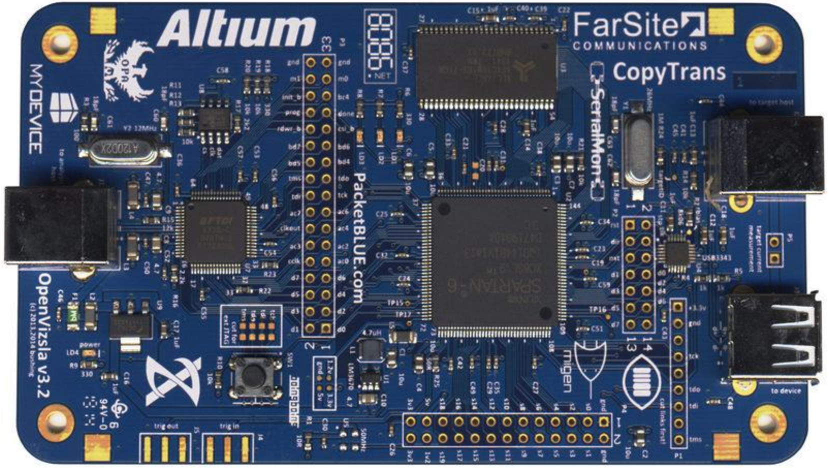
- (Besides usbmon and using a logic analyzer)

- Commercial USB sniffers
  - Beagle (475$ for FS, 1400$ for HS, 6000$ for SS)

- Open source USB sniffers
  - OpenVizsla - custom board with an FPGA
  - USBProxy Legacy - based on BeagleBone Black (or other Linux boards)
  - USBProxy 'Nouveau' - based on Facedancer

OpenVizsla

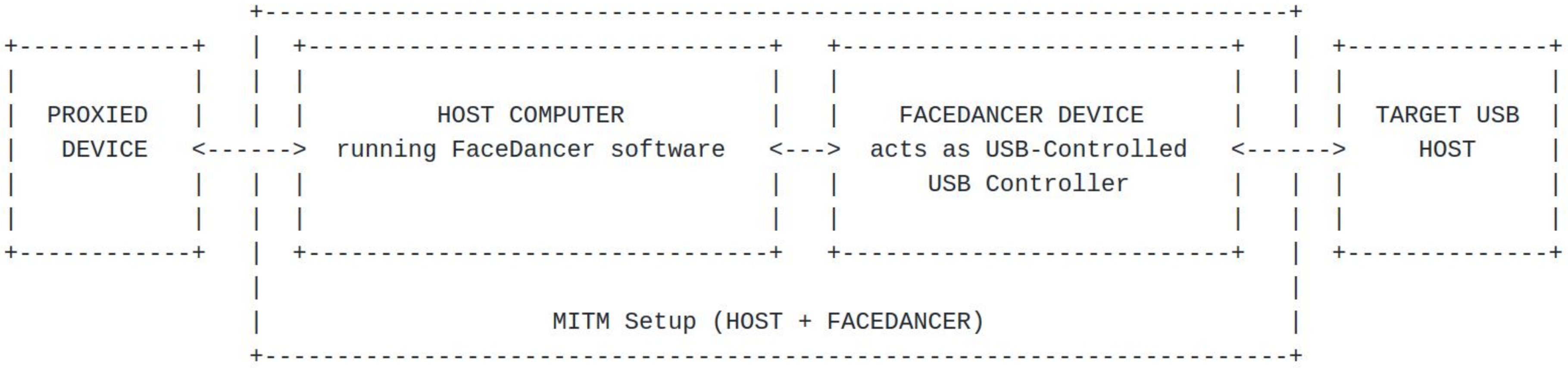
- FPGA-based open-source USB sniffer, supports low/full/high speed
- Software: original ov_ftdi or ViewSB
- Buy assembled board for 130$ (or assemble yourself)



Demo: Sniffing USB with OpenVizsla

USBProxy 'Nouveau'

- USBProxy 'Nouveau'
- Requires a MitM host computer and a Facedancer board
- Also allows MitM USB communication



PROXIED DEVICE <------> HOST COMPUTER running FaceDancer software <---> FACEDANCER DEVICE acts as USB-Controlled USB Controller <------> TARGET USB HOST

MITM Setup (HOST + FACEDANCER)

Demo: Sniffing USB with USBProxy 'Nouveau'

Hardware Keyloggers

- AirDrive Keylogger (30$ - 75$)
- KEYVILBOARD (80$)
- Maltronics WiFi KeyLogger (45$)

https://www.keelog.com/airdrive-keylogger-max-premium-usb-hardware-keylogger-with-wifi-and-flash-email-and-live-data-transfer/

Demo: Logging USB Keyboard with AirDrive Keylogger

Part 9: Defensive

Epilogue

Useful USB Links

- Materials for this talk
- USB Reverse Engineering: Down the rabbit hole
- USB 101: An Introduction to Universal Serial Bus 2.0

- Twitter: @ktemkin
- Hacking the USB World with FaceDancer
- usb-tools Discord channel

- Twitter: @mame82

whoami

- Andrey Konovalov <andreyknvl@gmail.com>
- Twitter: @andreyknvl
- GitHub: @xairy
- Telegram: @xa1ry
- More: xairy.github.io

Thanks!
Questions?

github.com/xairy/hardware-village

Andrey Konovalov <andreyknvl@gmail.com>