Virtualised USB Fuzzing using QEMU and Scapy

Breaking USB for Fun and Profit

Tobias Mueller

2015-10-01

Motivation

What's the problem?

- USB supported by every major OS
- USB widely deployed
- USB drivers in kernel space
- Not easy to assess security
  - Development board?
  - Inject messages into kernel?

Digital Voting Pen

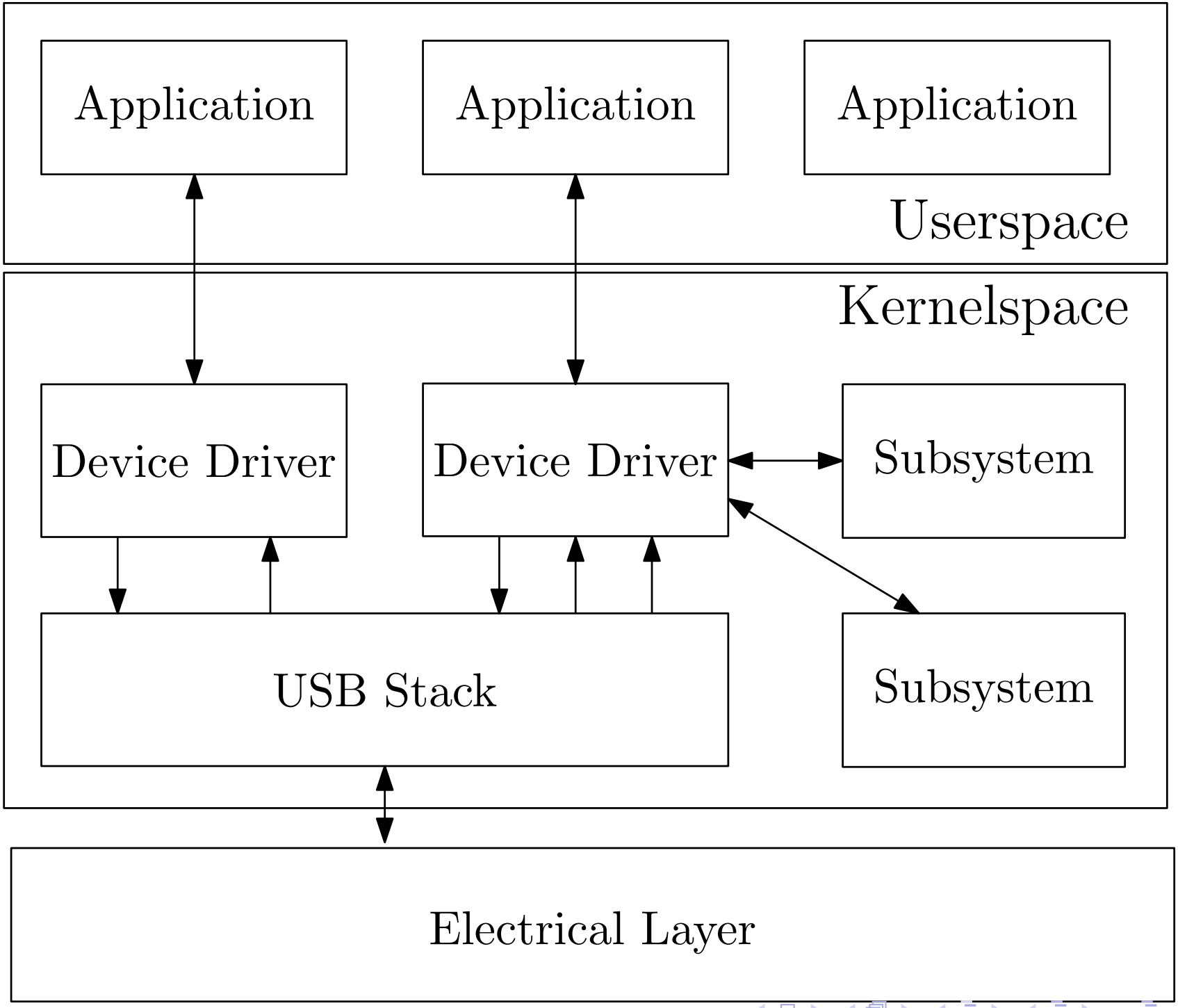
Yes, it uses USB. hehe

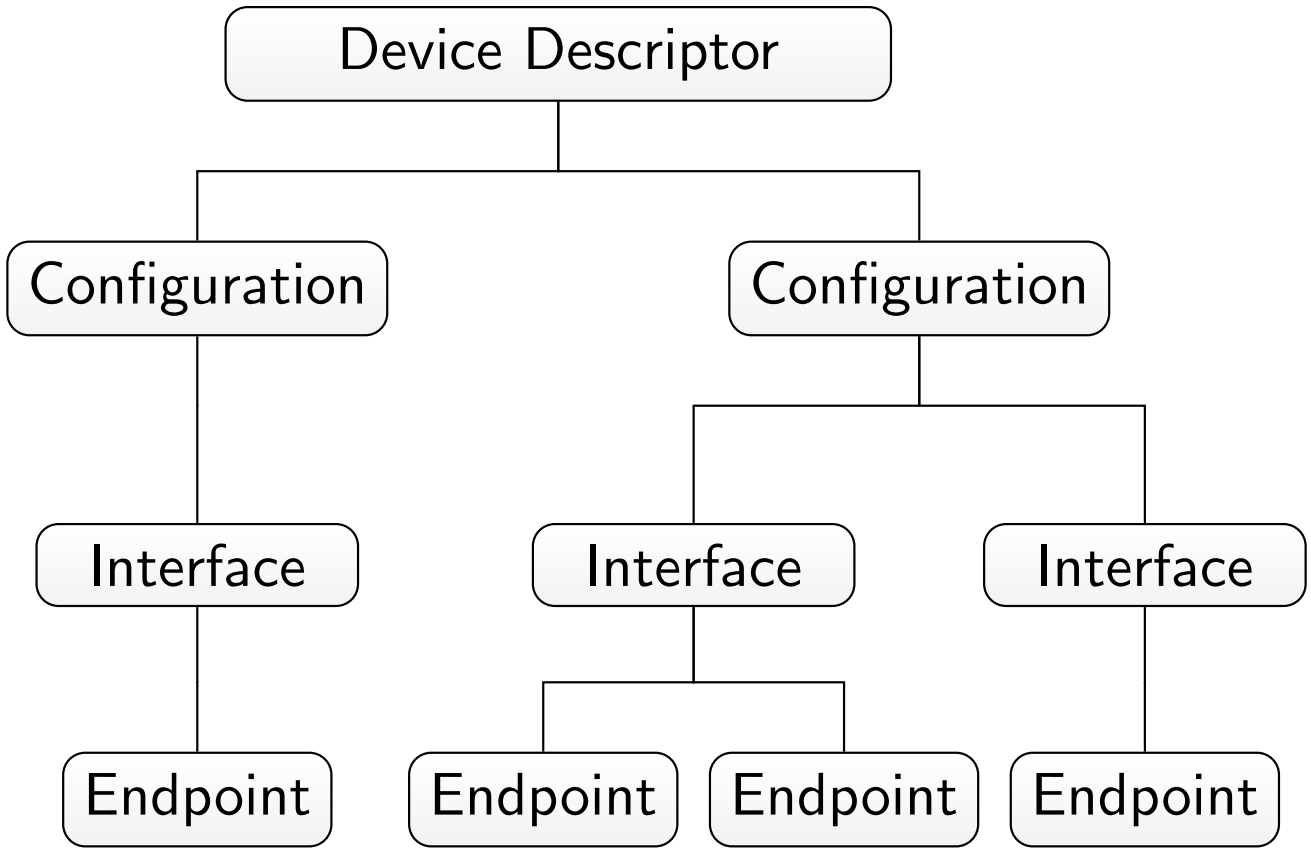
In-Flight entertainment

Based on Linux or VxWorks

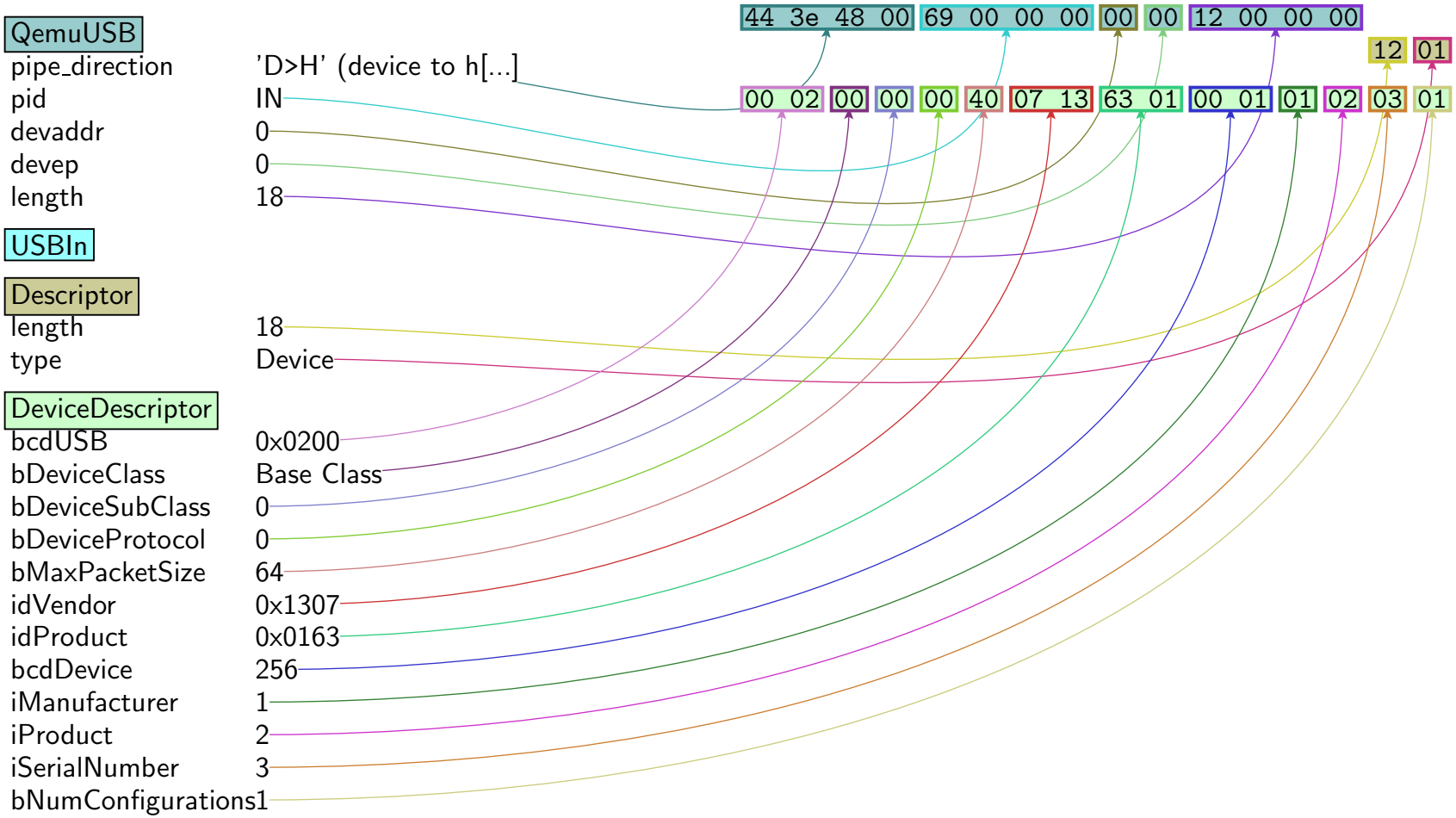
Architecture

- Host initiated communication
  - $\rightarrow$ polling
  - Yes, even with keyboards or mice
- packet-based
  - SETUP
  - IN
  - OUT


Application
Application
Application
Userspace
Kernelspace
Device Driver
Device Driver
Subsystem
USB Stack
Subsystem
Electrical Layer


Device Descriptor
Configuration
Configuration
Interface
Interface
Interface
Endpoint
Endpoint
Endpoint
Endpoint

Device Descriptor



Known USB issues

The Playstation 3 Hack

Configuration Descriptor overflow. . . m(

Known USB issues (cont.)

Solaris FAIL

Configuration Descriptor overflow by Andy Davies (CVE-2011-2295)

BadUSB

Put several classes onto one device

Physical Access?

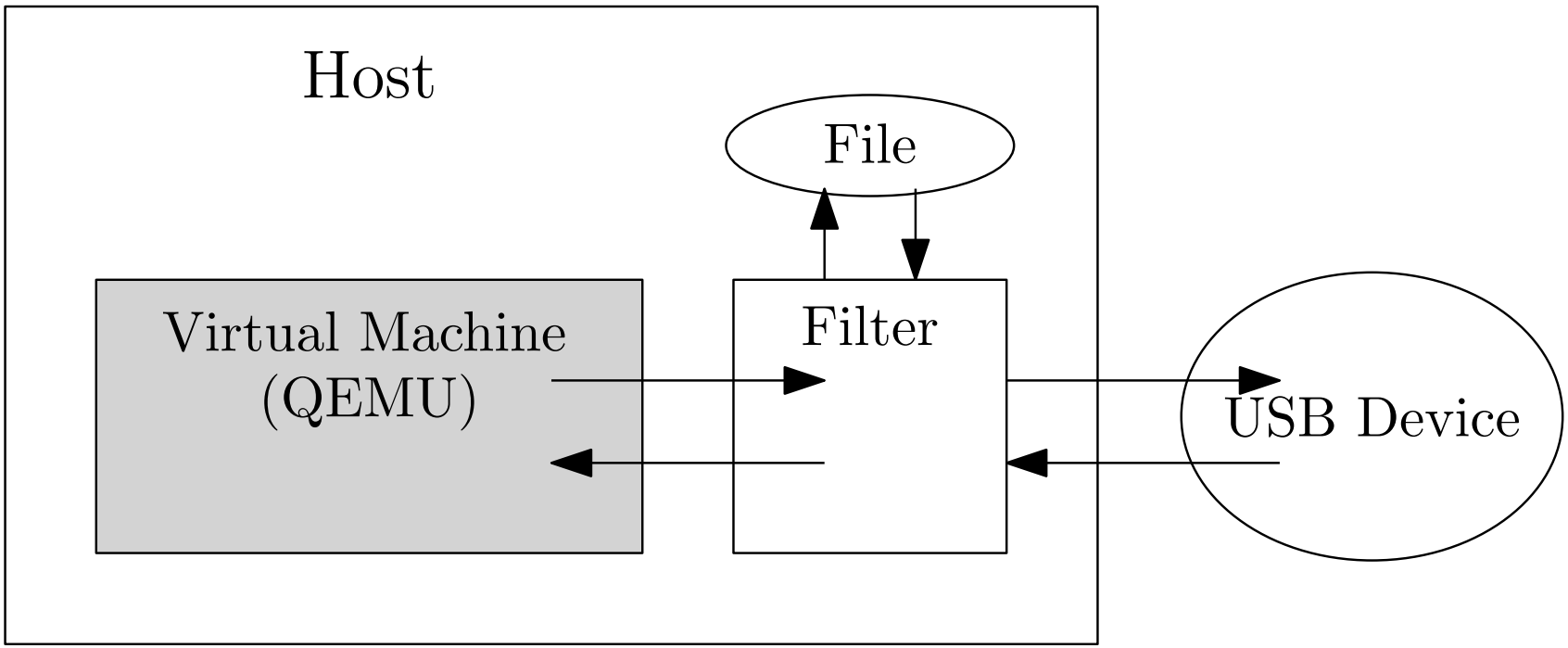
- Often argued that it's not in the OS's threat model
- except, it is...
- Not necessarily needed due to:
  - USB/IP
  - Wireless USB

Fuzzing

- Dumb Fuzzing
  - coined in late 80's
  - feed program with random(?) data
  - received a lot of attention $\sim$ 2004
- Smart Fuzzing
  - Modify existing valid structured data
  - Checksums
  - Cover more code
  - Patent encumbered?
- Scapy
  - Awesome (!) framework
  - sniff, manipulate, craft, send (Ethernet) packets
  - models packets in Python

Obtaining Valid USB communication

- Read specs :-(
- `mount none -t debugfs /sys/kernel/debug`
  `mount none -t usbmon`
  see Documentation/usb/usbmon.txt
  :-(
- Using QEMU: Implement filter to pipe out communication (originally done by Moritz Jodeit)


Host
File
Virtual Machine
(QEMU)
Filter
USB Device

QEMU

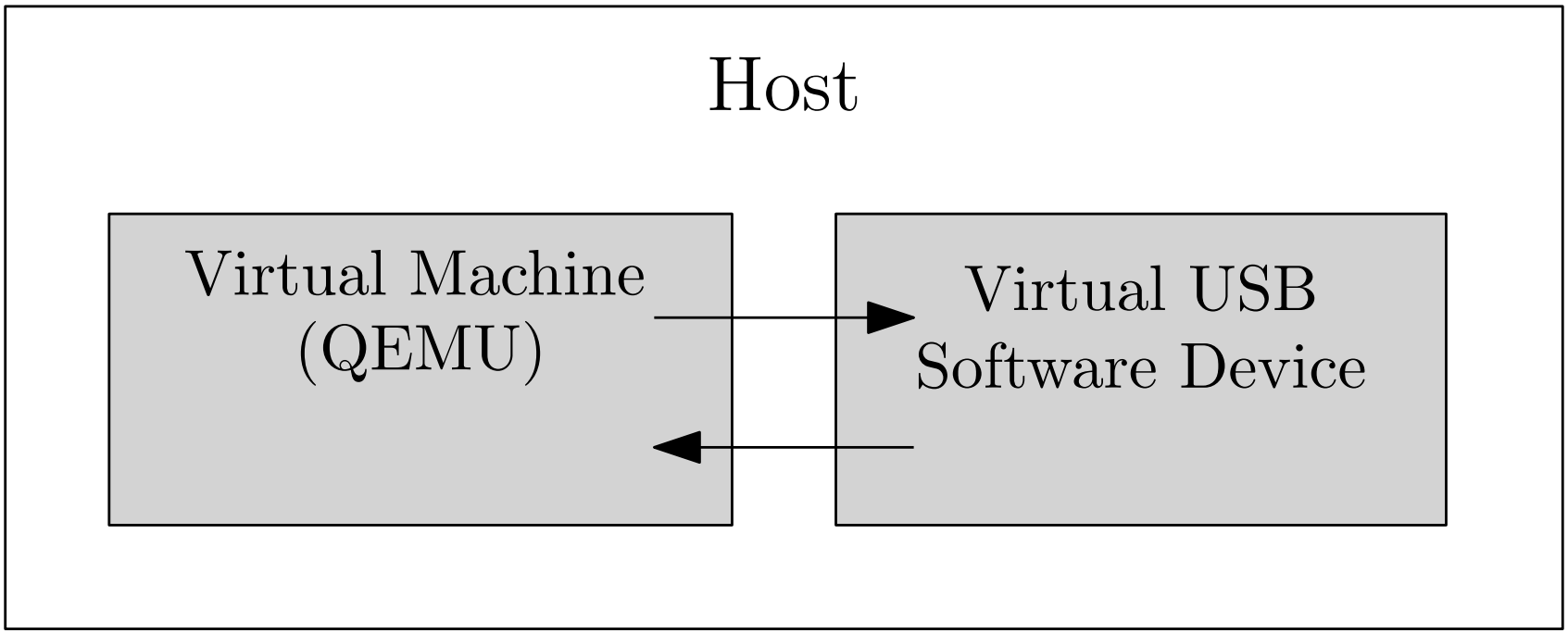
- Full virtualisation (not Xen, OpenVZ, UML, etc. . .)
- Free (as in speech) Virtualisation (not VMWare)
- Existing Virtual USB Drivers
- (Unusable) Existing infrastructure for USB indirection

Virtual USB Device

- Take simple existing MSD or Serial driver
- Write out / Read in USB packets

- Implement desired behaviour externally

- `cat` and `echo`
- Or enhancing Scapy to read/write from pipes
- $\rightarrow$ Automaton class


Host
Virtual Machine
(QEMU)
Virtual USB
Software Device

USB Stack stress testing

How many devices can you handle?

```
def run_simple_test(qemu, timeout=4, delete=False):
    qemu.usb_add('mouse')
    time.sleep(timeout)
    cmd = list('dmesg') + ['space'] \
            + ['minus'] + ['c'] + ['enter']
    qemu.sendkeys(cmd)
    usb_devices = qemu.usb_info()
    if delete:
        for device in usb_devices['usbdevices']:
            qemu.usb_del('%d.%d' %
              (device['busnr'], device['devaddr']))


    print qemu.cpu_info()
```

USB Fingerprinting

Targetted attacks

OS	Packet Sequence	Retries
Windows	SETUP, IN, OUT	3
Linux 2.6.33	SETUP (9x), RESET	4+2
OpenBSD 4.7	SETUP, IN, OUT	7
FreeBSD 8.0	SETUP, IN, OUT	6

Tabelle: USB Stack Fingerprints of various operating systems

Future Work

What's next?

- USB-3? (SuperSpeed, Device Initiated Communication)
- Making it work with GadgetFS
- Make that work on phones
- Get more OS fingerprints
- Exploit more drivers
- Run shellcode
- USB Firewall

Q&A

Questions?

Muchas Gracias!

Tobi(as) Mueller

Mail 4tmuelle@informatik.uni-hamburg.de

FF52 DA33 C025 B1E0 B910

92FC 1C34 19BF 1BF9 8D6D