Lowering the USB Fuzzing Barrier by Transparent Two-Way Emulation

Rijnard van Tonder
Herman Engelbrecht

Stellenbosch University

Motivation

- High-impact security bugs reside in the USB attack surface

- Challenging to explore due to
  - limited pure software solutions
  - hardware acquisition
  - inflexible hardware for security testing
  - knowledge requirement of USB

- Can we do better?

Existing Solutions

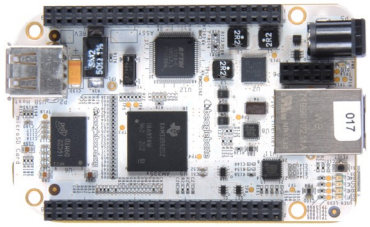
Software:

- Qemu emulation (MWR Labs, ‘11)
- Frisbee Lite (Davis, ‘12)

Hardware:

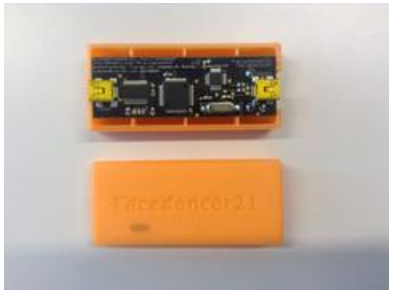
- USB Analyzer, Frisbee and GraphicUSB (Davis, ‘11)

- BeagleBone and USBProxy (Spill ‘14)

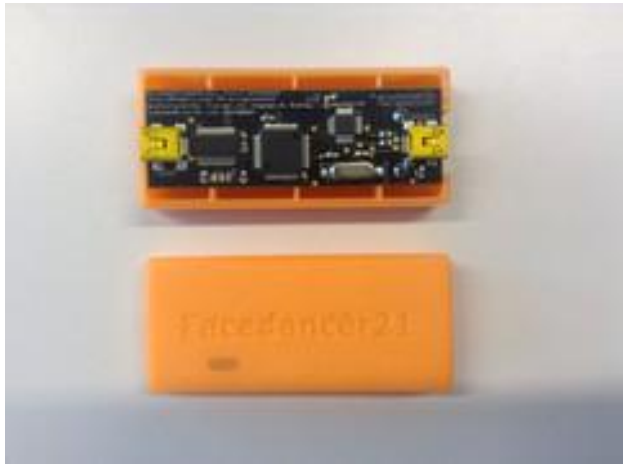


- Arduino (Ose, ‘11, Davis, ‘11)

- Facedancer (Goodspeed, Bratus, ‘12), umap (Davis, ‘13)



Fast	✓
Read and Write ability	✓
Man-in-the-middle	✓
Knowledge requirement	✓
Cost	✗
Flexible	✗



Fast	✗
Read and Write ability	✓
Man-in-the-middle	✗
Knowledge requirement	✗
Cost	✓
Flexible	✓

Contributions

- The TTWE USB fuzzing framework that
  - Is flexible,
  - Is cost-effective, and
  - Lowers the knowledge requirement
- Initial results and analysis of bug-hunting with TTWE
- New possibilities for USB fuzzing and attacks

USB Protocol Primer

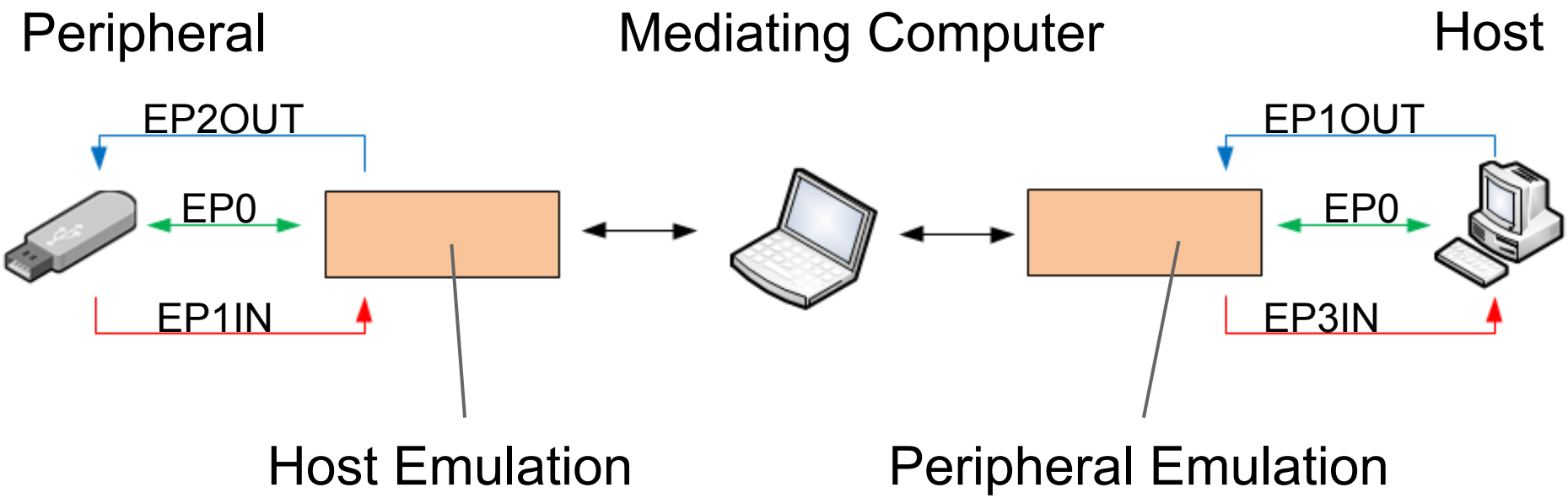
- Consists of *requests* and *descriptors* exchanged between *host* and *peripheral*
- USB defines device classes for peripherals
- Endpoints designate data *direction* and *address*
- Control transfers and Non-control transfers
- Packets
  - Token
  - Data
  - Handshake

TTWE

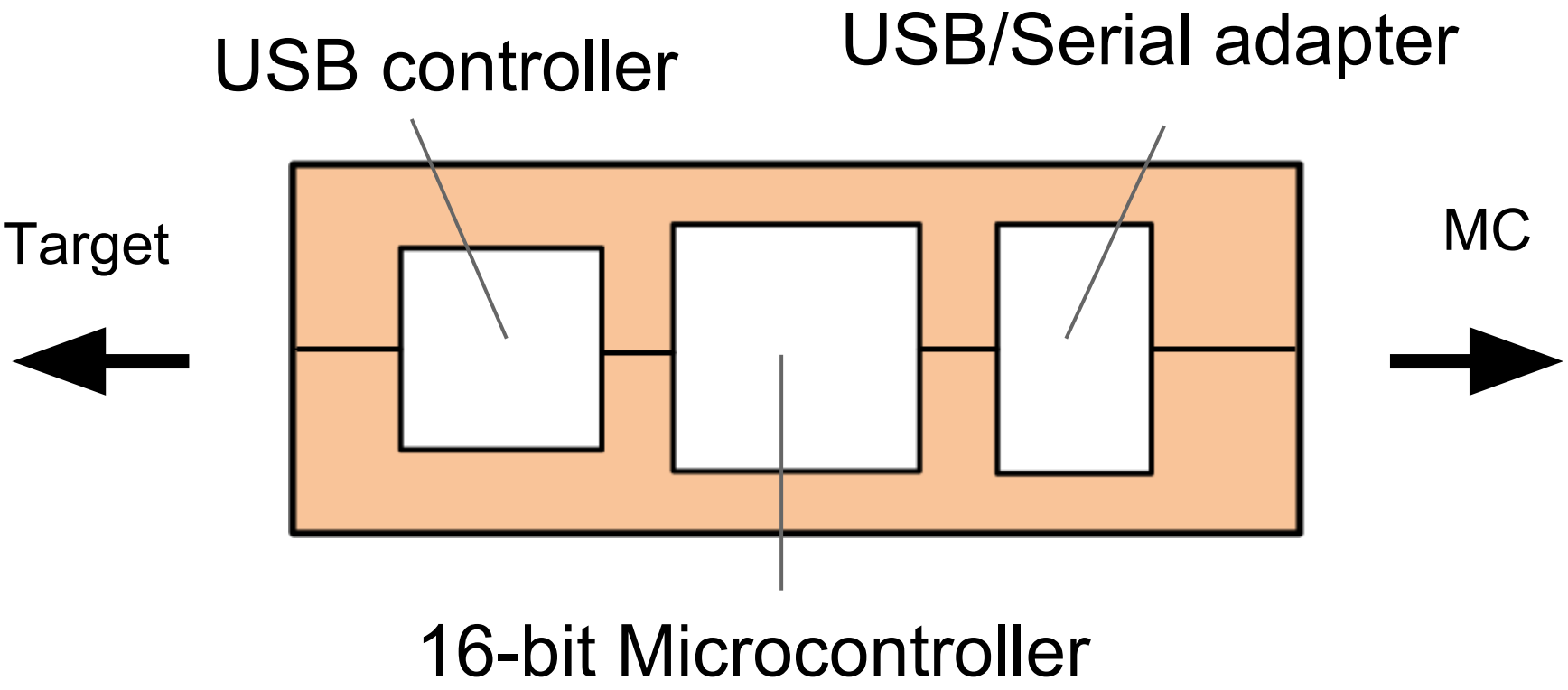
- Tap into the communication between host and peripheral
- Modify communication
- The Facedancer can emulate host or peripheral devices

- Emulate both simultaneously

Design


Peripheral
Mediating Computer
Host
EP2OUT
EP0
EP1IN
EP1OUT
EP0
EP3IN
Host Emulation
Peripheral Emulation

Hardware Implementation



Software Implementation

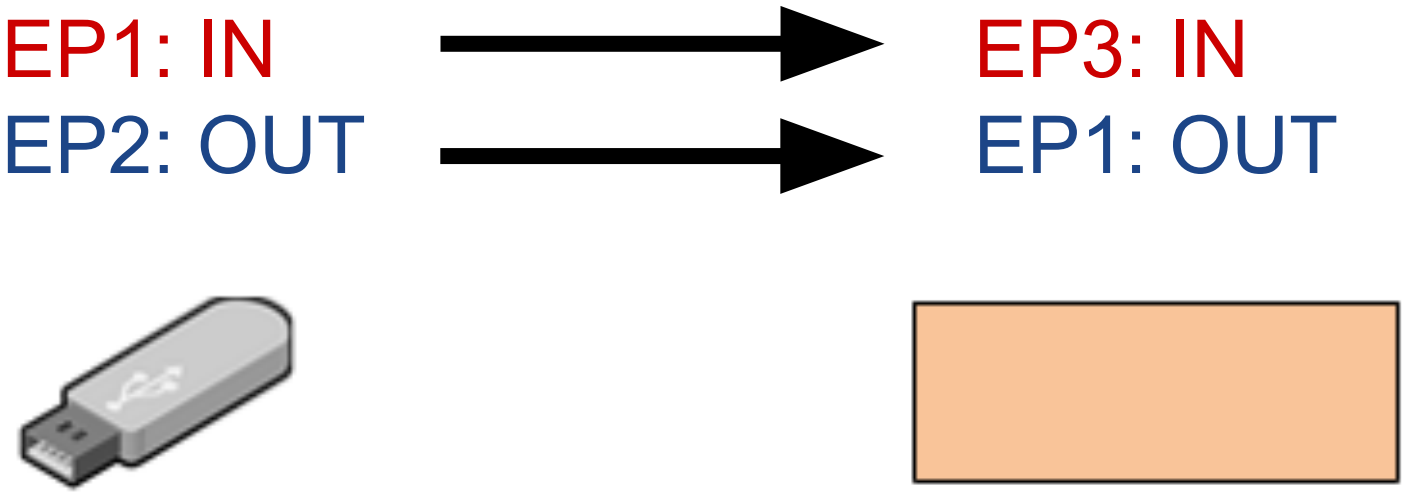
- Emulation drivers
  - Host and Peripheral mode
  - Communicate via named pipes

Two challenges:

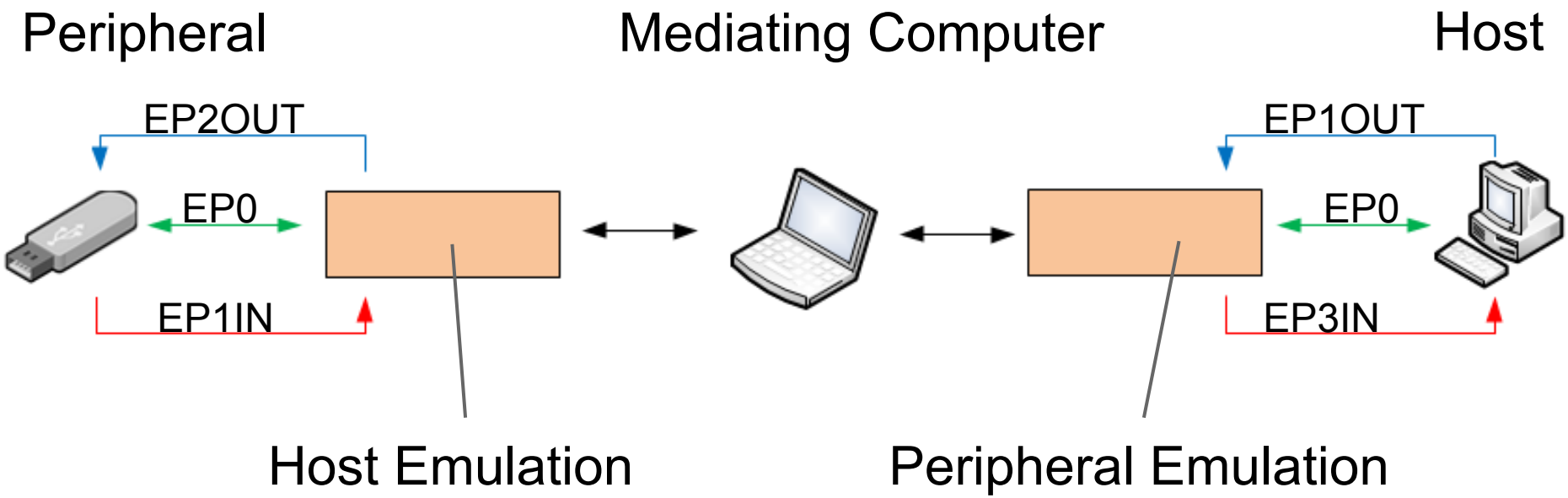
- Endpoint Hijacking
- Handshake emulation

Endpoint Hijacking

- Problem: hardcoded endpoint descriptors



Design


Peripheral
Mediating Computer
Host
EP2OUT
EP0
EP1IN
EP1OUT
EP0
EP3IN
Host Emulation
Peripheral Emulation

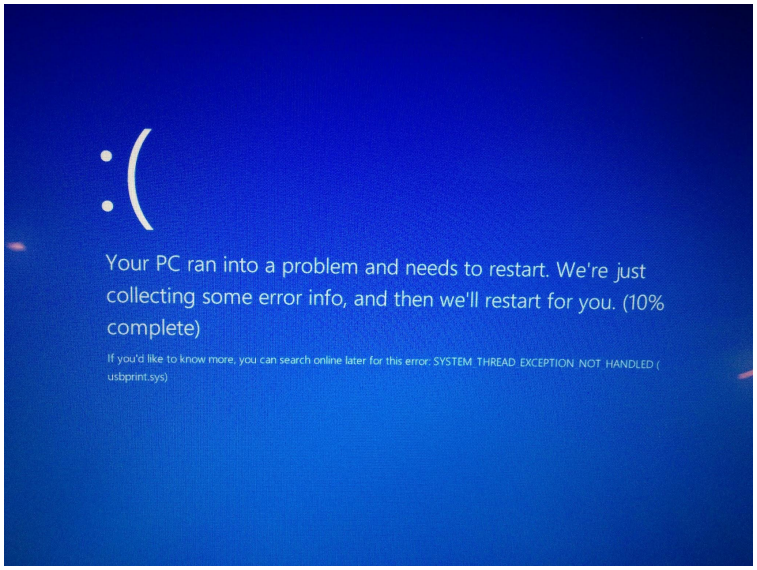
Transparent Emulation Results

- Mass storage device
  - Enumeration
  - SCSI data
  - Mount, read, and write ability

DIR	EP	DATA (Base 10)
OUT	[0]	[128, 6, 0, 1, 0, 0, 64, 0]
IN	[0]	[18, 1, 0, 2, 0, 0, 0, 64, 143, 5, 135, 99, 2, 1, 1, 2, 3, 1]
OUT	[0]	[0, 5, 25, 0, 0, 0, 0, 0]
OUT	[0]	[128, 6, 0, 1, 0, 0, 18, 0]
IN	[0]	[18, 1, 0, 2, 0, 0, 0, 64, 143, 5, 135, 99, 2, 1, 1, 2, 3, 1]
OUT	[0]	[128, 6, 0, 6, 0, 0, 10, 0]
IN	[0]	[10, 6, 0, 2, 0, 0, 0, 64, 1, 0]
OUT	[0]	[128, 6, 0, 2, 0, 0, 9, 0]
IN	[0]	[9, 2, 32, 0, 1, 1, 0, 128, 100]
OUT	[0]	[128, 6, 0, 2, 0, 0, 32, 0]
IN	[0]	[9, 2, 32, 0, 1, 1, 0, 128, 100, 9, 4, 0, 0, 2, 8, 6, 80, 0, 7, 5, **1**, 2, 64, 0, 0, 7, 5, **130**, 2, 64, 0, 0]
OUT	[0]	[128, 6, 0, 3, 0, 0, 255, 0]
IN	[0]	[4, 3, 9, 4]
OUT	[0]	[128, 6, 2, 3, 9, 4, 255, 0]
IN	[0]	[26, 3, 77, 0, 97, 0, 115, 0, 115, 0, 32, 0, 83, 0, 116, 0, 111, 0, 114, 0, 97, 0, 103, 0, 101, 0]
OUT	[0]	[128, 6, 1, 3, 9, 4, 255, 0]
IN	[0]	[16, 3, 71, 0, 101, 0, 110, 0, 101, 0, 114, 0, 105, 0, 99, 0]
OUT	[0]	[128, 6, 3, 3, 9, 4, 255, 0]
IN	[0]	[18, 3, 49, 0, 57, 0, 54, 0, 50, 0, 51, 0, 55, 0, 51, 0, 54, 0]
OUT	[0]	[0, 9, 1, 0, 0, 0, 0, 0]
OUT	[0]	[161, 254, 0, 0, 0, 0, 1, 0]
IN	[0]	[0]
OUT	[1]	[85, 83, 66, 67, 1, 0, 0, 0, 36, 0, 0, 0, 128, 0, 6, 18, 0, 0, 0, 36, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
IN	[3]	[0, 128, 4, 2, 31, 0, 0, 0, 71, 101, 110, 101, 114, 105, 99, 32, 70, 108, 97, 115, 104, 32, 68, 105, 115, 107, 32, 32, 32, 32, 32, 32, 56, 46, 48, 55]
IN	[3]	[85, 83, 66, 83, 1, 0, 0, 0, 0, 0, 0, 0, 0]
OUT	[1]	[85, 83, 66, 67, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
IN	[3]	[85, 83, 66, 83, 2, 0, 0, 0, 0, 0, 0, 0, 1]
OUT	[1]	[85, 83, 66, 67, 3, 0, 0, 0, 18, 0, 0, 0, 128, 0, 6, 3, 0, 0, 0, 18, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
…	…	…

Fuzzing Results

- “Dumb” fuzzing setup
- Printer Driver bug
  - Memory corruption
- Application DoS on print
  - Waits for ACK
- WiFi dongle
  - Invalid response to clear_feature
- Mass storage driver bug in printer
  - Malformed SCSI response



Limitations

- Slow
- Device timeouts
- Number of endpoints

Conclusion

- Flexible and inexpensive way to explore the USB attack surface
- Record and replay when fuzzing

Further avenues:

- TOCTTOU RIT attack (Mulliner, ‘12)
- Devices-as-seed-files

Questions

?

@rvtond

https://github.com/rvantonder/ttwe-proto

rvantonder@ml.sun.ac.za