VIA: Analyzing Device Interfaces of Protected Virtual Machines

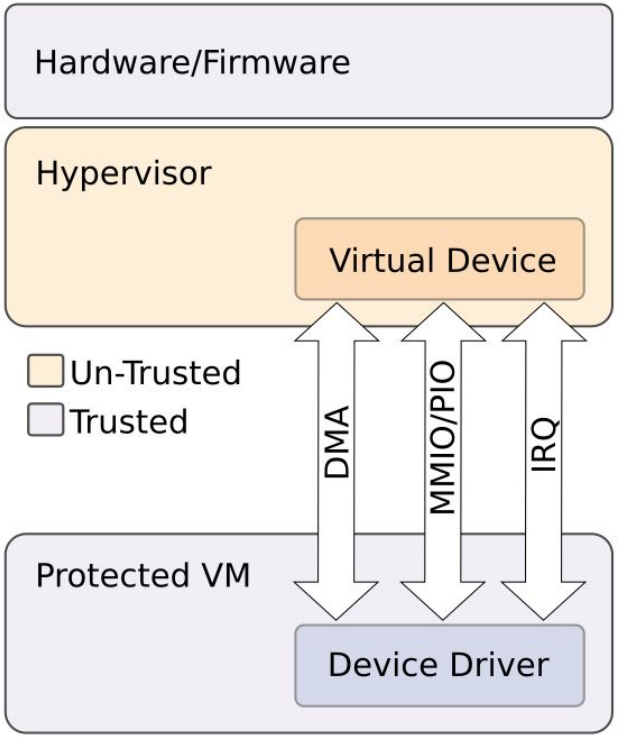
Felicitas Hetzelt (TUB), Martin Radev (AISEC*), Robert Buhren (TUB), Mathias Morbitzer (AISEC), Jean-Pierre Seifert (TUB)

Protected Virtualization (Motivation and Background)

New Technologies: AMD SEV(-ES, -SNP), INTEL TDX

- Protect complete commodity operating system
- Hypervisor excluded from TCB

Trust Boundary between virtual Devices and protected VM



Vulnerabilities in the Hardware-OS interface

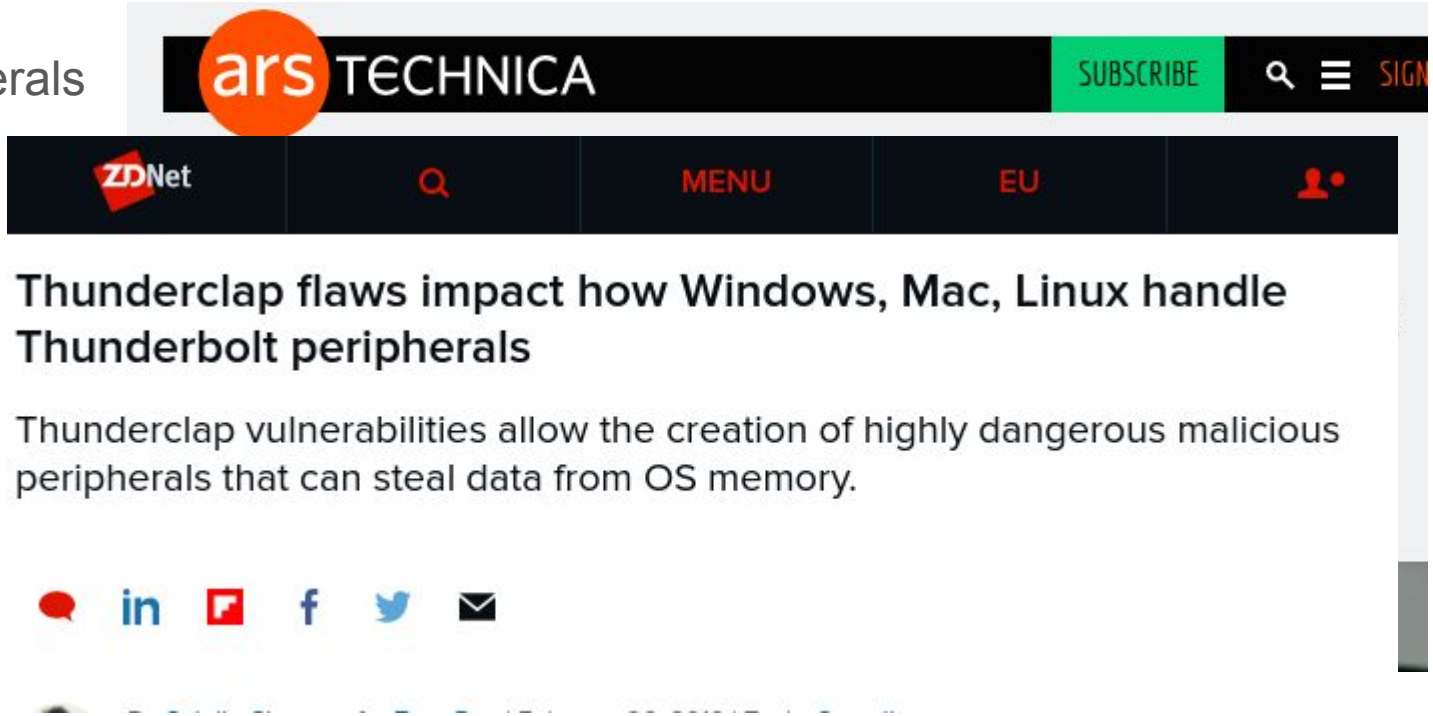
(virtual) Devices used to be trusted

Vulnerabilities in the Hardware-OS interface

- SoC Peripherals

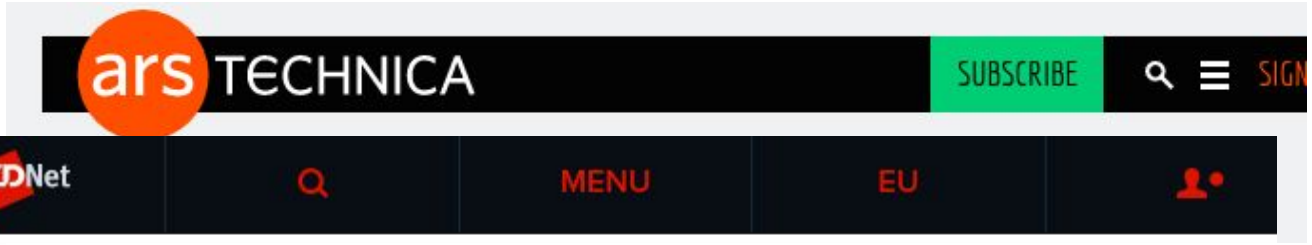
Vulnerabilities in the Hardware-OS interface

- SoC Peripherals
- PCI



Vulnerabilities in the Hardware-OS interface

- SoC Peripherals
- PCI
- USB



These are the bugs that were manually reported before USB fuzzing was integrated into syzbot.

USB drivers

- usb/core: memory corruption due to an out-of-bounds access in usb_destroy_configuration [fix] [CVE-2017-17558]
- usb/net/zd1211rw: possible deadlock in zd_chip_disable_rxtx
- usb/sound: use-after-free in __uac_clock_find_source [fix]
- usb/sound: slab-out-of-bounds in parse_audio_unit [fix]
- usb/media/em28xx: use-after-free in dvb_unregister_frontend [fix]
- usb/media/technisat: slab-out-of-bounds in technisat_usb2_rc_query
- usb/media/tm6000: use-after-free in tm6000_read_write_usb
- usb/net/qmi_wwan: divide error in qmi_wwan_probe/usbnet_probe [fix1, fix2] [CVE-2017-16649, CVE-2017-16650]
- usb/media/uvc: slab-out-of-bounds in uvc_probe
- usb/media/em28xx: use-after-free in em28xx_dvb_fini
- usb/media/em28xx: use-after-free in v4l2_fh_init
- usb/media/pvrusb2: WARNING in pvr2_i2c_core_done/sysfs_remove_group

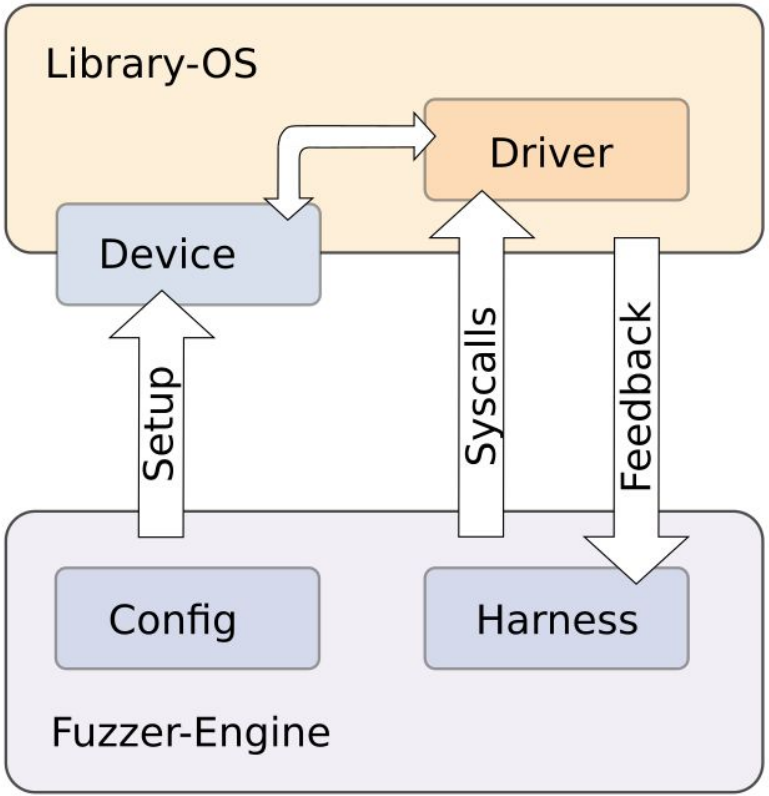
VIA’s Goal

Provide a generic tool to analyze the hardware interface of device drivers commonly used in virtual machines to find software vulnerabilities.

VIA’s Approach - Overview

Targeted driver fuzzing tool build on lkl and libfuzzer

- Target drivers loaded as shared library
- Generic VIRTIO, PCI and Platform device stubs
- Configuration files
- Userspace harness



Challenges in Fuzzing the Virtual Device Interface

- Low testcase throughput
  - Delays in driver code
  - Inefficient IO-Interception (VMEXIT, Page-Faults)
- No Interface for Coherent DMA Interception
  - Fresh values need to be provided on each access to coherent DMA area
- In-efficient Interrupt Scheduling
  - Driver stalls until interrupt is scheduled
  - Performance loss when triggering too many interrupts
- State Accumulation

VIA’s Approach - Details

- Improve Test Case Throughput
  - Remove delays in driver code (*delay, *sleep, schedule_timeout[_*], time_before/after)
- IO-Interception
  - Streaming DMA, MMIO and PIO: adapt existing kernel interfaces (read*, in*, sync_for_cpu, dma_unmap)
  - Coherent DMA: adapt ASAN instrumentation
- Interrupt Scheduling:
  - Track “waiting” workloads (wait_for_completion_*, *_wait_event_*) to schedule interrupts at useful execution points
- Reset State by Reloading Driver in each Iteration

VIA Performance

- 570 executions/s on average
- 163 improvement on average due to delay reduction
- 2706 improvement in executions/s (1915 without delay optimization) and 2.26 improvement in coverage compared to VM-based approach (Agamotto)

	# Executions / s			# Paths	
	VIA-D	VIA-ND	(Increase)	VIA-D	VIA-ND
8139cp	1.32	122.41	×**92.58**	1038	1040
acpi	8.00	8.00	×1.0	71	71
e100	63.19	231.98	×**3.67**	573	569
e1000	3.00	259.06	×**86.35**	1427	1535
e1000e	0.70	111.25	×**158.92**	1386	1579
gve	2.00	636.22	×**318.11**	147	594
ne2k-pci	1408.00	1658.00	×**1.18**	31	31
nvme	0.02	0.88	×**44.0**	260	291
qemu-fw-cfg	1254.00	1341.0	×**1.06**	35	37
rocker	171.01	203.25	×**1.19**	181	184
sungem	6.01	59.04	×**9.82**	924	1032
sunhme	195.00	428.00	×**2.19**	1025	1030
tpm-tis	2.00	857.00	×**428.50**	150	326
vio-balloon	1291.00	1328.00	×**1.03**	281	281
vio-blk	625.00	624.00	×1.00	333	333
vio-console	349.00	444.00	×**1.27**	352	352
vio-crypto	270.00	277.00	×**1.03**	258	258
vio-input	393.00	635.00	×**1.62**	299	299
vio-net	553.00	400.00	×0.72	1250	1257
vio-rng	1.00	2282.00	×**2282.00**	238	239
vmxnet3	37.07	59.94	×**1.62**	51	51

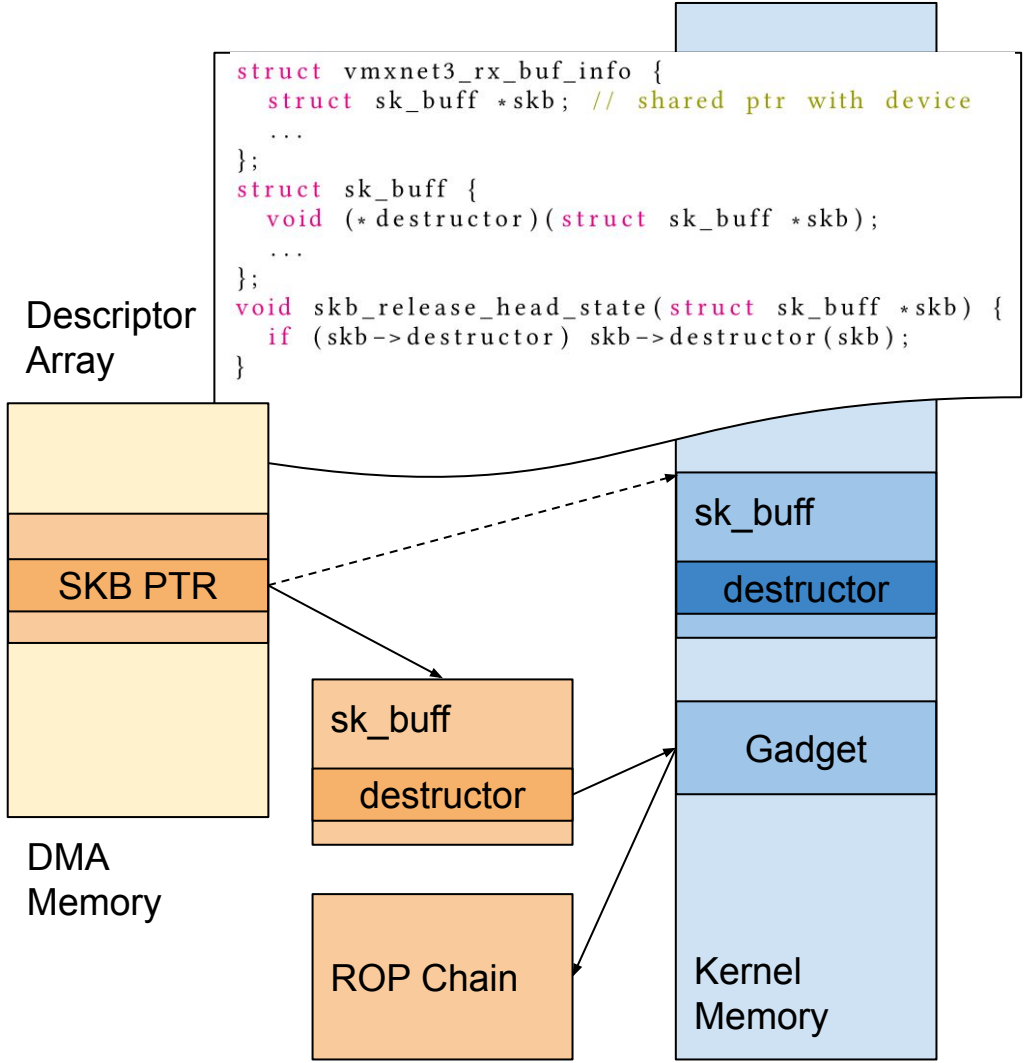
Bugs

- Analyzed **VIRTIO**, **PCI** and **Platform** drivers from **Qemu devices** and **Google confidential** VM (SEV)
  - **~50 bugs** across 22 analyzed drivers (2 drivers had no issues)
  - Missing sanitization
  - Incomplete / failed initialization
  - Shared control data
- Exploitability:
  - 23/50 bugs likely not exploitable
  - HV has advanced exploitation capabilities

Bug Class	Count
Out-of-Bounds access	14
Invalid memory access	10
Slab management	8
Device-shared pointer	5
Miscellaneous	3
Assertion failure (BUG)	4
Unbounded allocation	5
Deadlock	1

Device-shared pointer in vmxnet3

- Pointer to sk_buf placed in DMA memory area
- Device overwrites sk_buf pointer to point to device controlled memory
- Device points destructor function pointer to code gadget
- Gadget pivots stack to ROP chain in device controlled memory

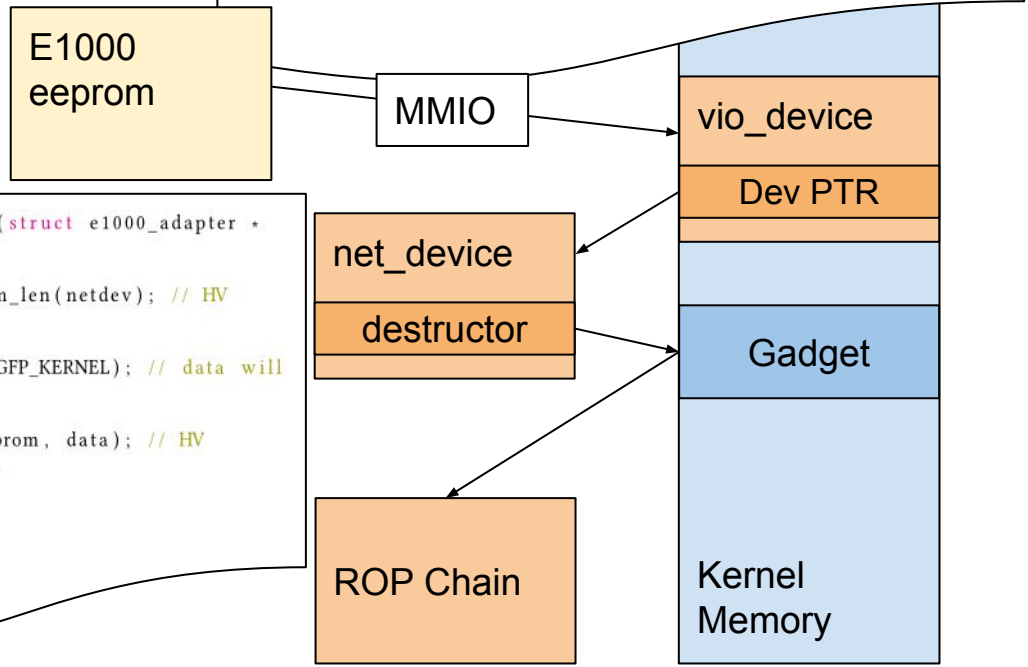


Use-After-Free in virtio_net

- Device induces virtnet_probe fail; virtio_device is freed; error value is not set
- Device induces overlapping allocation of e1000 eeprom

```
static int virtnet_probe(struct virtio_device *vdev) {
  ...
free:
  free_netdev(dev); // struct net_device is freed
  return err;       // err == 0
}
static void virtnet_remove(struct virtio_device *vdev)
{
  struct virtnet_info *vi = vdev->priv;
  ...
  unregister_netdev(vi->dev); // vi->dev is device
    controlled
  ...
}
```

```
static void e1000_dump_eeprom(struct e1000_adapter *
    adapter) {
  ...
  eeprom.len = ops->get_eeprom_len(netdev); // HV
    controls eeprom.len
  ...
  data = kmalloc(eeprom.len, GFP_KERNEL); // data will
    overlap virtnet_info
  ...
  ops->get_eeprom(netdev, &eeprom, data); // HV
    controlled data is copied
  ...
  kfree(data);
}
```



Limitations and Summary

- Applying VIA to 22 device drivers uncovered a large amount of vulnerabilities undermining the efficacy of protected virtualization technologies

- Intel TDX implements device white lists to limit the virtual device attack surface, however:
  - Many bugs affect drivers that are included in the white list
  - Cloud providers might have individual hardware requirements. E.g. none of the devices in the while list are used in the Google Confidential VM
- Limitations / Future Work:
  - No Concurrency
  - State Accumulation
  - Improved Fuzzing Methods

Thank you

https://github.com/file-citas/via