RUHR UNIVERSITÄT BOCHUM

RUB

NYX

Greybox Hypervisor Fuzzing
Using Fast Snapshots and Affine Types

Sergej Schumilo, Cornelius Aschermann, Ali Abbasi, Simon Wörner and Thorsten Holz

Chair for Systems Security
Ruhr-Universität Bochum

Motivation

Motivation

Hypervisor

Motivation

Hypervisor

Virtual Machine

Motivation

Hypervisor

Virtual Machine

Motivation

Hypervisor

Motivation

Virtual Machine

Motivation

Hypervisor

Virtual Machine

Related Work

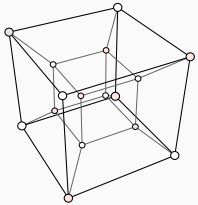
Henderson et al. "**VDF: Targeted Evolutionary Fuzz Testing of Virtual Devices**"

(RAID 2017)

Schumilo et al. "**Hyper-Cube: High-Dimensional Hypervisor Fuzzing**"

(NDSS 2020)

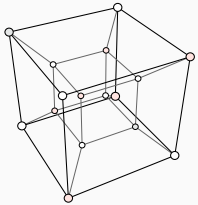
Our Approach



Hyper-Cube

[1] Schumilo et. al - NDSS 2020

Our Approach



Hyper-Cube

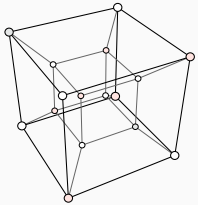
+

kAFL [2] /

REDQUEEN [3]

[1] Schumilo et. al - NDSS 2020
[2] Schumilo et. al. - USENIX Security 2017
[3] Aschermann et. al - NDSS 2019

Our Approach



Hyper-Cube

+

kAFL[2]/

REDQUEEN[3]

+

Other Improvements

[1] Schumilo et. al - NDSS 2020
[2] Schumilo et. al. - USENIX Security 2017
[3] Aschermann et. al - NDSS 2019

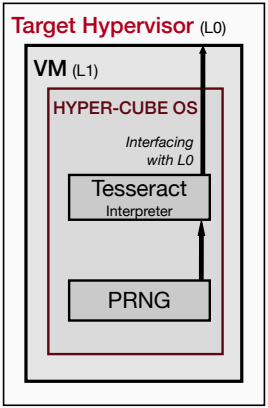
Design & Implementation

Contributions

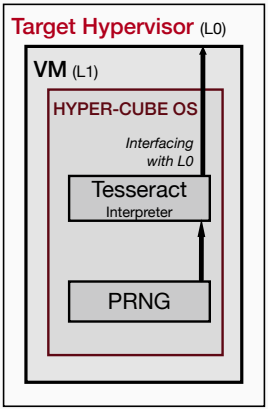
- **Hypervisor Fuzzing** (coverage-guided)
- **Fast Snapshots** (for stateful code)
- **Affine Type Mutation Engine** (for complex interfaces)

Coverage-Guided Hypervisor Fuzzing

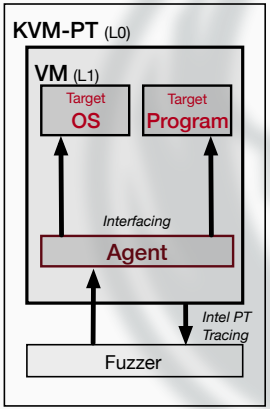
Hypervisor Fuzzing



Hypervisor Fuzzing

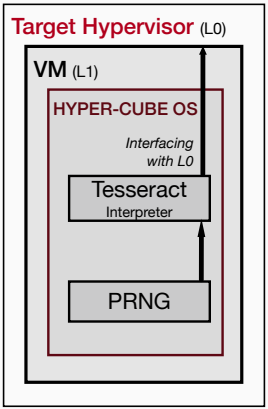


HYPER-CUBE

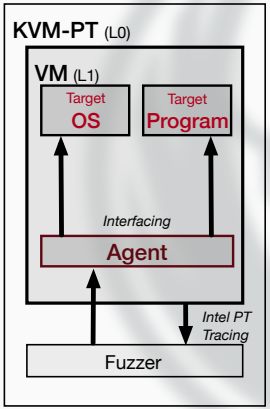


kAFL / REDQUEEN

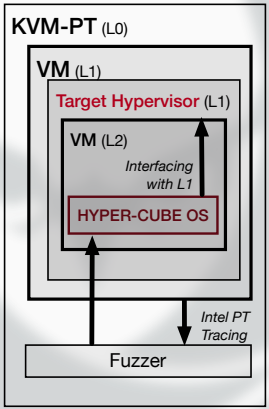
Hypervisor Fuzzing



HYPER-CUBE



kAFL / REDQUEEN



NYX

Nested Fuzzing Features

- **Special Hypercalls (L2-L0)**

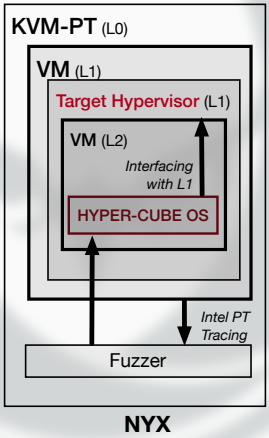
  (tranparent to the target hypervisor)

- **Intel PT Tracing (L1 only)**

  (by enabling tracing during VMX transitions)

- **Inter-VM SHM (L0 to L2)**

  (by translating HVA to L2 GPA)



Fast Snapshots

Fast Snapshots

- **Dirty Page Logging**

  (via stack instead of a bitmap)

Fast Snapshots

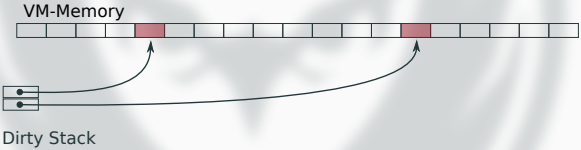
- **Dirty Page Logging**

  (via stack instead of a bitmap)

Fast Snapshots

- **Dirty Page Logging**

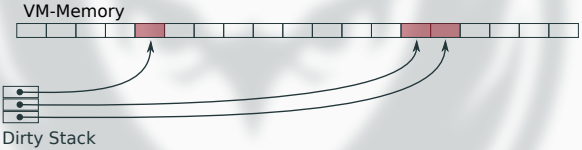
  (via stack instead of a bitmap)



Fast Snapshots

- **Dirty Page Logging**

  (via stack instead of a bitmap)



Fast Snapshots

- **Dirty Page Logging**

  (via stack instead of a bitmap)

- **Fast Device State Reset**

  (by flatten QEMU's VMState tree)

Fast Snapshots

- **Dirty Page Logging**

  (via stack instead of a bitmap)

- **Fast Device State Reset**

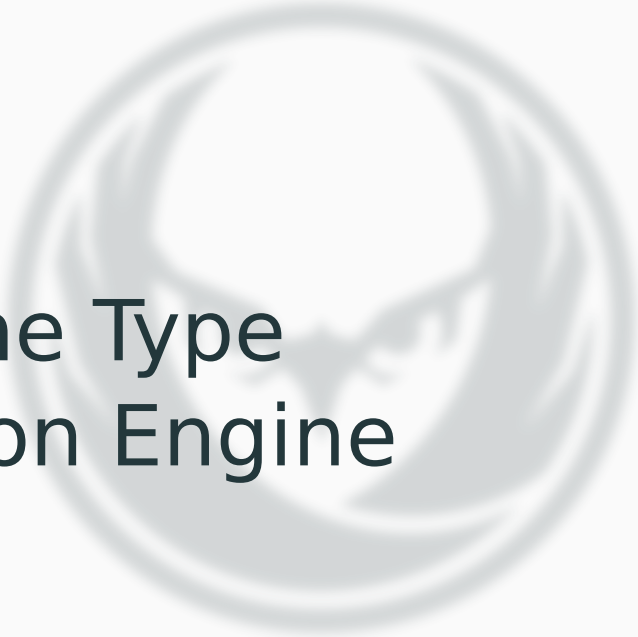
  (by flatten QEMU's VMState tree)

- **QEMU Disk COW Layer**

  (in-memory state)

O(1) Reset

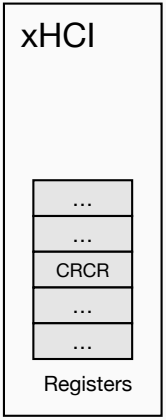
Affine Type Mutation Engine



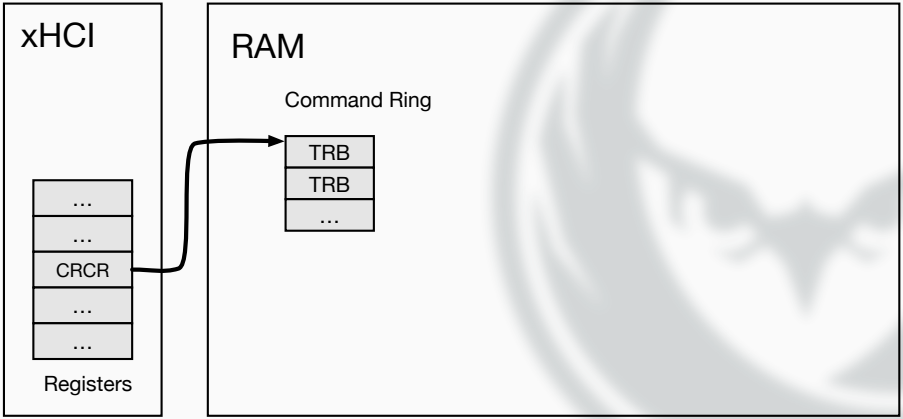
Hypervisor Attack Surface

- Memory-Mapped I/O (**MMIO**)
- Legacy Port I/O (**PIO**)
- Hypercalls
- Direct Memory Access (**DMA**)
- ...

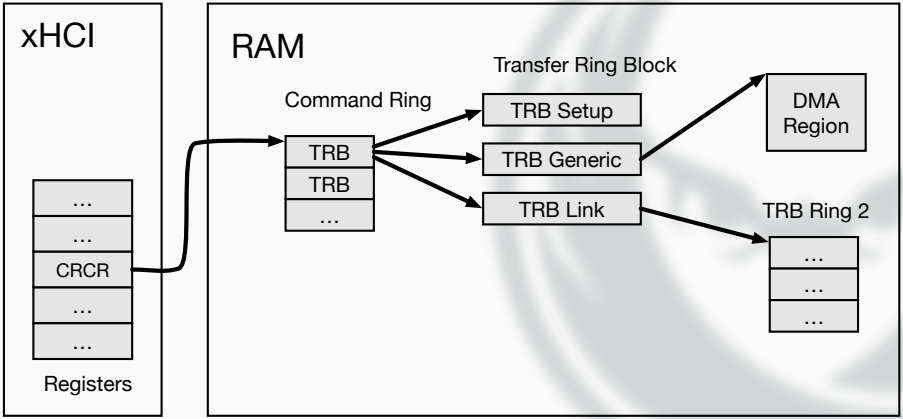
xHCI in a Nutshell



xHCI in a Nutshell



xHCI in a Nutshell



Bytecode

```
cmd_ring = alloc_cmd_ring()
cmd_ring.mmio_enable()
trb_msg = Data(0x01, 0x00, 0x20, 0x00, 0x20, ...)
cmd_ring.push_trb(data)
cmd_ring.free()
```

Bytecode

```
cmd_ring = alloc_cmd_ring()
cmd_ring.mmio_enable()
trb_msg = Data(0x01, 0x00, 0x20, 0x00, 0x20, ...)
cmd_ring.push_trb(data)
cmd_ring.free()
```

Grammar Fuzzing?

Bytecode

```
cmd_ring = alloc_cmd_ring()
cmd_ring.mmio_enable()
trb_msg = Data(0x01, 0x00, 0x20, 0x00, 0x20, ...)
cmd_ring.push_trb(data)
cmd_ring.free()
```

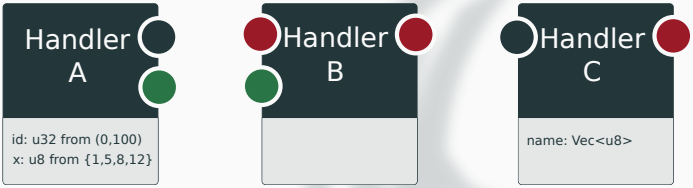
Mutated
AFL-Style

Bytecode

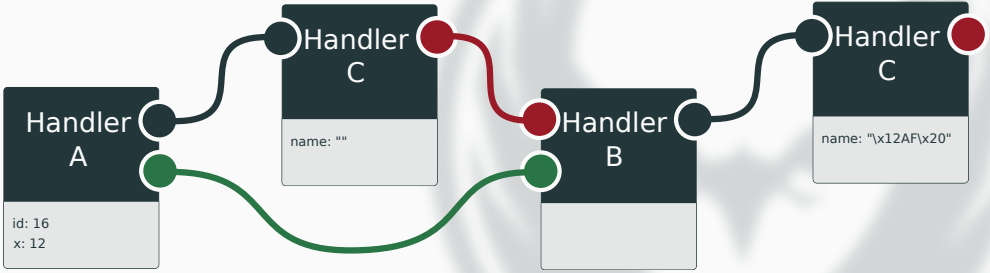
```
cmd_ring = alloc_cmd_ring()
cmd_ring.mmio_enable()
trb_msg = Data(0x01, 0x00, 0x20, 0x00, 0x20, ...)
cmd_ring.push_trb(data)
cmd_ring.free()
```

Not reused

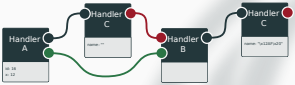
Graph Mutations



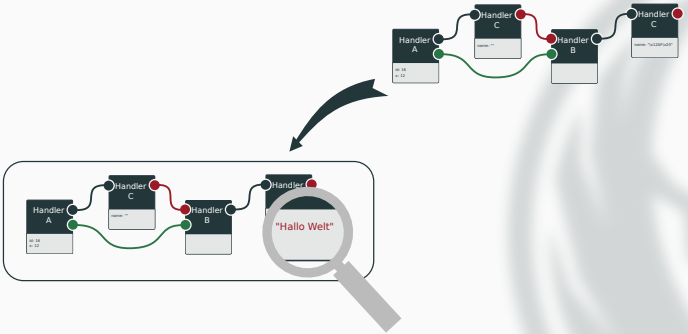
Graph Mutations



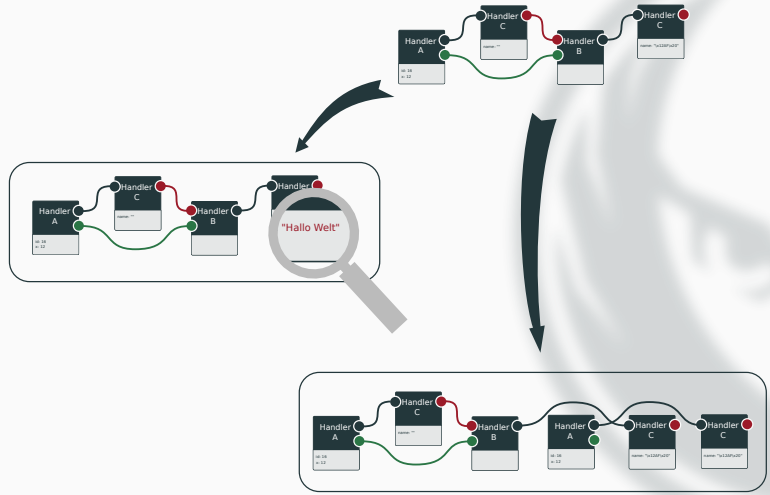
Graph Mutations



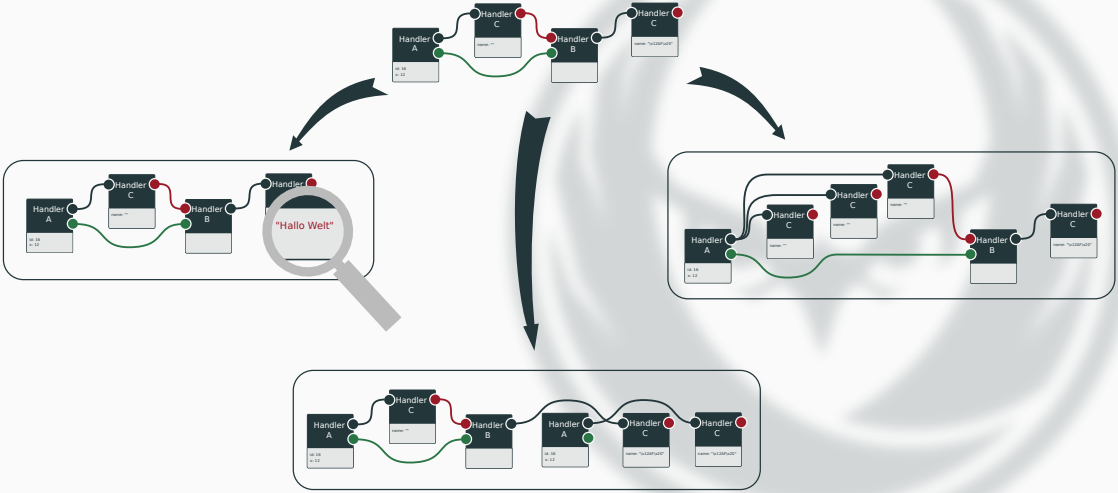
Graph Mutations



Graph Mutations

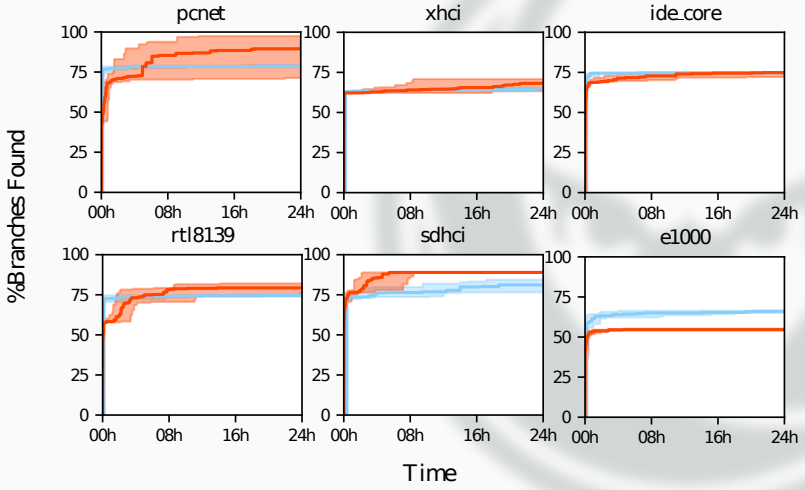


Graph Mutations



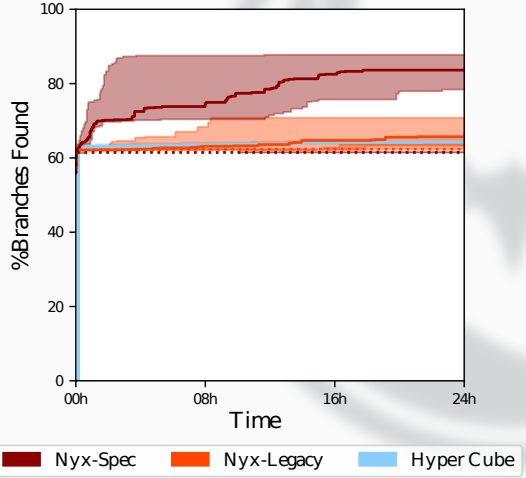
Evaluation

Hyper-Cube Spec



Complex Spec

QEMU 5.0.0 (xHCI)



Case Study: xHCI

bhyve: xHCI infinite loop

in pci_xhci_comple_commands

Case Study: xHCI

bhyve: xHCI infinite loop

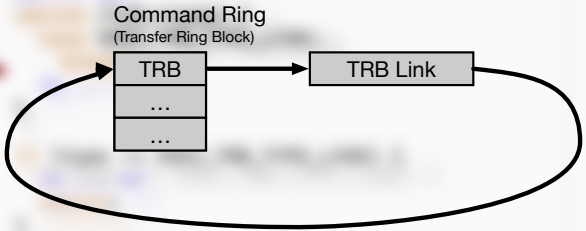
in pci_xhci_comple_commands

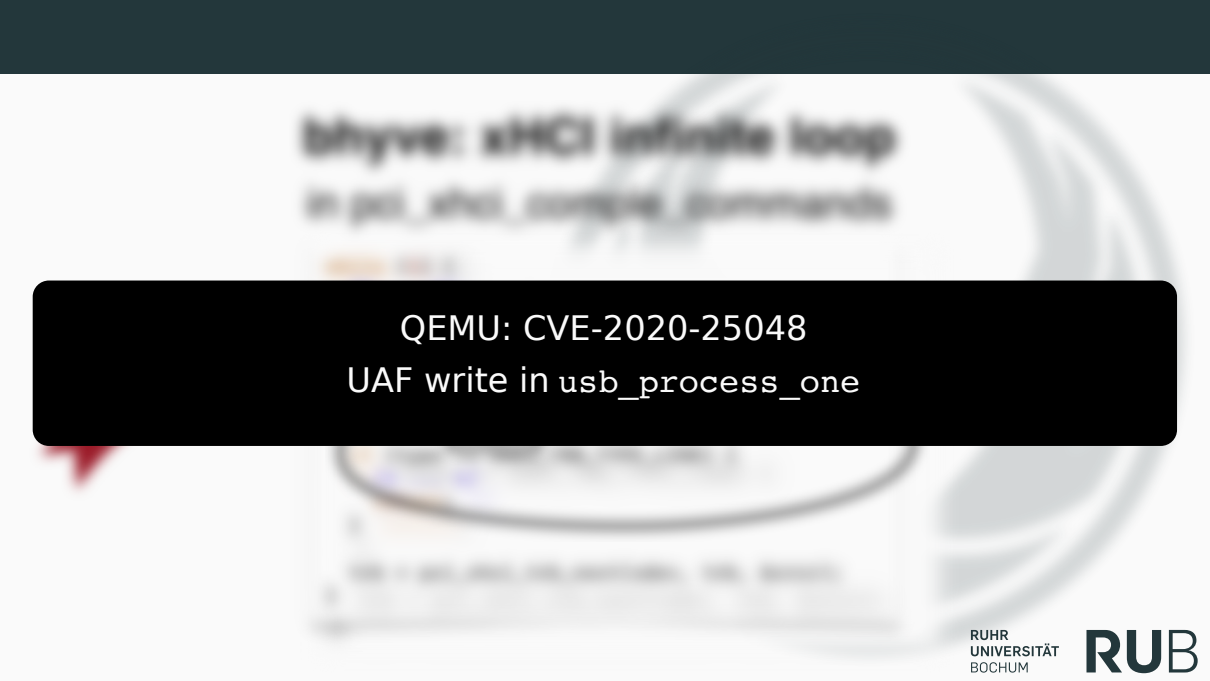
```
while (1) {
  /* ... */
  switch (trb->type) {
    case XHCI_TRB_TYPE_LINK:
      break;
    /* ... */
  }

  if (type != XHCI_TRB_TYPE_LINK) {
    /* ... */
    return;
  }

  trb = pci_xhci_trb_next(xdev, trb, &crcr);
}
```

Case Study: xHCI

bhyve: xHCI infinite loop

in pci_xhci_comple_commands





QEMU: CVE-2020-25048

UAF write in `usb_process_one`

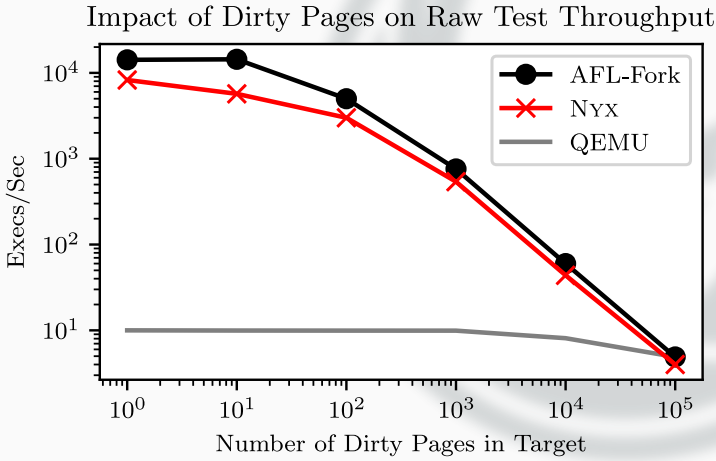
Findings

KVM/QEMU (5.0.0)

FreeBSD bhyve (12.1-RELEASE)

44
Bugs

QEMU-PT vs. AFL Forkserver



Conclusion

Conclusion

- **Fuzz Hypervisors** & Everything Below
- **Outperforms** Fast Blind Fuzzers
- **Full-System** Coverage Fuzzing

Thank You!

Q & A

sergej.schumilo@rub.de