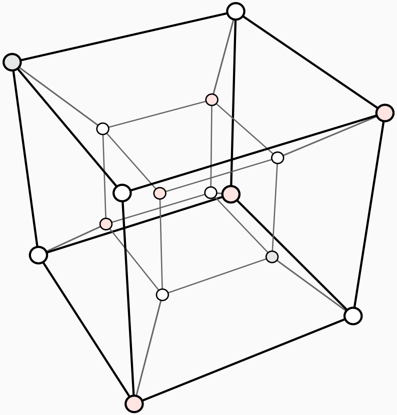


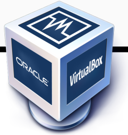
Hyper-Cube

High-Dimensional Hypervisor Fuzzing

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

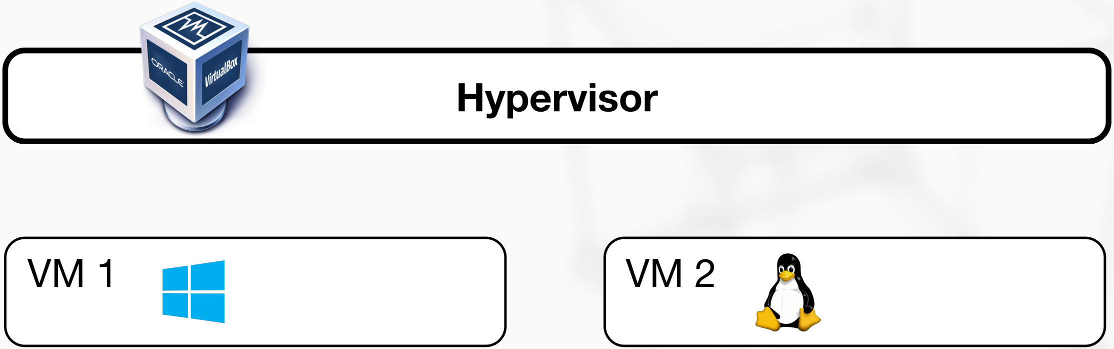
Chair for Systems Security
Ruhr-Universität Bochum

Motivation

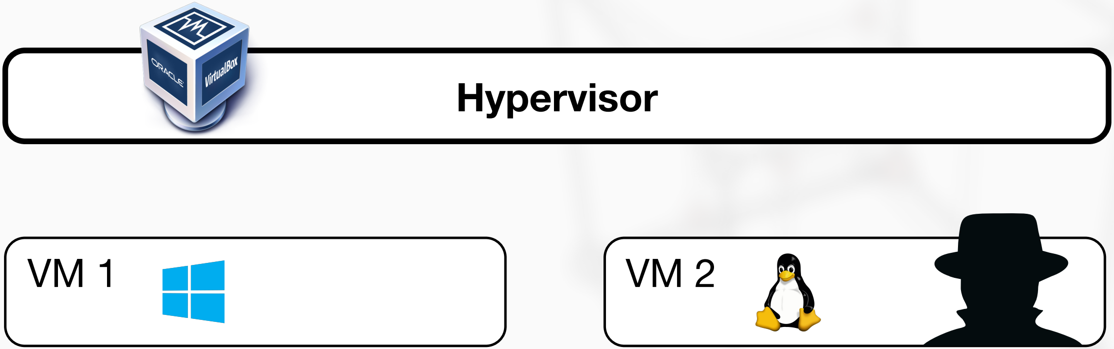


Hypervisor

Motivation



Motivation



Hypervisor

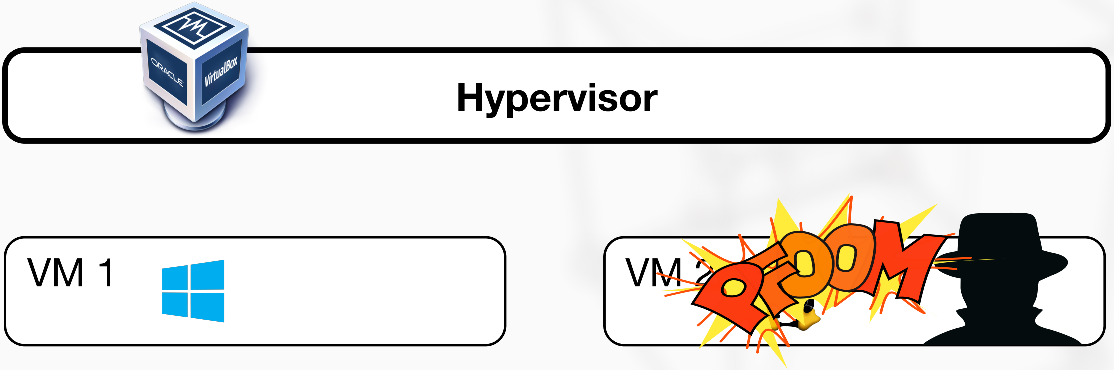
VM 1

VM 2

Malicious Guest

(Privileged; Running in Ring-0)

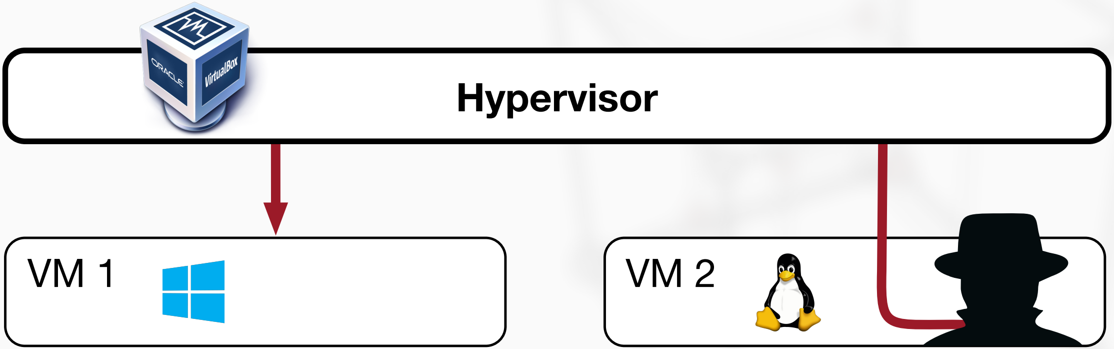
Motivation



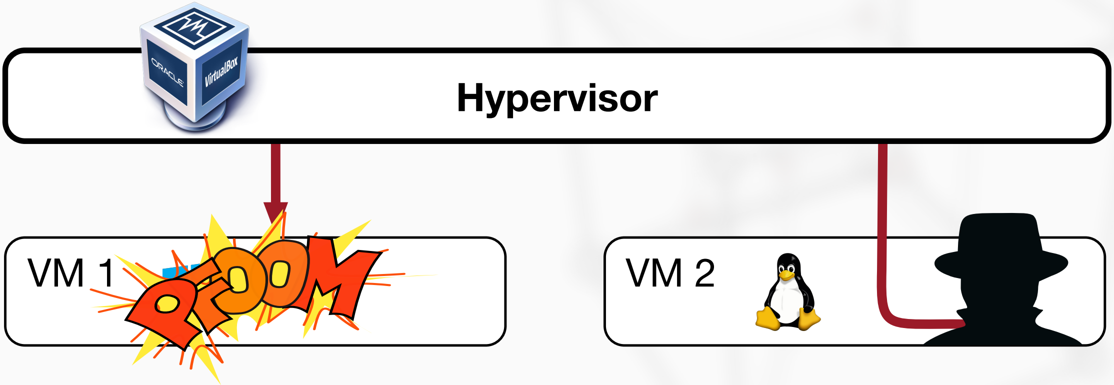
Local VM DoS

(Crash or Deadlock)

Motivation



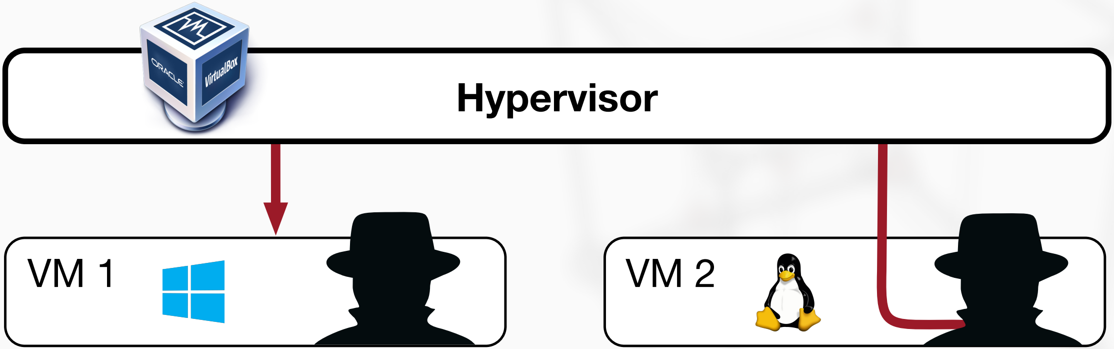
Motivation



Virtual Machine DoS

(Crash or Deadlock)

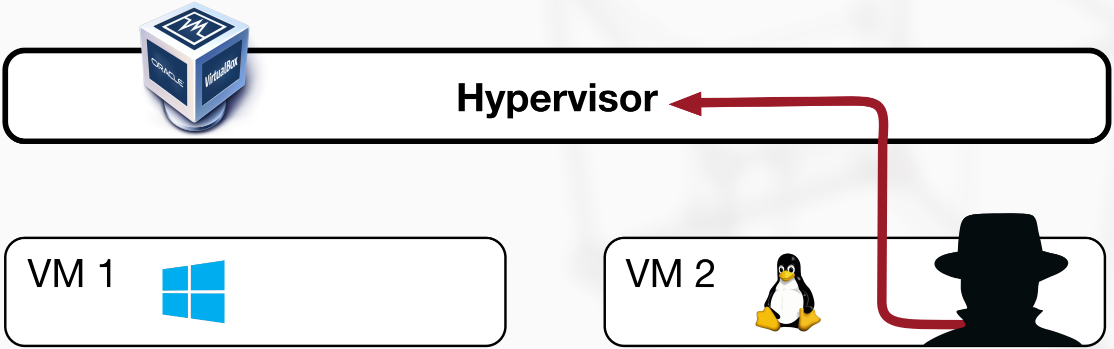
Motivation



Virtual Machine Escape

(Other Guest)

Motivation

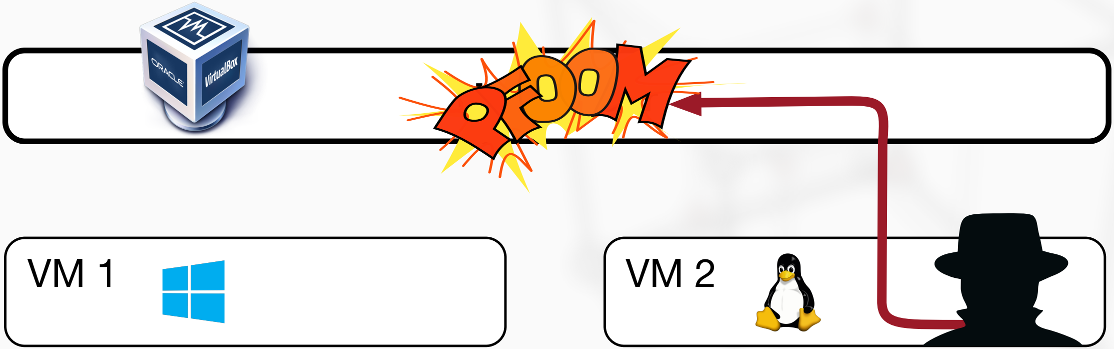


Hypervisor

VM 1

VM 2

Motivation



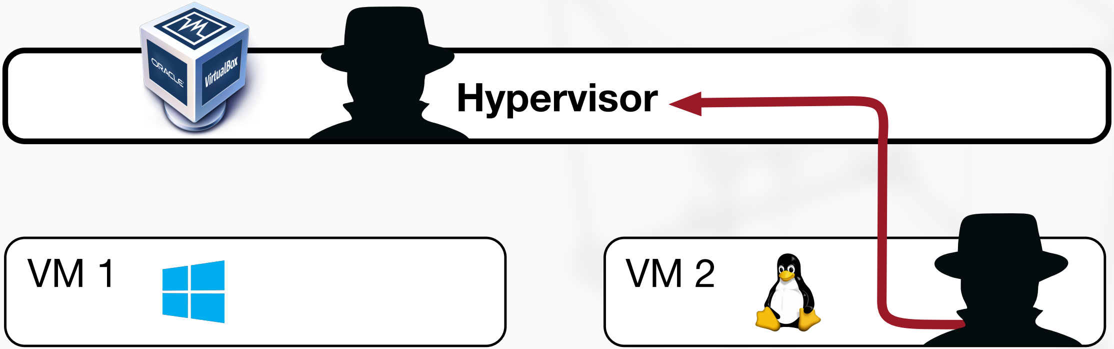
VM 1

VM 2

Host DoS

(Kernel Panic or Deadlock)

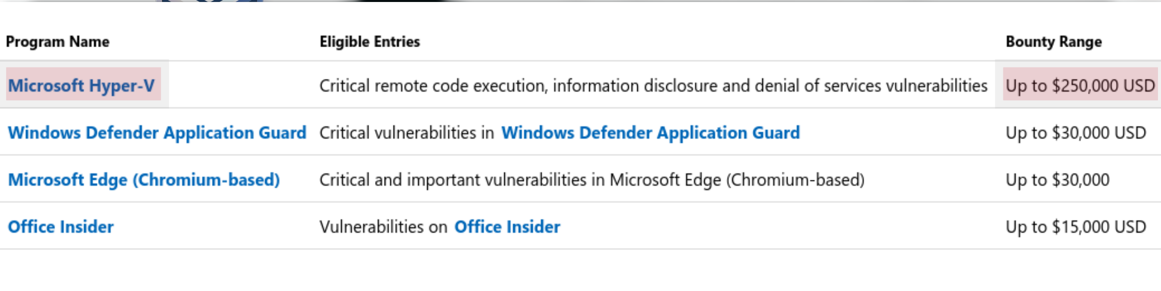
Motivation



Virtual Machine Escape

(Host)

Motivation



Program Name	Eligible Entries	Bounty Range
Microsoft Hyper-V	Critical remote code execution, information disclosure and denial of services vulnerabilities	Up to $250,000 USD
Windows Defender Application Guard	Critical vulnerabilities in Windows Defender Application Guard	Up to $30,000 USD
Microsoft Edge (Chromium-based)	Critical and important vulnerabilities in Microsoft Edge (Chromium-based)	Up to $30,000
Office Insider	Vulnerabilities on Office Insider	Up to $15,000 USD

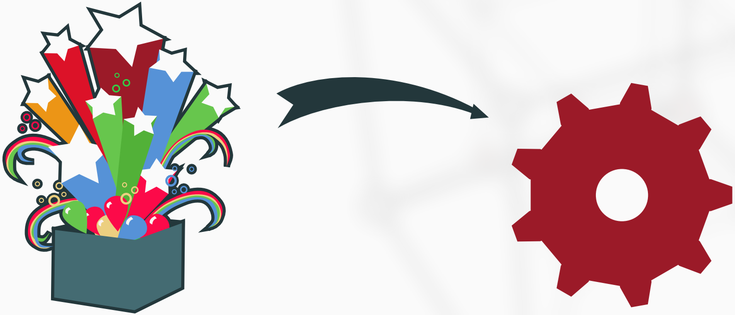
Virtual Machine Escape (Host)

Challenge

Challenge

Fuzzer of your Choice

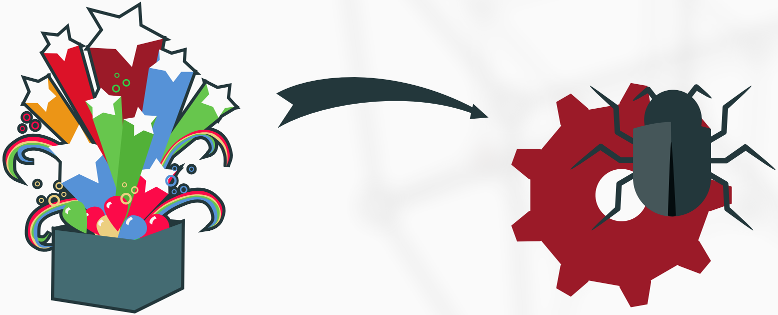
Challenge



Fuzzer of your Choice

Target Software

Challenge



Fuzzer of your Choice

Target Software

Challenge

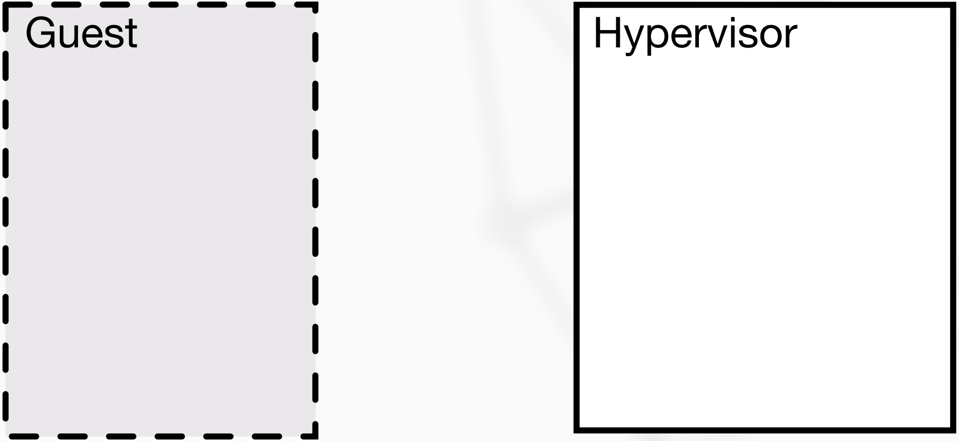
User Space Fuzzing

Challenge

Hypervisor Fuzzing

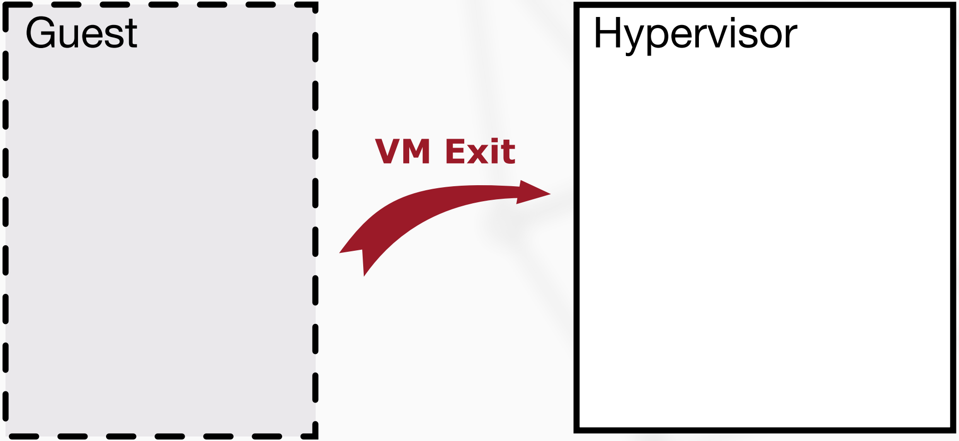
Attack Surface

Hypervisor Attack Surface



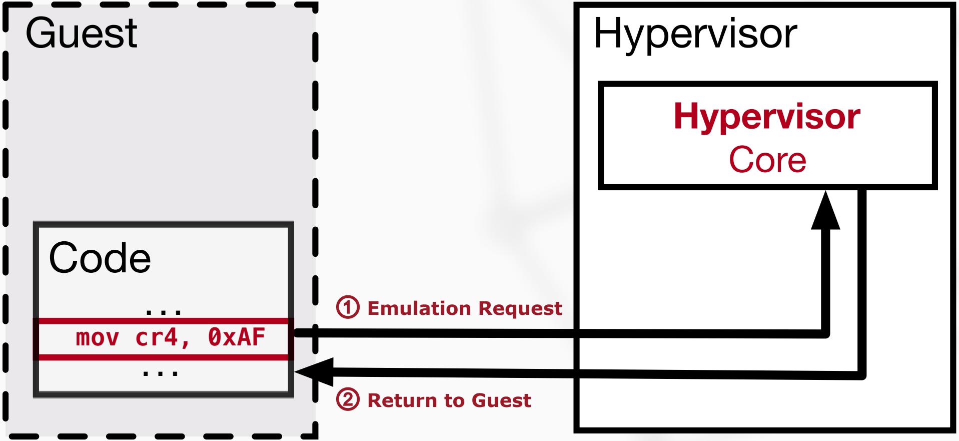
Hypervisor Attack Surface

Trap and Emulate



Hypervisor Attack Surface

Privileged Instructions



```
...
mov cr4, 0xAF
...
```

Hypervisor Attack Surface

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

Implementation

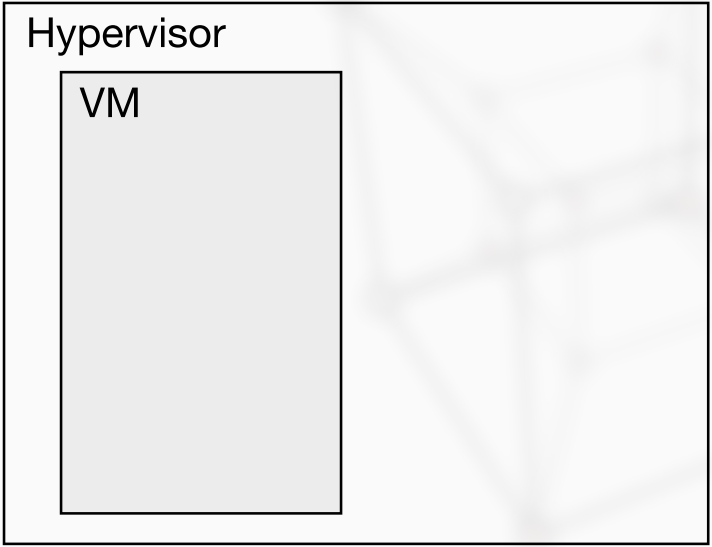
Design Goals

- x86 Hypervisor Agnostic
- Blackbox Fuzzing with High Througput
- High-Dimensional in Terms of
    - ➢ **Interfaces**
    - ➢ **Operations**

Our Approach

Hypervisor

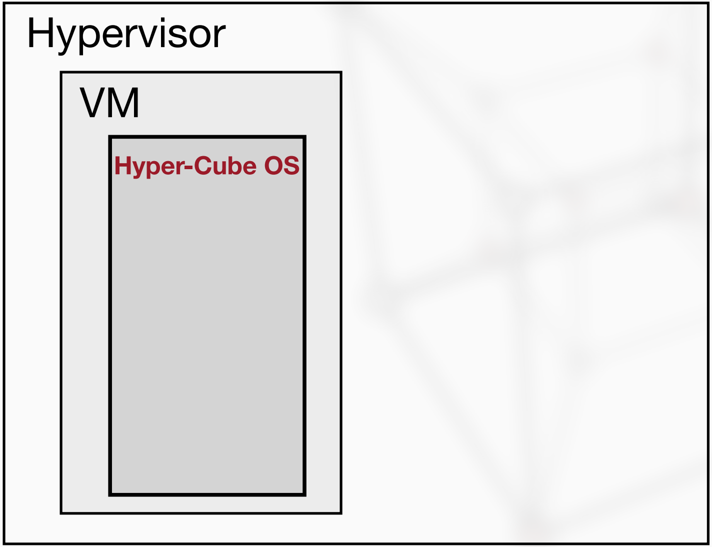
Our Approach



Hypervisor

VM

Our Approach

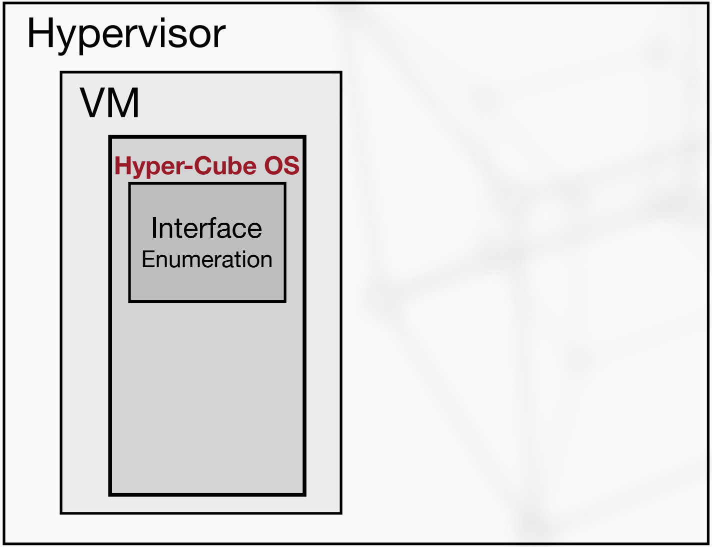


Hypervisor

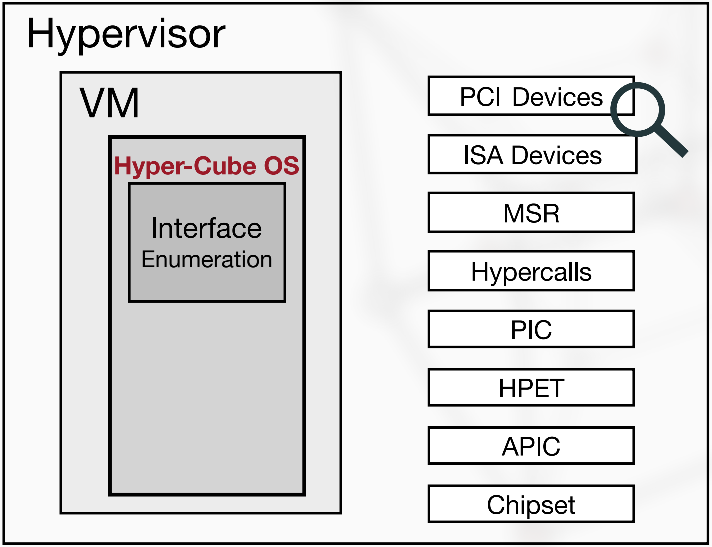
VM

Hyper-Cube OS

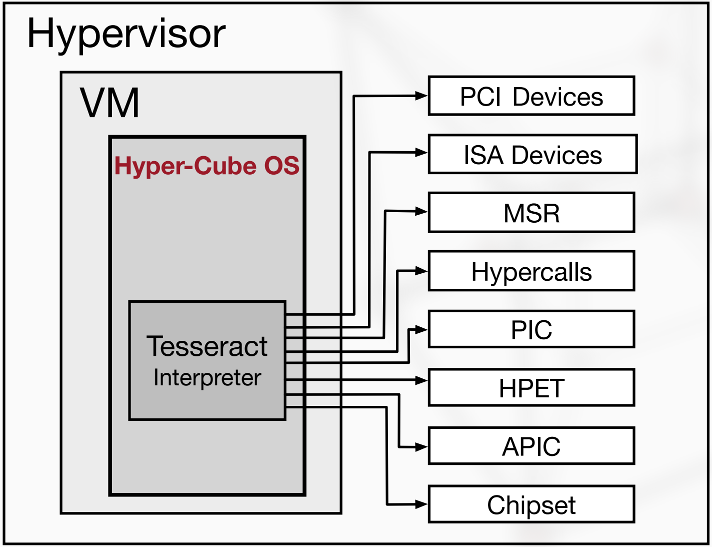
Our Approach



Our Approach



Our Approach



Tesseract Handlers

memset_mmio

write_msr

reads_io

writes_io

write_io

kvm_hypercall

memset_io

read_io

xor_io

read_mmio

write_mmio

io_write_scratch_ptr

reads_mmio

writes_mmio

xor_mmio

vmport

bruteforce_mmio

mmio_write_scratch_ptr

bruteforce_io

PRNG Stream

```
           ...
0120:2fff 1c27 ab47 5700
0128:adf2 3d60 092f 5488
0130:ec2d 9d1a 029d 56fd
0138:e0d1 a275 1f56 1d28
0140:ea78 a2fa db07 d60d
0148:1288 3a5a 91f9 1756
0150:1cae 31ad 9b9c 938e
0158:2a33 f597 6615 e267
0160:0117 1f16 b440 8a86
0168:9154 5b55 e4ca 9e3d
0170:9d19 ae79 efac e500
0178:8cdf 8c00 9a83 df76
0180:91fe d779 026c 2e2b
0188:9137 1ef8 eea3 d29c
0190:1789 5938 a36f 718a
0198:81e4 678c 20f5 fa0b
01a0:774d 07f1 cee3 62bc
01a8:d845 bc86 7631 6eac
           ...
```

Tesseract Interpreter

PRNG Stream

```
...
0120: 2fff 1c27 ab47 5700
0128: adf2 3d60 092f 5488
0130: ec2d 9d1a 029d 56fd
0138: e0d1 a275 1f56 1d28
0140: ea78 a2fa db07 d60d
0148: 1288 3a5a 91f9 1756
0150: 1cae 31ad 9b9c 938e
0158: 2a33 f597 6615 e267
0160: 0117 1f16 b440 8a86
0168: 9154 5b55 e4ca 9e3d
0170: 9d19 ae79 efac e500
0178: 8cdf 8c00 9a83 df76
0180: 91fe d779 026c 2e2b
0188: 9137 1ef8 eea3 d29c
0190: 1789 5938 a36f 718a
0198: 81e4 678c 20f5 fa0b
01a0: 774d 07f1 cee3 62bc
01a8: d845 bc86 7631 6eac
...
```

Robust Interpretation

Opcode Handler

vmport(0xbd4,0x10ea)
memset_io(0x426,0xce0,0x9dc,0xca8)

writes_mmio(0xec8,0xad,0x10ac,0x7e9)

bruteforce_mmio(0xce4,0xdfa,0xe31,0x322)

writes_io(0x4bb,0xb8,0xeb1,0x401)

memset_mmio(0x128,0xa73,0x2b3,0xa84)
read_mmio(0xbf3,0x907)

bruteforce_io(0x5c4,0x49a,0x94f,0xb1c)

xor_mmio(0x54b,0xa00,0xb51)

Evaluation

Tested Hypervisors

FreeBSD bhyve (12.0-RELEASE)

VirtualBox (5.1.37_Ubuntu r122592)

Parallels Desktop (14.1.3)

KVM/QEMU (4.0.1-rc4)

Intel ACRN (29360 Build)

VMware Fusion (11.0.3)

Results

Assert Failures	25
Null-Pointer Dereferences	13
Memory-Corruptions	8
Div-By-Zero (FP Exceptions)	5
Deadlocks	4

55
Bugs

Case Study: bhyve

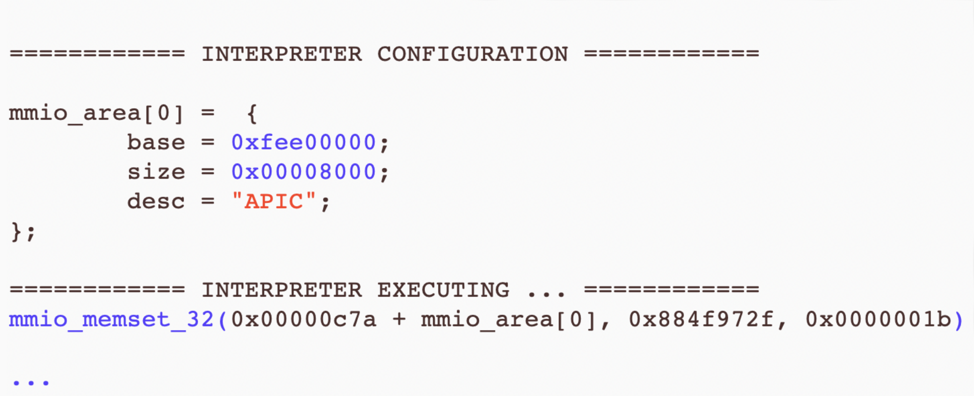
CVE-2019-12071

FreeBSD Kernel Denial of Service via Privileged Guest

Case Study: bhyve

CVE-2019-12071

FreeBSD Kernel Denial of Service via Privileged Guest



```
============ INTERPRETER CONFIGURATION ============

mmio_area[0] =  {
        base = 0xfee00000;
        size = 0x00008000;
        desc = "APIC";
};

============ INTERPRETER EXECUTING ... ============
mmio_memset_32(0x00000c7a + mmio_area[0], 0x884f972f, 0x0000001b)

...
```

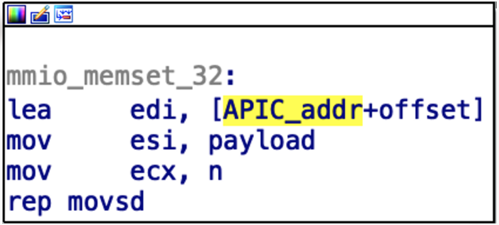
Case Study: bhyve

CVE-2019-12071

FreeBSD Kernel Denial of Service via Privileged Guest

```
mmio_memset_32(0x00000c7a + mmio_area[0], 0x884f972f, 0x0000001b)
```

Translates to



```
mmio_memset_32:
lea      edi, [APIC_addr+offset]
mov      esi, payload
mov      ecx, n
rep movsd
```

Case Study: bhyve

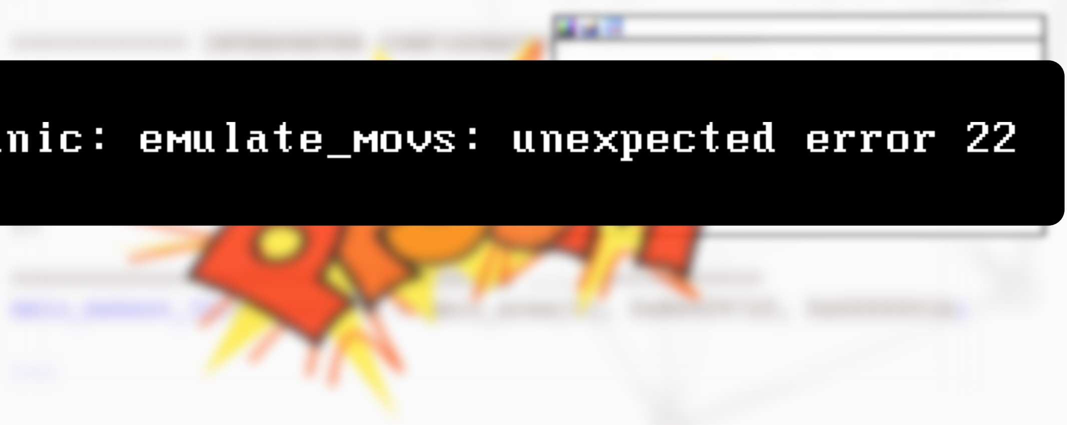
CVE-2019-12071

FreeBSD Kernel Denial of Service via Privileged Guest

Case Study: bhyve

CVE-2019-12071

FreeBSD Kernel Denial of Service via Privileged Guest



```
panic: emulate_movs: unexpected error 22
```

CVE Rediscovery

CVE-2015-3456

VENOM Vulnerability

CVE Rediscovery

CVE-2015-3456

VENOM Vulnerability

TCG Mode: **5.8** sec

(average time in seconds over 20 runs each)

CVE Rediscovery

CVE-2015-3456

VENOM Vulnerability

TCG Mode:	**5.8** sec
KVM Mode:	**49.7** sec

(average time in seconds over 20 runs each)

Hyper-Cube vs. VDF

VDF: Targeted Evolutionary Fuzz Testing of Virtual Devices

RAID 2017: Research in Attacks, Intrusions, and Defenses

- AFL-based Fuzzing Approach
- Fuzzing of Specific Device Emulators

Hyper-Cube vs. VDF

Fuzzing 15 Device Emulators (QEMU-2.5.0)

Hyper-Cube vs. VDF

Fuzzing 15 Device Emulators (QEMU-2.5.0)

HYPER CUBE { **13**/15 More Coverage

VDF { **2**/15 More Coverage

Hyper-Cube vs. VDF

Fuzzing 15 Device Emulators (QEMU-2.5.0)

HYPER CUBE

- **13**/15 More Coverage
- **9**/15 Crashed

VDF

- **2**/15 More Coverage
- **4**/15 Crashed

Hyper-Cube vs. VDF

Fuzzing 15 Device Emulators (QEMU-2.5.0)

HYPER CUBE

- **13**/15 More Coverage
- **9**/15 Crashed
- **10 Minutes** Each

VDF

- **2**/15 More Coverage
- **4**/15 Crashed
- **≈ 60 Days Each**

Conclusion

Conclusion

- **Novel Technique** to Fuzz Hypervisors
- **Outperforms** Coverage-Guided Fuzzers
- **Full-System** Fuzzing

Thank You!

Q & A