Hypervisor security

Evgeny Yakovlev, DEFCON NN, 2017

whoami

- Low-level development in C and C++ on x86
- UEFI, virtualization, security
- Jetico, Kaspersky Lab
- QEMU/KVM developer at Virtuozzo

Agenda

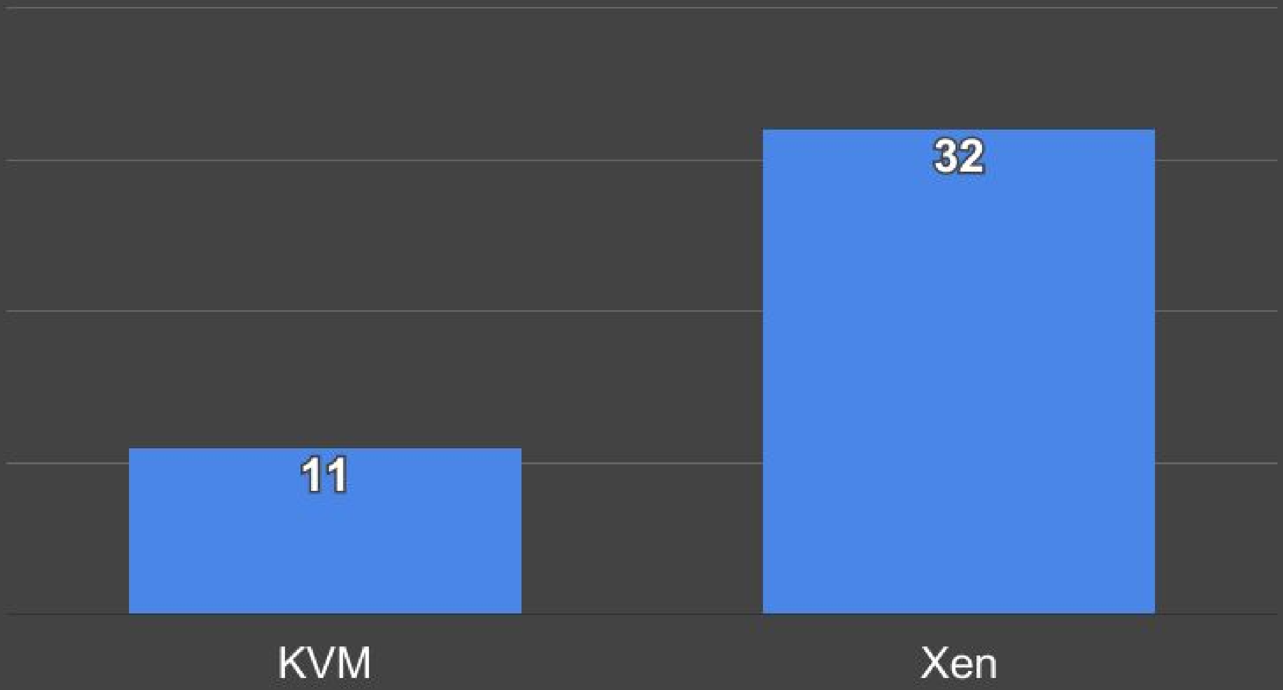
- Why hypervisor security
- How hypervisors work
- Threat model & attack surface
- Virtualization bugs
- Questions

Why hypervisor security

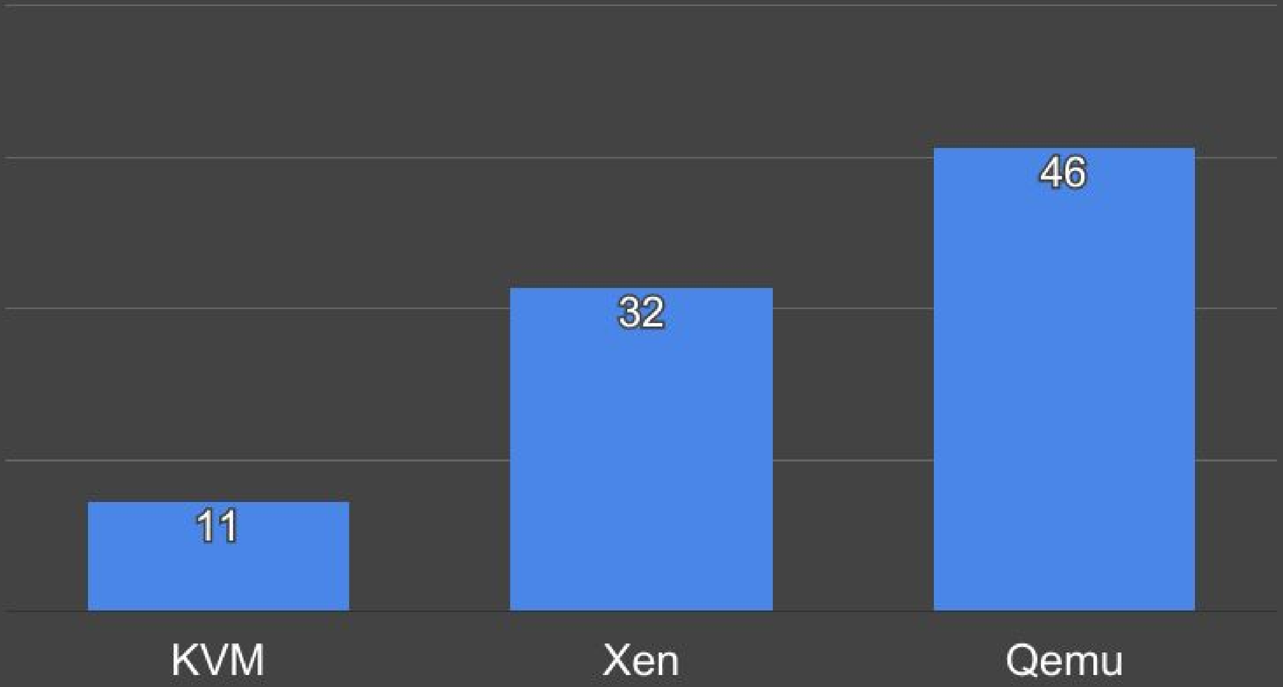
Virtualization is relied upon in many areas:

- Server and cloud
- Embedded and automotive
- Desktop security
- R&D

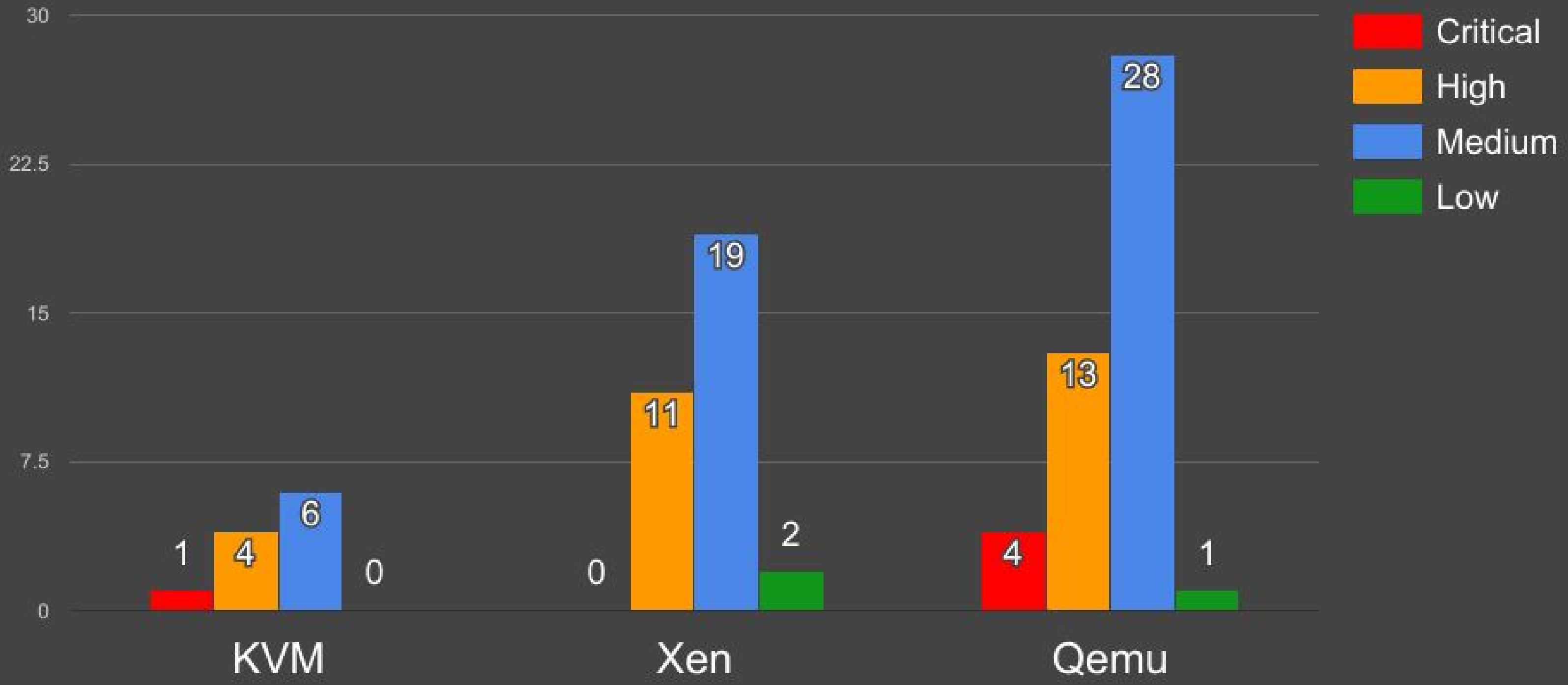
Hypervisor CVEs 2016



Hypervisor CVEs 2016 + Qemu



CVEs by severity 2016 + Qemu



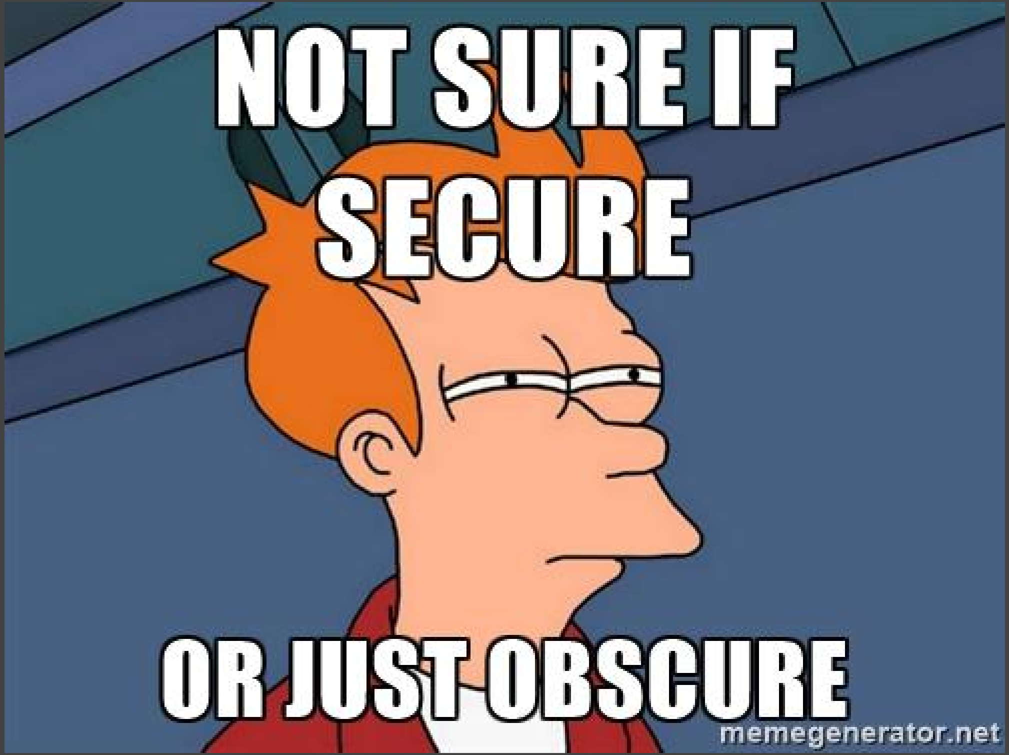
CVE statistics summary

- KVM has smallest code base
- Qemu is a device emulator - huge code base
- Both KVM and Xen rely on Qemu for device emulation

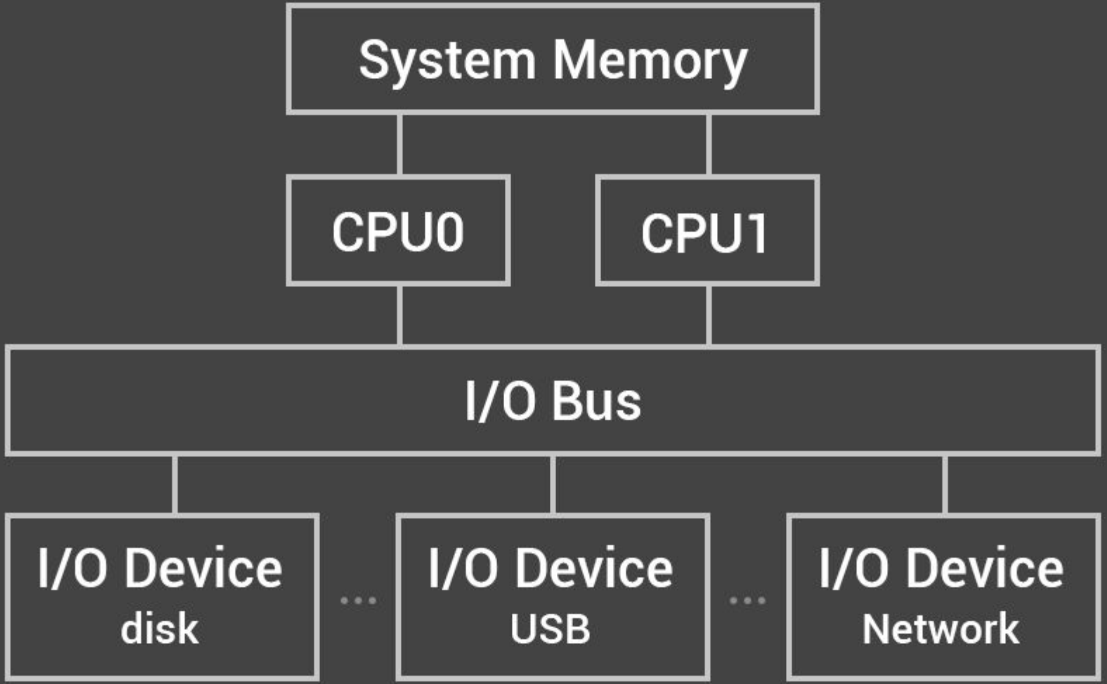
CVE statistics summary

- Microsoft HyperV has 0 reported CVEs
- VmWare ESXi has no reported VM escapes in 2016

CVE statistics summary



Computer system as an API



- CPUs (x86 ISA)
- Memory (segmentation, paging)
- I/O (MMIO, PIO)
- Networking protocols

Emulation and Virtualization

Emulation

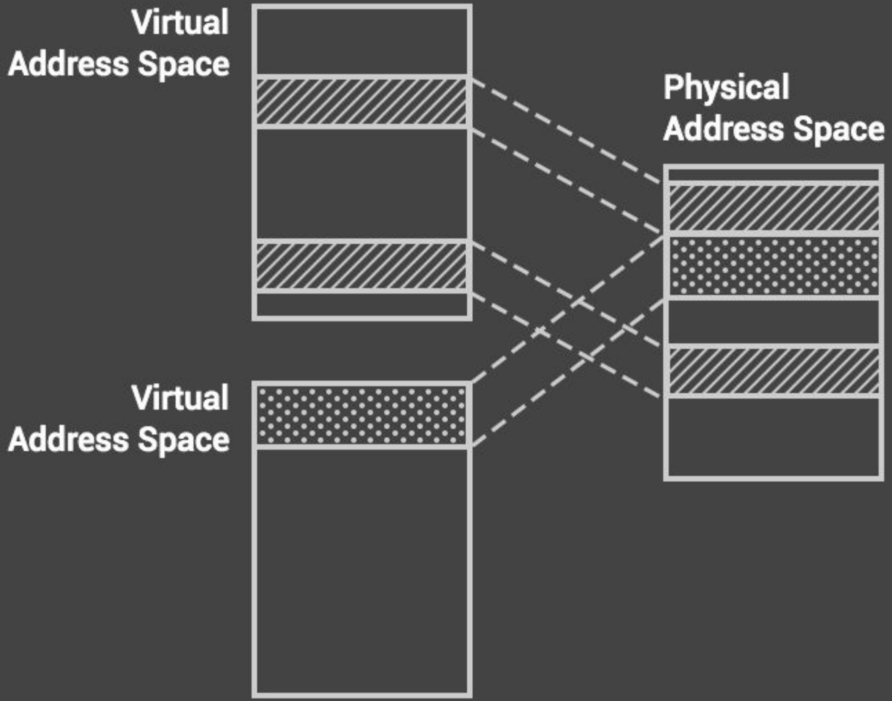
Imitation of a different system, usually software

Virtualization

Partitioning same system into multiple virtual instances

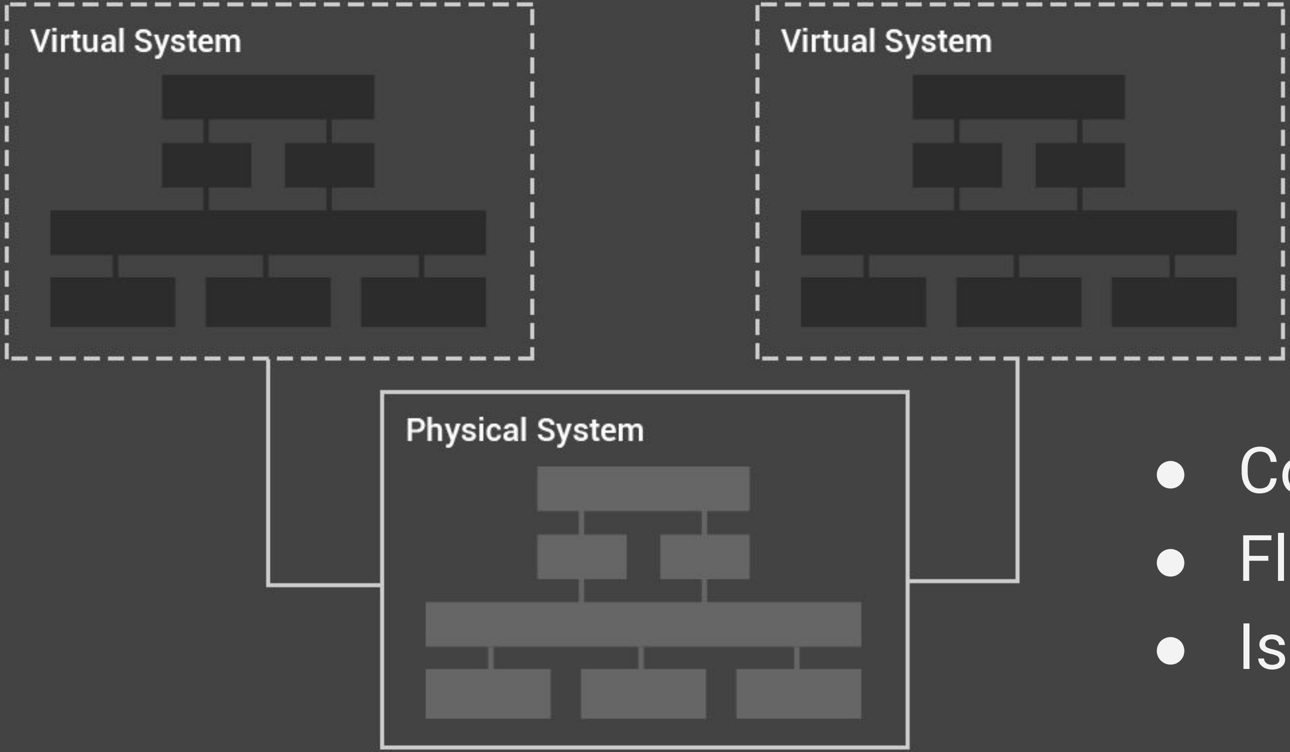
Both are implementation of a computer system API

Virtual memory, for example



An early example of resource virtualization — memory address space

Virtual systems

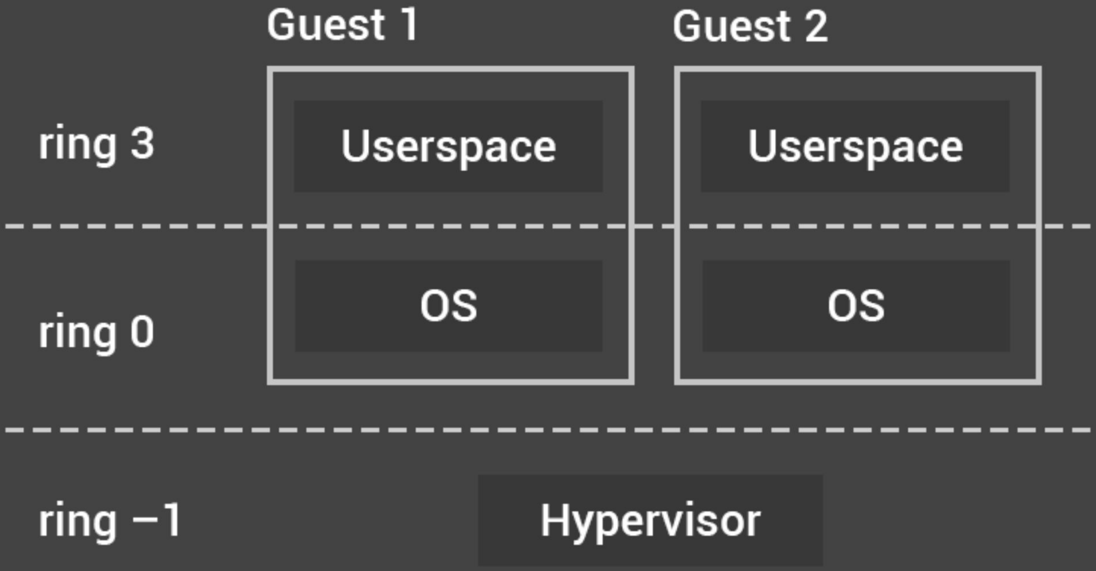


- Consolidation
- Flexibility
- Isolation

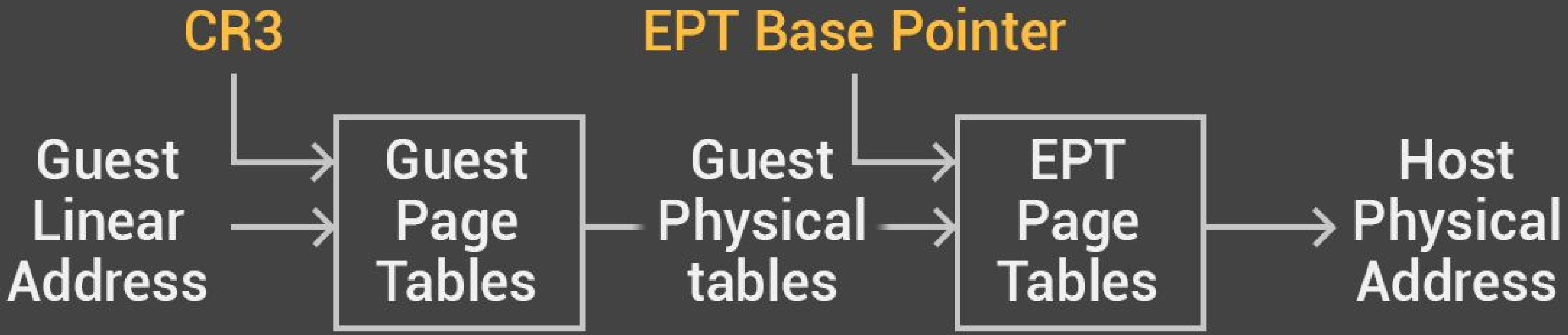
Hypervisor privileged role

- Drives virtualization hardware
- Executes in privileged CPU mode
- Manages VMs and platform resources.
- Emulates hardware requests
- Provide services to enlightened OS

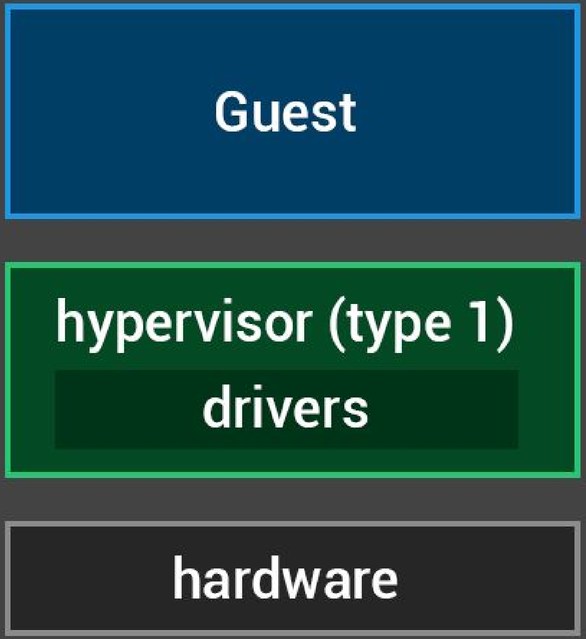
Hypervisor privileged role



Hypervisor privileged role

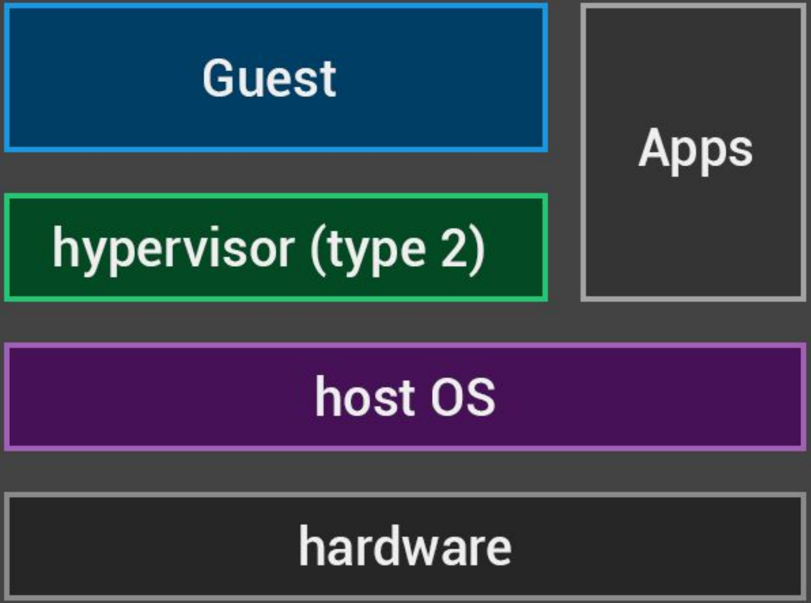


Type-1 hypervisor



- Hypervisor runs on bare-metal
- Can be a very small code base..
- .. but has to solve driver problem
- All OS kernels run in isolated environment and don’t touch hardware
- Xen, Hyperv, VMWare ESX

Type-2 hypervisor



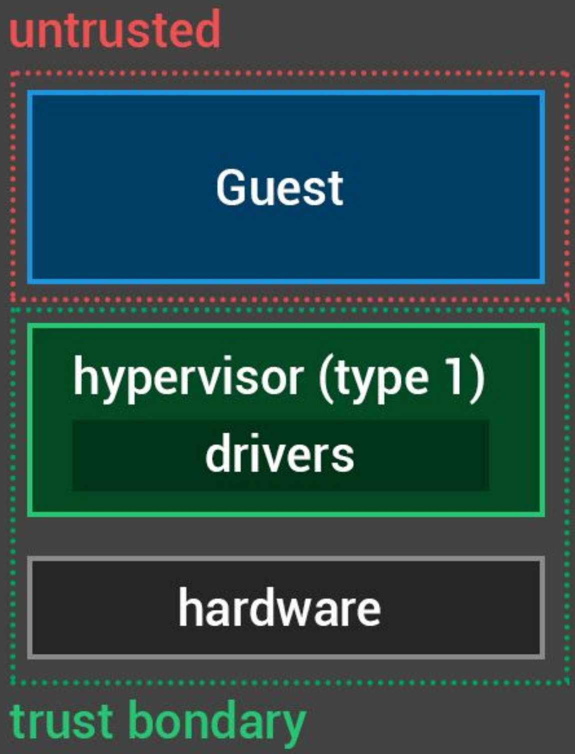
- Hypervisor is an OS component
- Host OS provides all drivers
- Huge trusted computing base.

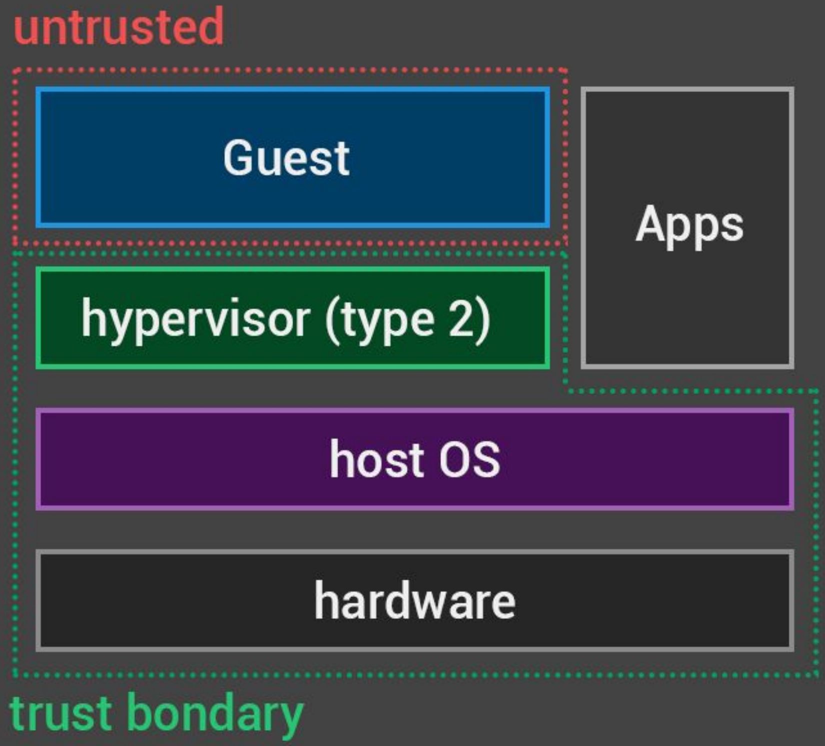
- KVM, VirtualBox, VMWare workstation

Threat model

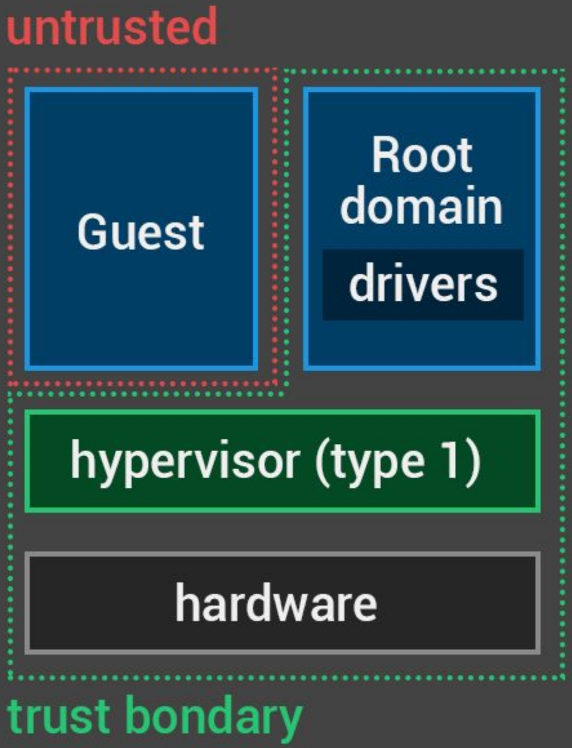
- Hypervisor is a privileged code base
- Hardware and firmware are usually trusted
- VMs should never be trusted by hypervisor
- VM may not trust hypervisor or other VMs

Type-1 vs Type-2 trust boundary





Type-1 root domain trust



- Type-1 often runs a special guest, a root domain
- More trusted than normal guest
- Runs OS kernel to drive host hardware
- Runs device (para-)virtualization stack
- Xen Dom0

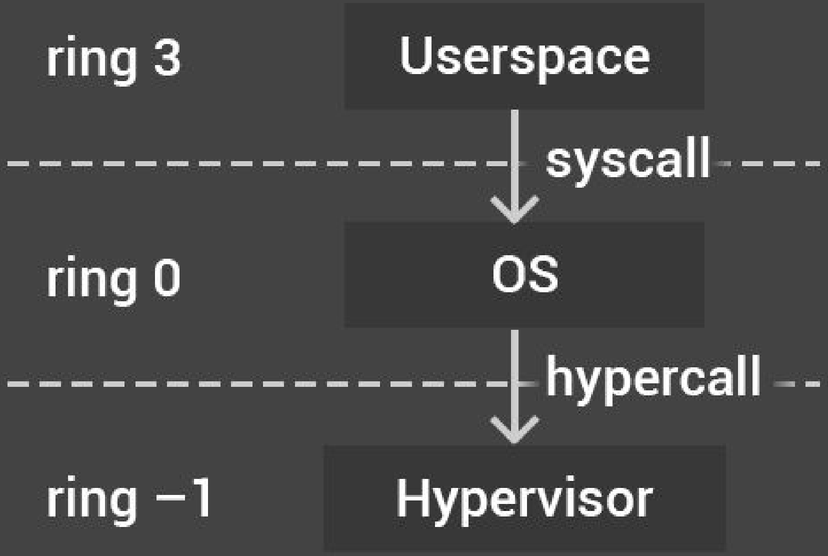
Security threats

- Denial of service
- Privilege escalation (VM-local or VM-host)
- Information leak

Attack surface

- Hypercalls
- MMIO and device emulation
- Paravirtualization
- Side-channels

Hypercalls



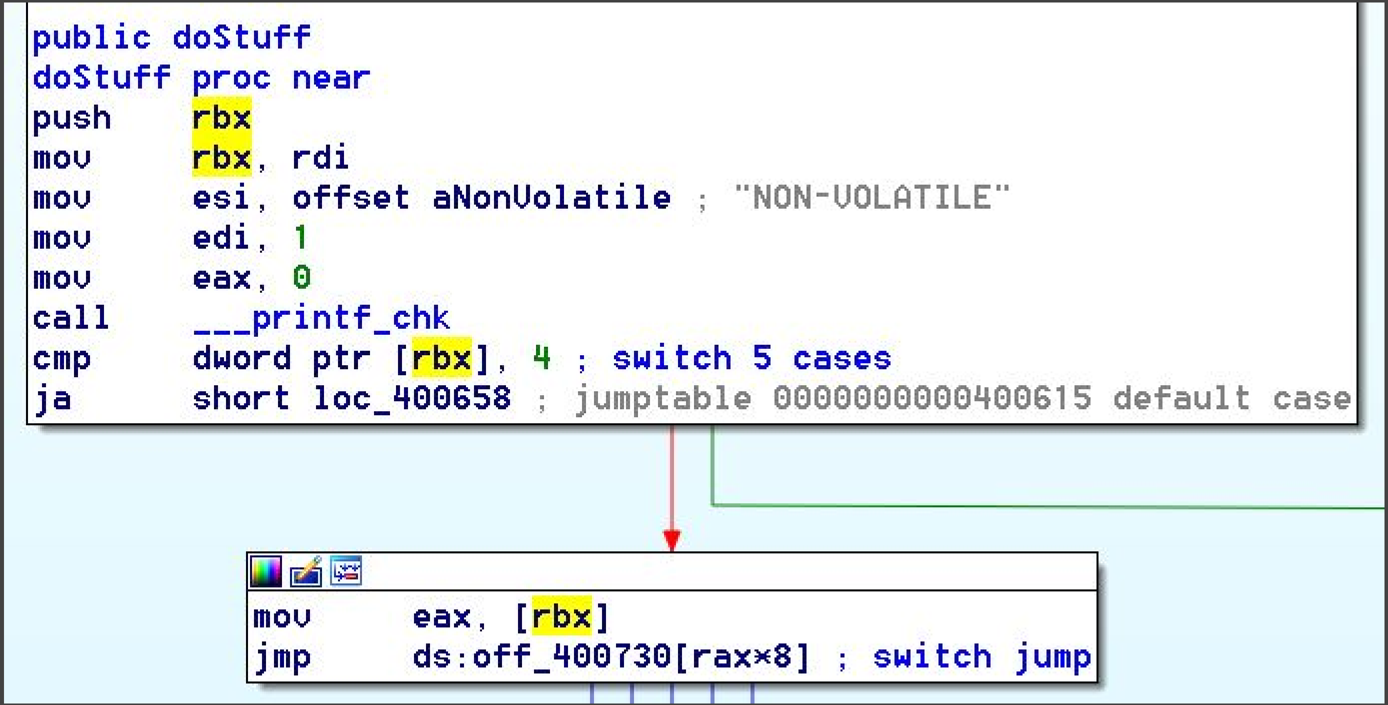
Hypercalls are services for hypervisor-aware guest:

- Virtual hardware and events
- Memory management
- Cross-VM communication
- Crash handling
- Security

Hypercalls

- Relatively small surface, can be fuzz-tested
- Usually available only in guest ring 0
- Not a lot of issues, especially in HWVMs
- Be wary of double-fetch bugs

Double-fetch bugs



```
public doStuff
doStuff proc near
push    rbx
mov     rbx, rdi
mov     esi, offset aNonVolatile ; "NON-VOLATILE"
mov     edi, 1
mov     eax, 0
call    ___printf_chk
cmp     dword ptr [rbx], 4 ; switch 5 cases
ja      short loc_400658 ; jumptable 0000000000400615 default case
```

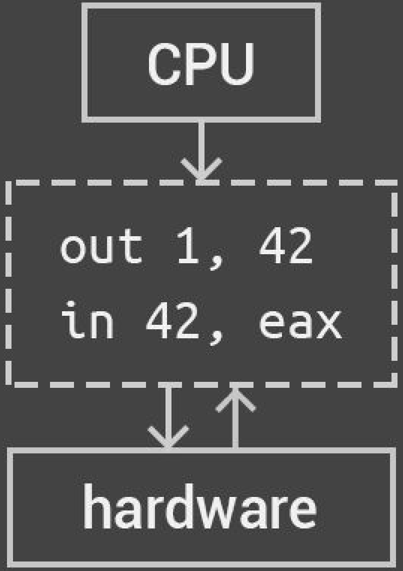
```
mov     eax, [rbx]
jmp     ds:off_400730[rax*8] ; switch jump
```

http://tkeetch.co.uk/blog/?p=58

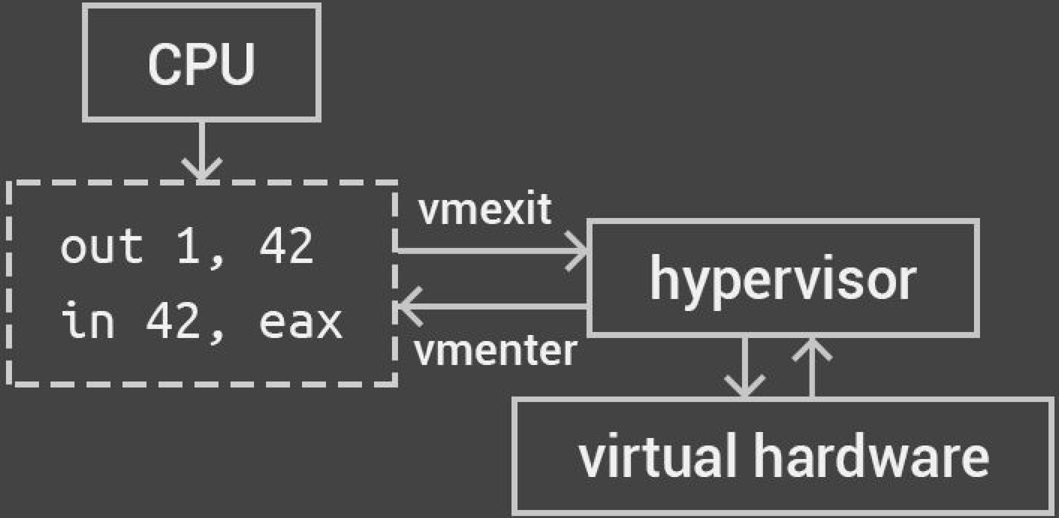
Device emulation

- Huge code base with bad track record
- Obscure CPU features and registers
- Complicated hardware with dodgy corner cases
- MMIO instruction decoding

Device emulation: vmexit

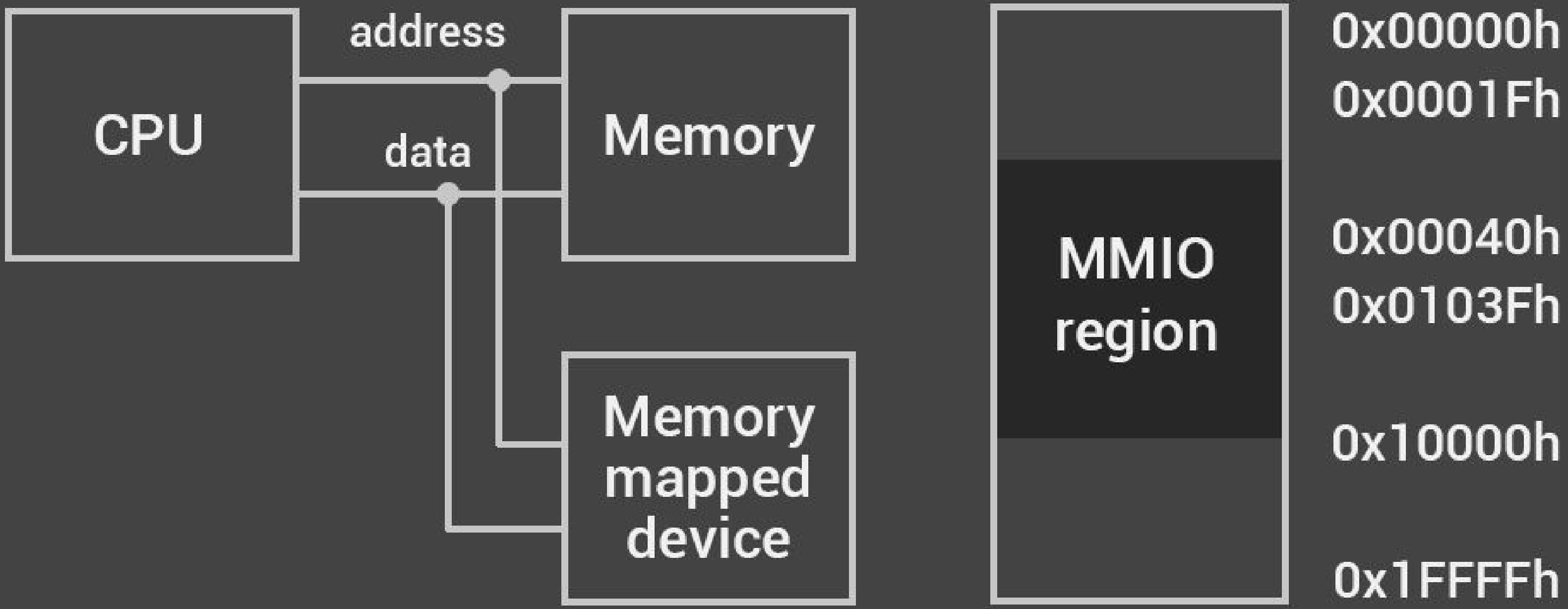


```
out 1, 42
in 42, eax
```



```
out 1, 42
in 42, eax
```

Device emulation: MMIO



Device emulation

- KVM MMIO emulation - 5k LoC, 50% KVM CVEs
- Most of Xen CVEs
- Most of qemu CVEs
- Google even decided to roll its own emulator

Device emulation: Dark Portal

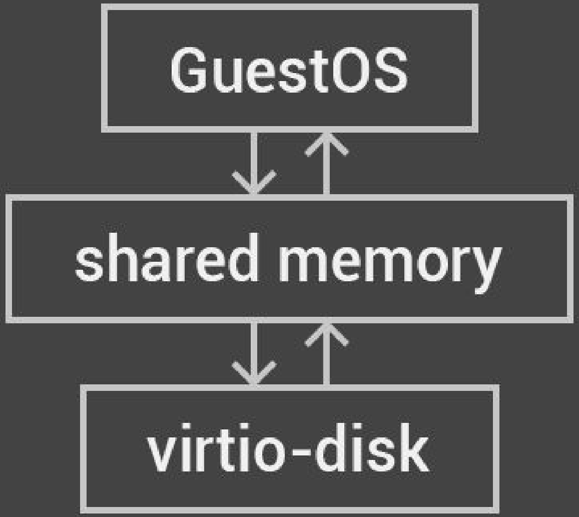
VGA VM escape:
http://www.powerofcommunity.net/poc2016/wei.pdf

- Attacker controls VGA bank offset register
- Bank offset used as unbounded offset into array
- Attacker can read or write 32bit value anywhere

Paravirtualization

- Vmexit is a huge performance hit
- Hardware emulation using vmexits is slow
- Shared memory is fast
- Let’s build virtual hardware protocol on shared memory!
- Virtio, VMBus, Xen

Paravirtualization



- Virtio-disk talks to guest driver through shared memory
- More performance
- But guest OS needs a specific driver
- Also some interesting security challenges

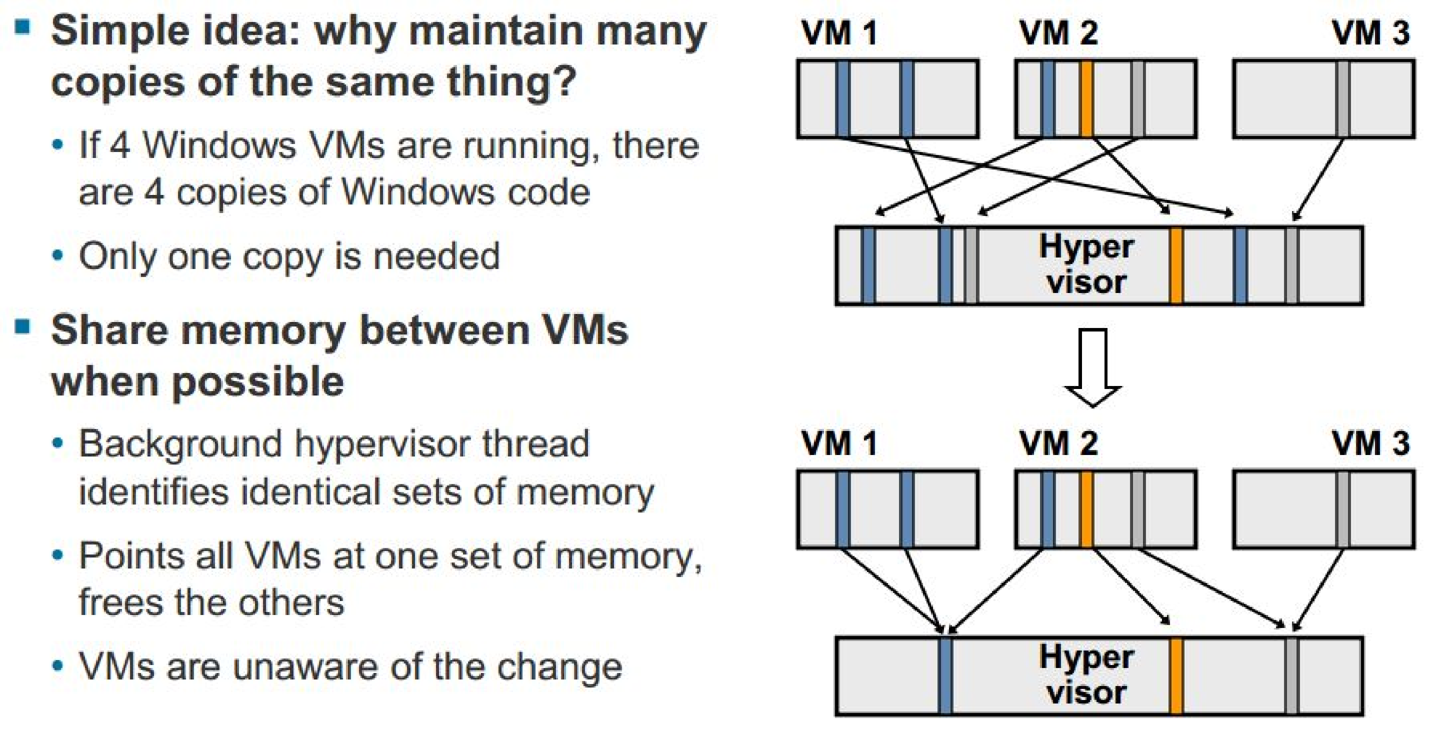
Paravirtualization

- Shared memory simplified device interface and implementation
- Ring buffers on shared memory vulnerable to double-fetch bugs
- Xenpwn: https://youtu.be/XOb--niy_0M

Side-channels

- Not an attack on hypervisor itself
- Co-located VM information leaks
- Hardware optimizations (CPU caches, DRAM timings)
- Memory deduplication

Memory deduplication



- **Simple idea: why maintain many copies of the same thing?**
  - If 4 Windows VMs are running, there are 4 copies of Windows code
  - Only one copy is needed
- **Share memory between VMs when possible**
  - Background hypervisor thread identifies identical sets of memory
  - Points all VMs at one set of memory, frees the others
  - VMs are unaware of the change

VM 1
VM 2
VM 3
Hyper visor
VM 1
VM 2
VM 3
Hyper visor

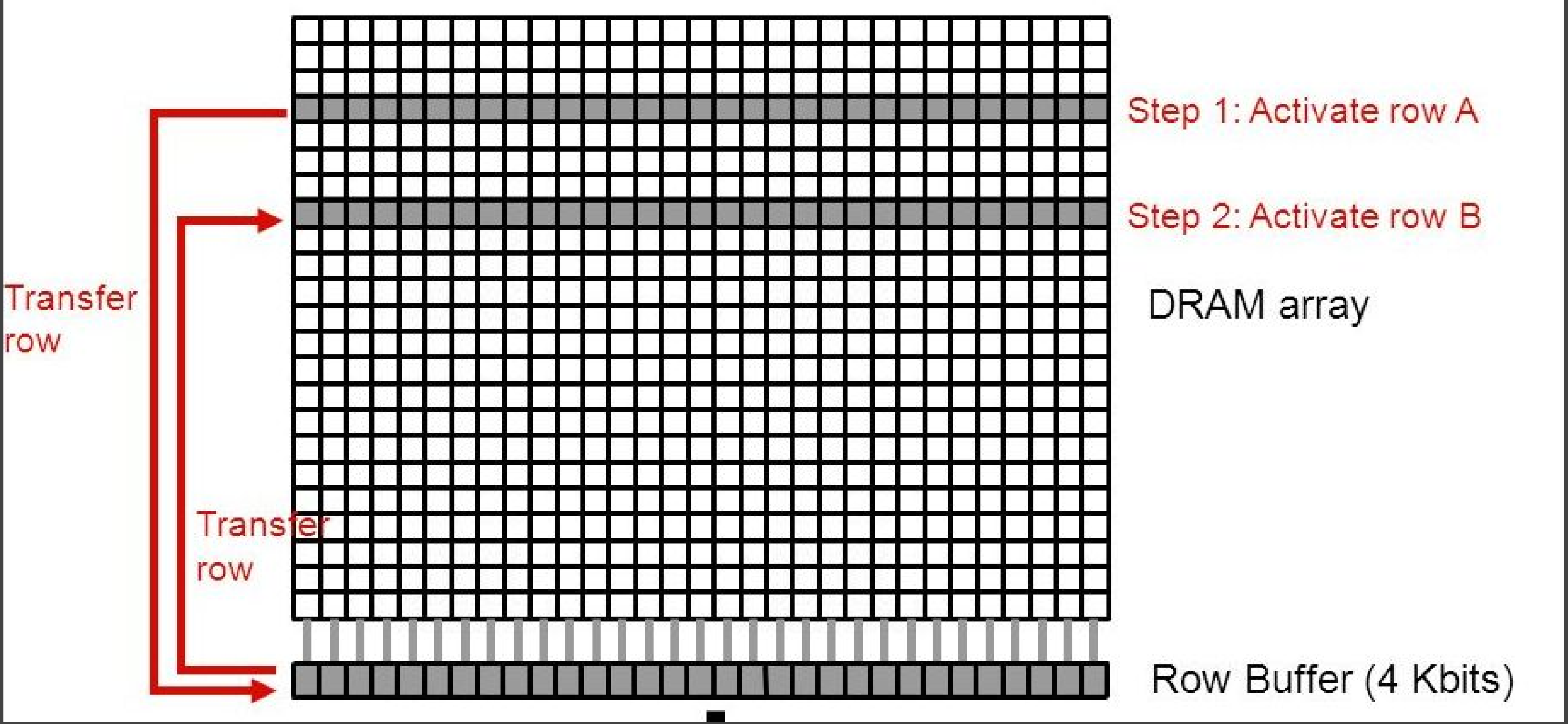
Memory deduplication

- Turns out deduplicated memory has measurable access time delays
- Attacker VM can “guess” co-located memory contents
- “CAIN: Silently Breaking ASLR in the Cloud” https://www.usenix.org/node/191961

Caches and DRAM timings

- CPU cache optimizes memory access
- “Evict and measure” attacks
  - Attacker VM evicts cache line
  - Victim VM fetches it back
  - Attacker VM measures evicted line access timing
- DRAM has an internal cache line too!

Caches and DRAM timings



Conclusion

- Hypervisors are getting more attention lately
- Device emulation is historically most vulnerable
- Paravirtualization is probably next big thing
- Co-localed leaks are very promising
- A lot of undiscovered problems in closed-source products

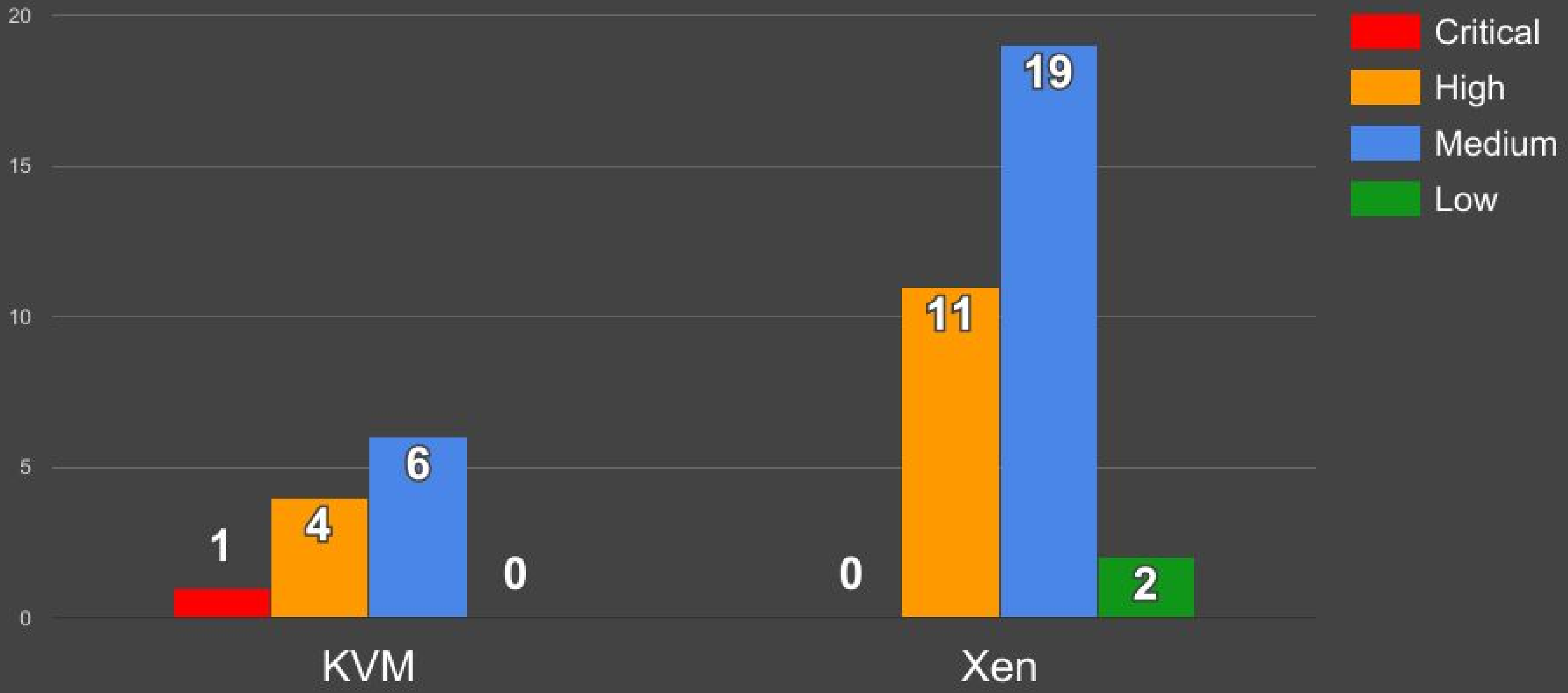
Thanks! Questions?

insoreiges at gmail dot com
https://github.com/warfish
https://www.facebook.com/evgeny.yakovlev.5268

Motivations

- I’m a developer and i build stuff
- I want my stuff to be secure
- Researchers make stuff more secure
- Let’s share knowledge

CVEs by severity 2016



Device emulation: XSA-190

http://xenbits.xen.org/xsa/advisory-190.html

“Xen 4.7.x and earlier does not properly honor CR0.TS and CR0.EM, which allows local x86 HVM guest OS users to read or modify FPU, MMX, or XMM register state information belonging to arbitrary tasks on the guest by modifying an instruction while the hypervisor is preparing to emulate it.”

Conclusion

- Moving code to user space (KVM split-IRQ chip)
- Fuzzing entry points and devices
- Live patching