Say Goodbye to Virtualization for a Safer Cloud

Dan Williams, Ricardo Koller, Brandon Lum
IBM T.J. Watson Research Center
Isolation is important for the cloud

![Diagram showing isolation in the cloud.](Image)
Isolation is important for the cloud

- Virtualization is the gold standard for isolation
Why are VMs the gold standard for isolation?

- Virtual machines use a low level interface to the host
Why are VMs the gold standard for isolation?

- A high level interface increases the chances of hitting a bug like the one above

“Dirty COW” (CVE-2016-5195)
Lower level interface

Less code

Fewer vulnerabilities

Stronger isolation
• The level of interface has nothing to do with virtualization

• Virtualization makes things worse with regards to isolation
For historic reasons we tend to equate the interface mechanism with what it was created for.
We will show how to construct a system with a low level interface that uses process mechanisms.
Interface level ≠ mechanism

Virtualization mechanisms

- Traditional VMs
  - Low-level interfaces
  - Expose syscalls through hypercalls

Process mechanisms

- LibOS or microkernelization
  - Add library OS components

High-level interfaces

- Dune, gvisor-kvm
  - Traditional process
An application using files uses a file system provided by the host.
Adjust the interface level

- Can adjust the level of interface by moving the implementation to user level
How “much” isolation are we really gaining by going left?

What’s the isolation “cost” of virtualization?
- Measure the amount of code used as a proxy for measuring isolation.
Want to measure how much of the kernel is exposed
– Kernel function tracing (ftrace)

12 kernel functions

6 kernel functions
Comparing all options

Low-level interfaces

High-level interfaces

Traditional VMs

Traditional process

Low-level interfaces

Virtualization mechanisms

Process mechanisms

TCP/IP

FS

QEMU

KVM

TAP

kvm

Linux

TCP/IP

FS

...
Comparing all options

Virtualization mechanisms:
- KVM
- QEMU

Low-level interfaces:
- TAP
- TCP/IP

LibOS or microkernelification:
- Add library OS components

High-level interfaces:
- Traditional VMs
- Traditional process

Process mechanisms:
- LibOS or microkernelification

Traditional VMs:
- Add library OS components

Libraries:
- FS

Networking:
- Linux
  - KVM
  - QEMU
  - TAP
  - TCP/IP
  - FS
Dangers of virtualization mechanisms

- kvm is better than socket
- The white area in kvm represents the cost of managing virtualization hardware
What’s the catch?

• How do we design systems to achieve better isolation in the cloud? Considering these:
  – Generality
  – Performance
  – Maintenance

• What isolation metric to use?

Nabla containers: https://nabla-containers.github.io/
Dangers of virtualization mechanisms

- tap reduces the amount of networking needed as it's being pushed up
- tap is even lower than kvm
Discussion

• How do we design systems to achieve better isolation in the cloud?

• What isolation metric to use?
  – Can our systems be bug free?
  – Is less code better?
    • What about sanitizing code? More code is better in that case

Nabla containers: https://nabla-containers.github.io/
How can we ensure the protection of secrets in a multi-tenant cloud?
Metric for Isolation

- Thin interface is proxy for “less complexity to exploit”

![Diagram showing a thin interface with 32 system calls]

- Application

- Interface

- Linux

32 system calls
Metric for Isolation

- Thin interface is proxy for “less complexity to exploit”

- Want to measure how much of the kernel is exposed
  - Kernel function tracing (ftrace)

13 kernel functions
Interface level ≠ mechanism

- **Process (High)**
  - DUNE
  - Unix Process, Containers
  - Panoply, Scone

- **Devices (Low)**
  - Traditional VMs
  - FUSE, DPDK, Userfaultd, UML
  - Haven, Graphene

- **Virtualization**
- **sysenter (Process)**
- **SGX**
- **HW Interface Mechanism**
Dangers of virtualization mechanisms

- **qemu**: 358 reqs/s
- **kvm**: 4250 reqs/s
Linux Kernel's LOC increase

https://www.linuxcounter.net/statistics/kernel
Linux Kernel's LOC increase

Commit 4ceb5db97
“Dirty COW”
(CVE-2016-5195)

https://www.linuxcounter.net/statistics/kernel
Containers vs VMs for isolation?

**Container**
- Guest Application
- High level interface
- Linux
- TCP/IP
- FS

**VM**
- Guest Application + OS
- QEMU
- Low level interface
- KVM
- Linux

**QEMU**
- TCP/IP
- FS
Interface level ≠ mechanism

Virtualization mechanisms:
- Traditional VMs
- Dune

Low-level interfaces:
- Expose syscalls through hypercalls
- LibOS or microkernelification

High-level interfaces:
- Add library OS components
- Traditional process
• The level of interface has nothing to do with virtualization
Lower level interface

Less code

Fewer vulnerabilities

Stronger isolation

- Virtualization adds code and complexity
Comparing all options
Dangers of virtualization mechanisms

- qemu is the best you can do by taking stuff out of the kernel