The ideal versus the real:
a brief history of secure isolation in
virtual machines and containers

Allison Randal
University of Cambridge
Between the idea
And the reality
Between the motion
And the act
Falls the Shadow

Secure Isolation
Secure Isolation
Secure Isolation

Diagram showing multiple instances of OS within a Host OS container.
Secure Isolation
a securely isolated process, running on a kernel, containing an OS image
1950s

• Multiprogramming\textsuperscript{1, 2}
  – multitasking
  – multiprocessing: I/O processors and multiple CPUs
  – time-sharing
  – increase utilization
  – risk of disruption
  – complex to program

• kernel isolation\textsuperscript{3, 2}

1960s

- Capabilities
  - B5000\(^1\) descriptors
  - theoretical\(^2\) protected memory, ownership, subsets
  - MIT implementation on (modified) PDP-1\(^3\)
  - Chicago Magic Number Machine\(^4\)
  - CAL-TSS\(^4\)
  - Provably Secure Operating System\(^5\)\(^6\)

1960s

- **VMs**
  - M44/44X\(^1\) virtual memory
  - CP-40/CMS\(^2\), CP-67/CMS\(^3\) for IBM System/360 interrupt separation, paged guest memory, simulated devices, efficient utilization

- **OS**
  - Multics\(^4\)
  - Unix\(^5\)

---


1970s

- Capabilities
  - Plessey System 250
    telephone-switch controller
  - CAP hardware and OS
  - Intel iAPX 432
    poor performance
  - IBM System/38

1970s

- **VMs**
  - VM/370\(^1\) for IBM System/370 virtual memory hardware
  - “Since a privileged software nucleus has, in principle, no way of determining whether it is running on a virtual or a real machine, it has no way of spying on or altering any other virtual machine that may be coexisting with it in the same system. [...] In practice no virtual machine is completely equivalent to its real machine counterpart.”\(^2\)

- **OS**
  - BSD\(^3\)
  - chroot\(^4\) filesystem namespaces

---


1980s

- personal computing\(^1\) & monolithic servers
- hardware without virtualization support\(^2\)
- general purpose OS
- Intel x86\(^3\) “a crash program...to save Intel’s market share”\(^4\)
- RISC\(^5\) vs CISC

1990s

- Containers
  - POSIX.1e capabilities¹
  - Linux Kernel capabilities²
  - Plan 9 namespaces³ filesystem, process, network, memory
- VMs
  - Disco⁴ binary translation
  - VMware⁵
- Google scale?

2000s

- Web 2.0, smaller/lighter
- VMs
  - Denali\(^1\) \(^2\) paravirtualization
  - Xen\(^3\) multitenancy as a business
  - Amazon Web Services\(^4\) cloud, VM orchestration
  - x86 hardware virtualization\(^5\)
  - KVM\(^6\) (with QEMU)

---


2000s

- **Containers**
  - FreeBSD Jails\(^1\) & Solaris Zones\(^2\)
    - filesystem, process, network, resource limits
  - Linux VServer\(^3\) and OpenVZ\(^4\)
  - Linux namespaces\(^5\) filesystem, process, IPC, network
  - Linux cgroups\(^6\) resource/process control
  - LXC\(^7\) cgroups, namespaces, capabilities

- **Borg**\(^8\) workload orchestration

---

2010s

- Containers
  - Docker\textsuperscript{1} mass adoption
  - Linux user namespaces\textsuperscript{2}
  - Kubernetes\textsuperscript{3} workload orchestration

Myths: VM performance

- ukvm\textsuperscript{1} renamed to hvt
- LightVM\textsuperscript{2} faster Xen
- NEMU\textsuperscript{3} minimal QEMU

\textsuperscript{1}D. Williams and R. Koller. Unikernel Monitors: Extending Minimalism Outside of the Box. In 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16), 6, 2016.
\textsuperscript{3}https://github.com/intel/nemu
Myths: container security

• Kata Containers\(^1\) (was Intel Clear Containers\(^2\))
  – QEMU+KVM
• gVisor\(^3\)
  – kernel
  – devices
  – syscall filtering
• Depends on kernel security\(^4\)\(^5\) and “self-protection”\(^6\)

\(^1\)https://katacontainers.io/
\(^3\)https://github.com/google/gvisor
Myths: VM security

- Lines of code only vague potential for security\(^1\) \(^2\)
- Attack vectors\(^3\)
  - *source*: VM guest (Xen 71%, KVM 66%)
  - *target*: Ring -1, Dom0, host (Xen 80%, KVM 76%)
- Instruction emulation, arbitrary, unfiltered\(^4\)
- Depends on kernel security\(^5\) and “self-protection”\(^6\)

Myths: VM security

- Separate kernel mitigates some classes of vulnerabilities
- Speculative execution vulnerabilities
  - Spectre, NetSpectre\textsuperscript{1,2}
  - Meltdown\textsuperscript{3}
  - Foreshadow, L1TF\textsuperscript{4,5}

Lasciate ogne speranza, voi ch'intrate

–Dante Alighieri, “Inferno”

(Common translation: Abandon all hope, ye who enter here)
Positive directions

- Capabilities
  - Capsicum
  - CHERI
  - Fuchsia

- Hardware
  - RISC-V
  - Open Titan

- OS
  - OpenBSD pledge, unveil

---

6. pledge(2) manpage, https://man.openbsd.org/pledge.2
7. unveil(2) manpage, https://man.openbsd.org/unveil.2
Future directions

- Reexamine the full stack: hardware, kernel, OS, hypervisor/containers, guest, application workloads
- Synthesis: architecture/systems/security
Questions?