Advanced Unix System Administration

Lecture 24
April 26, 2007

Steven Luo
<sluo+decal@OCF.Berkeley.EDU>

Virtualization

- So what is virtualization anyway?
  - Strictly, should refer only to a certain type of what marketing calls “virtualization” nowadays
  - In the loose “marketing” definition, virtualization refers to any technology that allows the running of multiple full-fledged virtual servers on one machine
  - For convenience, we adopt this marketing definition as a banner for these related technologies

Virtualization

- Why virtualization?
  - Consolidation – run many services on one machine
    - Makes management easier in many cases
    - Increases hardware utilization, reducing costs
    - Reduces power consumption
  - Flexibility
    - Easy to create and destroy servers as needed
  - Security
    - Isolate services from each other

Virtualization

- Why not virtualization?
  - Creates single point of failure
    - This creates part of the interest in migrating virtual servers between machines
  - Features inside the virtual server
    - There are usually restrictions as to what you can do inside a virtual server
  - Security
    - A compromise of the host will lead to a compromise of all the virtual servers
    - If the isolation is buggy, a compromise of one virtual server might result in a full compromise

Virtualization

- Isolation
  - Virtual server processes run in a separate context in the same kernel
    - Context has varying degrees of isolation – usually includes at least filesystem, network, and resource isolation
    - What this context is called depends on the technology
  - Lightest-weight technology, but most limiting
    - Very little overhead – 1-3% in most implementations; supports potentially 100s of servers
    - Control of networking, hardware, etc. is limited

Virtualization

- Isolation con't
  - Implementations
    - FreeBSD jail: early implementation, very lightweight, isolation of network incomplete
    - Linux-Vserver: extension of various Linux features to create “contexts”; good isolation, but no in-context network control; various quirks
    - Solaris zones: “containers” offer good isolation, no in-container network control; some quirks
    - OpenVZ/Virtuozzo: “virtual environments” offer excellent isolation, in-container network control; most overhead

Virtualization

- Hypervisor-based virtualization
  - Pioneered by IBM in the 1970s
  - A thin “hypervisor” runs directly on the hardware and directs access
  - The hypervisor presents an interface looking like the bare hardware to kernels which it hosts
    - On architectures meeting the Popek-Goldberg requirements, this is easy
    - Otherwise, can use “paravirtualization” with minimal modifications to the guest kernel

Virtualization

- Hypervisors con't
  - Flexible and relatively lightweight, management more difficult
    - Can run different OS kernels, providing more choice, more control, and more isolation
    - More overhead than in-kernel isolation solutions – can be up to 10%
    - Each container has its own full OS, making central management more difficult
    - Resource sharing tends to be more inflexible than in-kernel isolation solutions
    - Technologies tend to be architecture-specific

Virtualization

- Hypervisors con't
  - Implementations
    - z/VM – the original IBM hypervisor, only runs on IBM mainframes (System 370 and up)
    - Xen – implements paravirtualization on x86, full virtualization on x86 hardware with extensions and IA-64; overhead of 3-8%
    - Newer versions of VMware, future versions of Microsoft Virtual Server on x86
    - Sun's Logical Domains on UltraSPARC T1

Virtualization

- Full virtualization
  - Provides emulation of a full hardware system
  - Code runs on host CPU where possible (compare emulation)
  - Heaviest-weight, slowest option
    - Few advantages over hypervisor virtualization on supported hardware
    - Most difficult management
    - Lots of overhead – 30-50% is typical

Virtualization

- Full virtualization
  - Implementations
    - Classic Vmware (x86)
    - MS Virtual Server (x86)
    - QEMU with the KQEMU module (x86)
    - VirtualBox (x86)