Advanced Systems Security: Virtual Machine Systems

Trent Jaeger
Systems and Internet Infrastructure Security (SIIS) Lab
Computer Science and Engineering Department
Pennsylvania State University

March 18, 2010
Two Directions

- OS Security from Reference Monitor perspective
  - Mediation
    - LSM
  - Tamperproof
    - Linux and TCB
  - Simple enough to verify
    - Correct code
    - Correct policy
Basis for OS Security

• Isolation
  ‣ A protection domain defines a boundary of isolation

• Based on
  ‣ Rings
  ‣ Address spaces
  ‣ Access control policy

• Do these work in modern OSes?
Virtual Machine Systems

- Protection domain is extended to operating systems on one physical platform
  - Invented for resource utilization
- But, also provide a potential security benefit due to default
  - ISOLATION
- How does VM isolation differ from OS isolation?
Virtual Machines

- Instead of using system software to enable sharing, use system software to enable isolation

Virtualization

- “a technique for hiding the physical characteristics of computing resources from the way in which others systems, applications, and end users interact with those resources”

Virtual Machines

- Single physical resource can appear as multiple logical resources
Virtual Machine Architectures

- **Full system simulation**
  - CPU can be simulated

- **Paravirtualization (Xen)**
  - VM has a special API
  - Requires OS changes

- **Native virtualization (VMWare)**
  - Simulate enough HW to run OS
  - OS is for same CPU

- **Application virtualization (JVM)**
  - Application API
Virtual Machine Types

• **Type I**
  - Lowest layer of software is VMM
  - E.g., Xen, VAX VMM, etc.

• **Type II**
  - Runs on a host operating system
  - E.g., VMWare, JVM, etc.

• Q: What are the trust model issues with Type II compared to Type I?
VM Systems and Ref Monitor

• How does a VM System improve ability to achieve reference monitor guarantees?

• Mediation
  ‣ Mediation between VM interactions

• Tamperproof
  ‣ Protection boundaries between OS

• Simple Enough to Verify
  ‣ Code that needs to be correct?
  ‣ Policy
VAX VMM

- A1-assured VMM system
- Carefully crafted VMM
- Mediation
  - VM interaction
- Tamperproof
  - Minimal TCB
- Simple enough to verify
  - Code assurance
  - Policy assurance: MLS policy, Biba policy, privileges
VAX VMM Design

The diagram illustrates the VAX VMM (Virtual Machine Monitor) design, showing the relationship between different levels of access and security:

- **Applications (Top Secret)**: Ultras OS
- **Applications (Secret)**: VMS OS
- **Applications (Unclassified)**: VMS OS

These applications are secured by the VMM Security Kernel, which communicates with several device drivers:

- Memory Device
- Disk Device
- Print Device
- Display Device

The diagram indicates a hierarchical security model with varying levels of access and protection.
VAX VMM Reference Monitor

• Key design tasks
  ‣ Virtualize processor
    • Make all sensitive instructions privileged
  ‣ More rings
    • Need a new ring for the VMM
  ‣ I/O emulation
  ‣ Self-virtualizable

• What components constitute the VAX VMM reference monitor?
VAX VMM Policy

- MLS
  - Control secrecy
- Biba
  - Control integrity
- Privileges
  - Exceptional accesses
  - Audited
  - There are more of these than meets the eye!
- How is the protection state modified?
VAX VMM Evaluation

- **Mediation**: ensure all security-sensitive operations are mediated?
  - Virtualizing instructions, I/O emulation
  - VM-level operations? Privileges

- **Mediation**: mediate all resources?
  - VMM level

- **Mediation**: verify complete mediation?
  - A1-assured at VMM level
VAX VMM Evaluation

- **Tamperproof**: protect VMM?
  - Similar to Multics (no gatekeepers, but some kind of filters); authentication in VMM; protection system ops in VMM; fixed system?

- **Tamperproof**: protect TCB?
  - All trusted code at ring 0; trusted path from VMs for admin;

- **Verification**: verify code?
  - A1-assured at VMM level

- **Verification**: verify policy?
  - MLS and Biba express goals and policy; Privileges are ad hoc
VAX VMM Tasks

• Despite A1 assurance still several challenges in VAX VMM system
  ‣ Device driver management; no network
  ‣ Amount of assembler code
  ‣ Covert channel countermeasures
  ‣ Implications of ‘privileges’

• Nonetheless, interesting mechanisms
  ‣ Trusted path administration
  ‣ Architecture of VMM
  ‣ Virtualization for security
Modern VM Systems

- The development of a virtual machine monitor for x86 systems unleashed VMs on the masses
  - Why did this take so long?
- VMware, Xen, KVM, NetTop, …
  - Everyone is a virtual machine monitor now
- How do we implement a reference validation mechanism for these systems?
  - What granularity of control?
Isolation and Network

- VMware and NetTop assume that the VMM (and privileged VM) will isolate guest VMs
- Then, the problem is to control inter-VM communication
  - Only other communication is via the network
- VMware uses firewall
- NetTop is built on VMware where only VMs of the same label may communicate
VMs as Processes

- Type II VM systems can treat VMs as processes
- **KVM** uses SELinux to control access of VMs as if they are a process
  - VMs are processes to the host OS
  - VMs can access host OS resources (files)
- Uses SELinux to control VM access
Control of VMM Resources

- There are many virtual machine monitor resources that may be used to communicate
  - Memory, devices, IPC, …

- sHype adds reference monitor for some objects (IPC) and the privileged VM uses for networking

- Xen Security Modules (XSM) adds reference validation on the Xen hypervisor’s distribution of these resources
  - Less trust in privileged VMs, so finer-grained policy results

- Minimizing TCB versus simplicity
Xen as a Reference Monitor?

- Reference Monitor
  - XSM in Xen
  - Scope includes dom0 Linux and user-level

- Mediation
  - XSM to control VMM operations
  - SELinux in dom0; use network to communicate

- Tamperproof
  - Xen has a much larger TCB, and more flexible

- Verification
  - Code – lots
  - Policy – SELinux style
Trusted VMs

- VMware and NetTop assume that the privileged VM (there is only one in these systems) prevents information flow (like a kernel)

- Thus, the only information flows between VMs are via networking
  - Privileged VM controls inter-VM communication via networking

- sHype controls IPC and networking at hypervisor level
  - Privileged VM uses hypervisor as policy store
Take Away

• VM Systems provide isolation
  ▸ Between OSes/apps that may be untrusted

• VM Systems enable a small TCB
  ▸ Type 1 VMMs
  ▸ A1-Assured, like VAX VMM

• VM Systems can mediate inter-VM actions
  ▸ Virtualized operations
  ▸ Inter-VM operations