Secure Out-of-band Remote Management of Virtual Machines with Transparent Passthrough

Shota Futagami, Tomoya Unoki, and Kenichi Kourai
Kyushu Institute of Technology, Japan
Remote Management of VMs

• **IaaS clouds**
  • Run virtualized systems and provide virtual machines (VMs)

• **Out-of-band remote management**
  • Enable users to manage VMs via virtual devices
    • E.g., virtual serial devices, keyboards, and video cards
  • Even on network configuration errors and at boot time
Information Leakage by Insiders

• Virtual devices may be managed by untrusted cloud operators
  • 28% of cybercrimes are caused by insiders [PwC'14]
  • 35% of admins have accessed sensitive information [CyberArk'09]
• Cloud operators can easily eavesdrop on and tamper with I/O of out-of-band remote management
Previous Approaches

• Several systems trusting the hypervisor have been proposed
  • Encrypt I/O between users and the hypervisor [Egawa+ CloudCom'12]
    • Virtual devices handle only encrypted data
  • Encrypt I/O in users' special VMs [Butt+ CCS'12]
    • These VMs are protected by the trusted hypervisor
Issues (1/2)

• The hypervisor can be compromised by insiders
  • Cloud operators can abuse rich interfaces to privileged components
• Not all the hypervisors are clearly separated from privileged components
  • KVM runs the hypervisor inside the host operating system
  • The entire host operating system has to be trusted
Issues (2/2)

- The entire virtualized system cannot be managed by cloud operators
  - Untrusted cloud operators cannot manage the hypervisor
  - Managing the hypervisor and the rest independently is unrealistic
- Cryptography management is problematic
  - It is not easy to securely manage cryptographic keys
VSBypass

• Achieve secure out-of-band remote management outside the virtualized system
  • Run **shadow devices** outside the virtualized system
  • Intercept I/O requests using **transparent passthrough**
    • Process them in shadow devices
  • Prevent information leakage and tampering
System Architecture

- Run the entire virtualized system in an outer VM (cloud VM) using nested virtualization
  - Shadow devices run outside the cloud VM
  - Remote users access shadow devices instead of virtual devices
  - The cloud VM and shadow devices run on top of the cloud hypervisor
Threat Model

- **Cloud operators may be untrusted**
  - Have full control over the virtualized system including the hypervisor
- **Cloud providers are trusted**
  - The TCB is the cloud hypervisor and shadow devices
  - A few trusted admins maintain the TCB
    - Admins form an administrative hierarchy
Advantages (1/2)

- **Difficult for cloud operators to attack the TCB**
  - The attack surface to the cloud hypervisor is much smaller
    - Only the hardware interface is provided

- **Support any virtualized systems**
  - The entire virtualized system is virtualized using the cloud VM
  - VSBypass does basically not depend on the internals
Advantages (2/2)

- Cloud operators can manage the entire virtualized system
  - VSBypass does not need to trust the guest hypervisor
  - It enables clearly separating the responsibility of system management
- Cryptography is not necessary
  - VSBypass does not rely on encryption to prevent information leakage
  - Users can use the existing remote management software
Shadow Devices

• **Use virtual devices of a proxy VM as shadow devices**
  • Assign minimum resources to a proxy VM
  • Pause a proxy VM after the boot
• **Bind a proxy VM to a user VM using EPT**
  • Transparently perform secure out-of-band remote management by specifying proxy VM's ID

![Diagram](image)
**Transparent Passthrough**

- **Intercept an I/O request of a user VM in the cloud hypervisor**
  - A VM exit occurs directly to the cloud hypervisor
  - By executing an instruction for port- and memory-mapped I/O
  - Not forward the request to the guest hypervisor as usual
  - The cloud hypervisor handles that request
I/O Processing in Shadow Devices

- **Redirect the intercepted I/O request to a shadow device**
  - Find a proxy VM from the trapped user VM using EPT
  - Send that request as if the request is issued by the proxy VM

- **Return the result to the user VM**
  - Intercept a completion event to the proxy VM
  - Redirect that event to the cloud VM
Sharing VRAM

- A user VM directly accesses VRAM in its main memory
  - The cloud hypervisor cannot intercept that access with low overhead
- A shadow video card shares the VRAM of a user VM
  - Process the VRAM as if it were the virtual device of the user VM
  - Track updates of the VRAM using the log-dirty mechanism
Virtual Interrupts in Shadow Devices

• Redirect virtual interrupts caused in shadow devices to a user VM
  • Deliver interrupts via the interrupt server in the virtualized system
  • Virtual interrupts include no sensitive information
• Share a ring buffer using an ultracall [Miyama+ ESORICS'17]
  • Receive interrupts by polling
Experiments

• We conducted experiments to show the effectiveness of VSBypass
  • Eavesdrop on I/O of out-of-band remote management
  • Measure the performance of a virtual serial console and GUI remote access
• Comparison with the traditional system without nested virtualization
Eavesdropping

• *We eavesdropped on I/O data in virtual devices*
  • Record input/output characters in the virtual serial device
  • Record keys in the virtual keyboard and screenshots in the virtual video card

• *We could not obtain any data in VSBypass*
  • We could capture all data in the traditional system
Performance of a Virtual Serial Console

• The response time between a character input and its echo-back in an SSH client
  • VSBypass was 1.3 ms longer due to nested virtualization

• The throughput of outputting a text file to an SSH client
  • VSBypass was almost the same
Performance of GUI Remote Access

- The response time between a key press and a screen update in a VNC client
  - VSBypass was 13 ms longer due to the screen update
- The update time of user VM's screen in a VNC client
  - VSBypass was 23 ms longer
Related Work

• **Device passthrough**
  • Enable direct access to physical devices from VMs
  • Transparent passthrough in VSBypass is applied to virtual devices

• **CloudVisor [Zhang+ SOSP'11]**
  • Encrypt the memory/disks of user VMs in the cloud hypervisor
  • Protect no virtual devices for out-of-band remote management

• **V-Met [Miyama+ ESORICS'17]**
  • Offload IDSes outside the virtualized system with nested virtualization
  • Provide deep VM introspection to obtain the internal state of user VMs
Conclusion

• **We proposed secure out-of-band remote management outside the virtualized system**
  • Run the virtualized system in a VM using **nested virtualization**
  • Intercept I/O requests outside it using **transparent passthrough**
  • Process them in **shadow devices**

• **Future work**
  • Create/destroy proxy VMs with the boot/shutdown of user VMs
  • Support the migration of user VMs with the state of shadow devices