Virtually Secure

Oded Horovitz
VMware R&D

vmware®

Talk Overview

Setup
- Virtualization 101
- Talk Focus

VM Introspection
- Capabilities
- Sample Use Cases (and demos)

Magics
- Retrospective Security

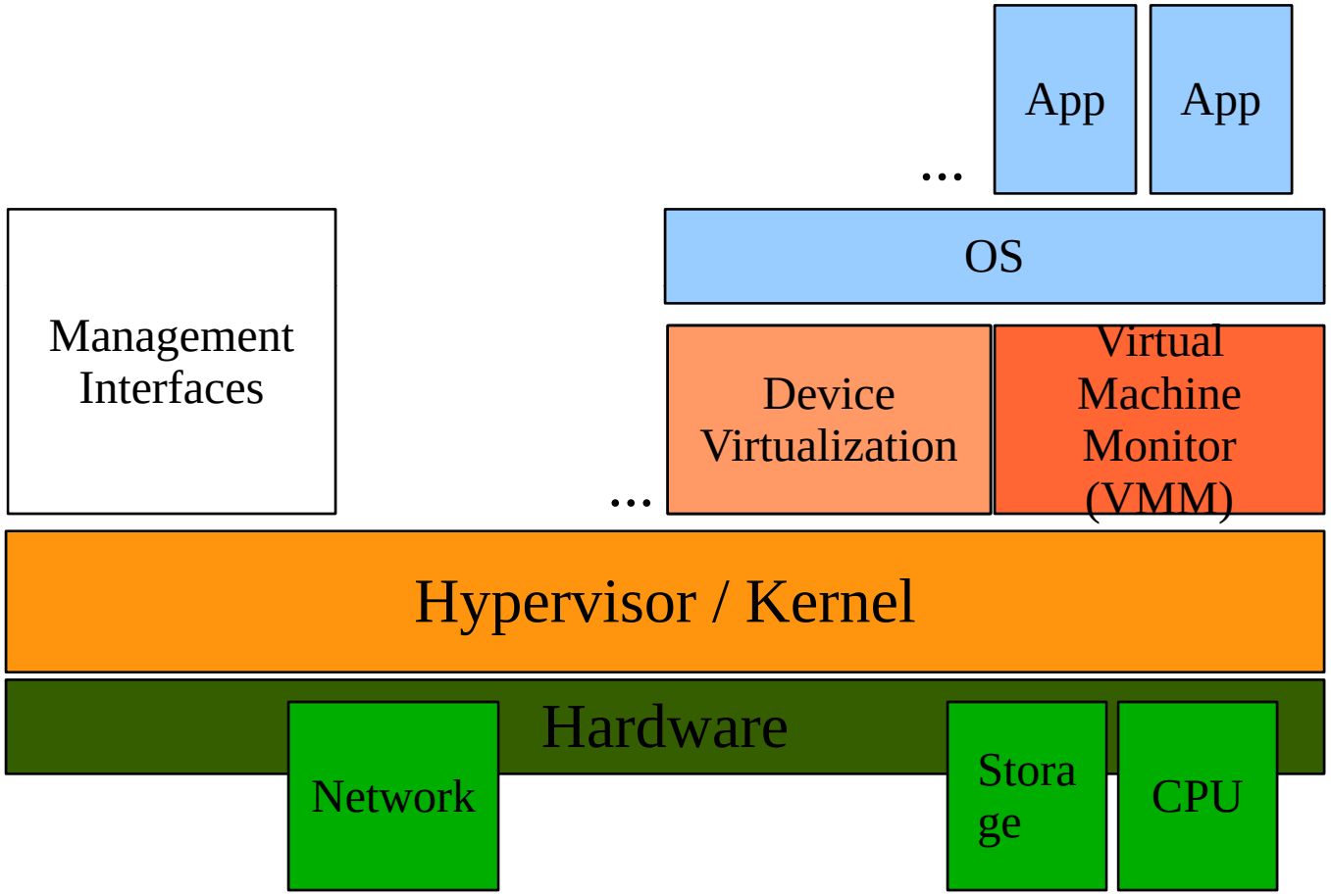
Misc & QA

Setup

Virtualization 101

Key Terms

- VMM
- Hypervisor
- Hosted
- Bare Metal



Setup

Talk Focus

Virtualization Based Capabilities

- Better than physical
- Hypervisor as a Base of Trust
- Security as an infrastructure service

Also Important But not Today

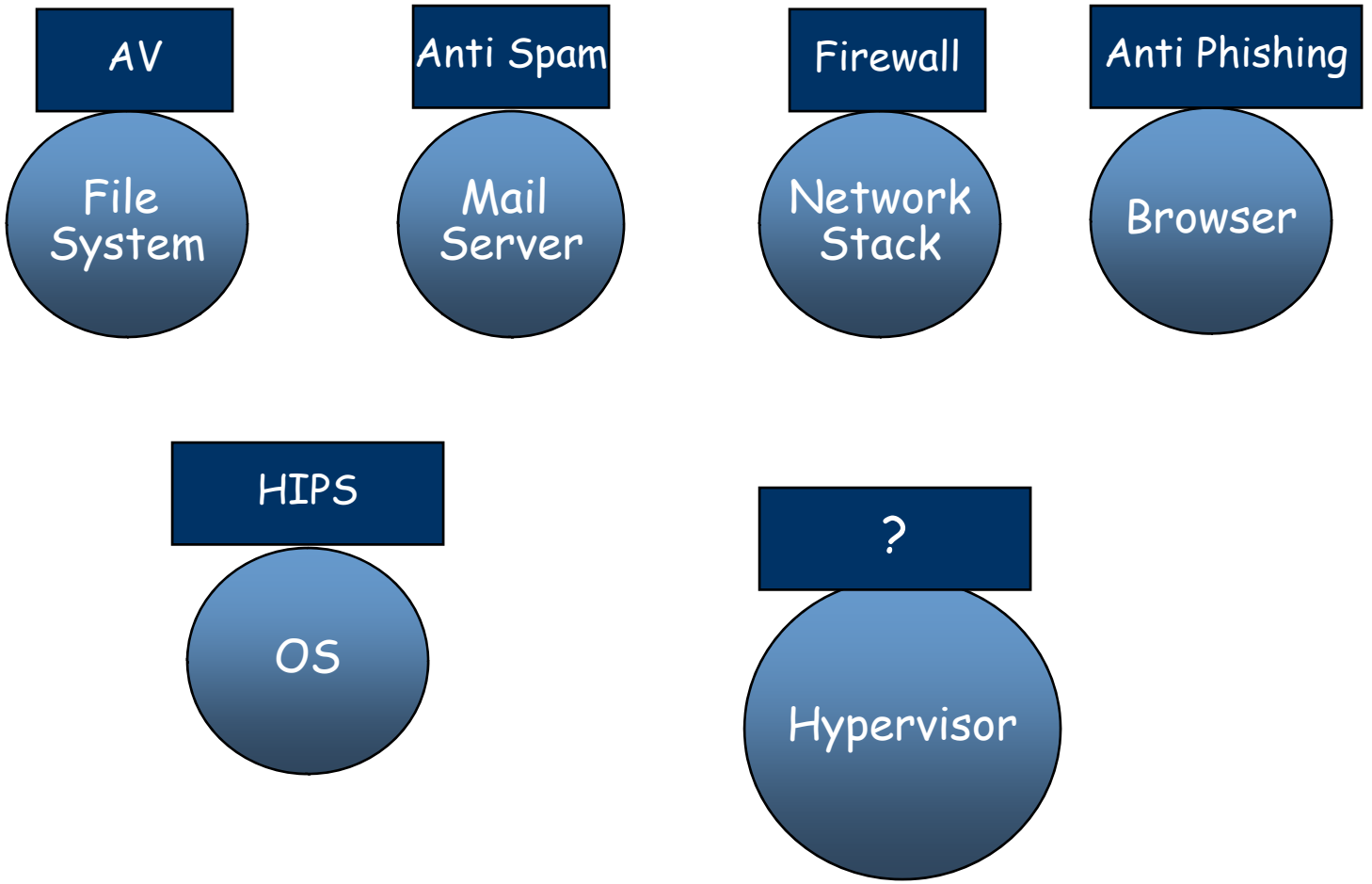
Secure Virtualization Infrastructure

- Secure & Manageable Platform

Physical Equivalent Security

- Support existing tools and agents
- Prevent security coverage loss when P2V

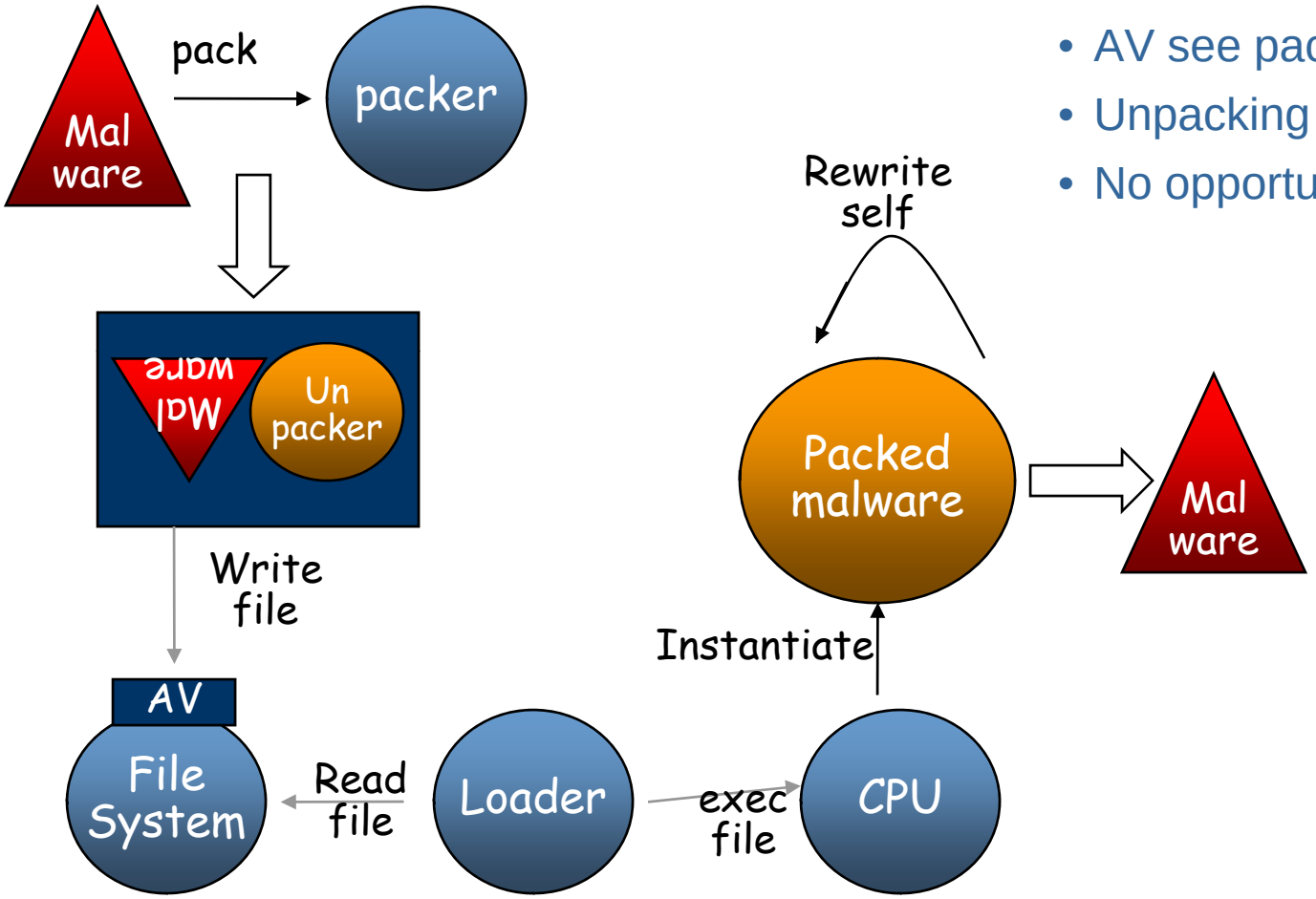
Introspection — Security Agent – common agents



Physical Security - Shortcomings

Code Packing

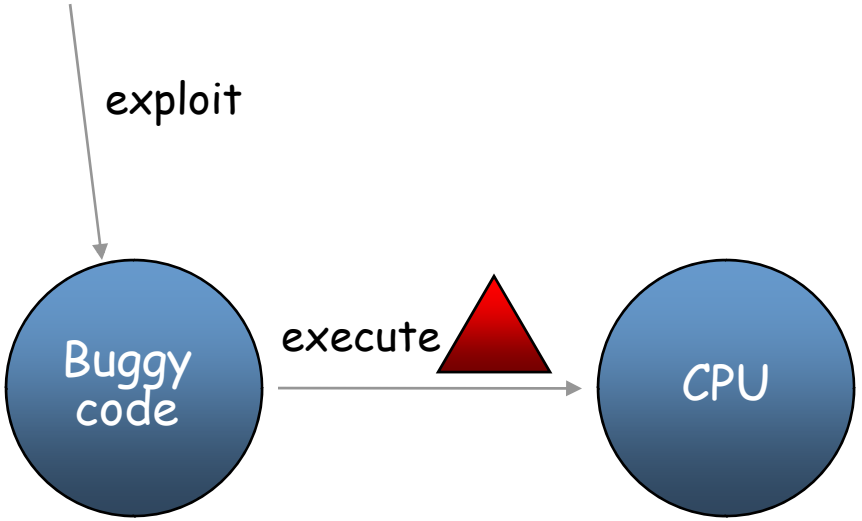
- AV see packed file
- Unpacking method is unknown
- No opportunity for detection



Physical Security - shortcoming

Vulnerabilities

- Buggy service is exploited
- New code is injected
- File system never sees the new code (unless it is paged out..)



Existing solutions

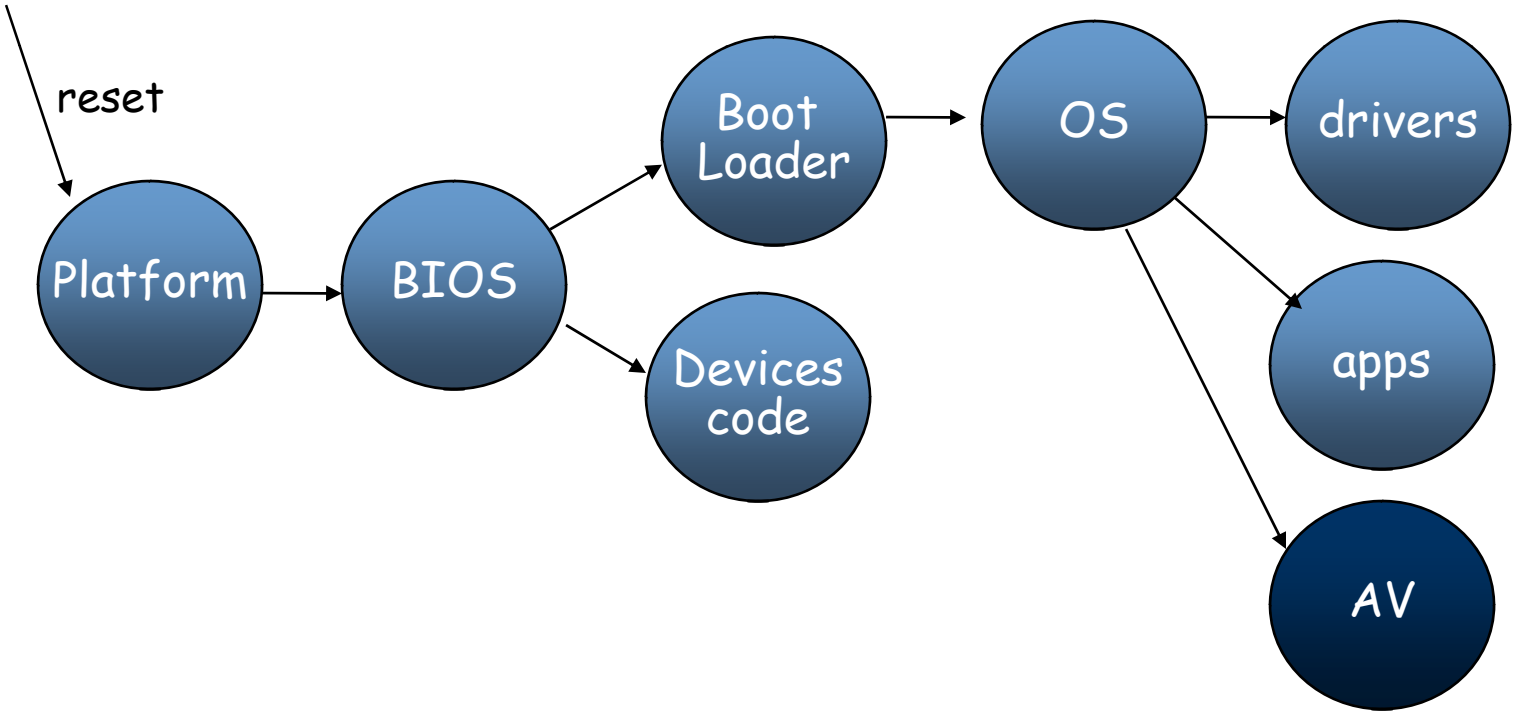
- Program shepherding
- ASLR
- NX

No good coverage for kernels

Introspection — Physical Security - shortcoming

OS coverage

- Agent is depended on its host (instantiated by host)
- A window of opportunity exist to subvert system
- Solution - Boot into alternate OS and scan?



VM Introspection

CPU events

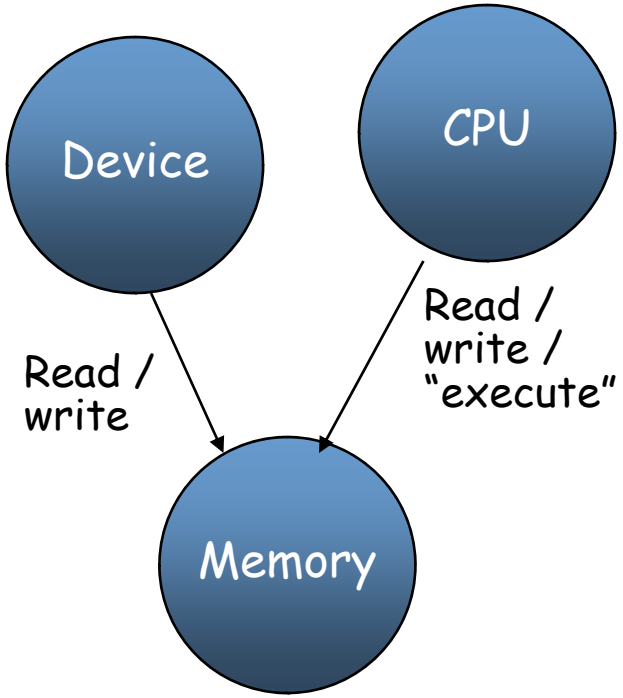
- Privileged instruction
- Exceptions
- Interrupts
- I/O
- Arbitrary Instruction op-code
- Instruction breakpoint
- Control flow

VM Introspection

Memory event

- Granular CPU read / write
- Granular device read / write
- Linear addressing
- Page granularity
- Physical addressing

HV unfriendly
High overhead



VM Introspection

Security API's

- Designed for security productization
- Agent runs within a VM
- Capabilities
  - Memory access events
  - Selected CPU events
  - VM lifecycle events
  - Access to VM memory & CPU state
  - Page Table walker

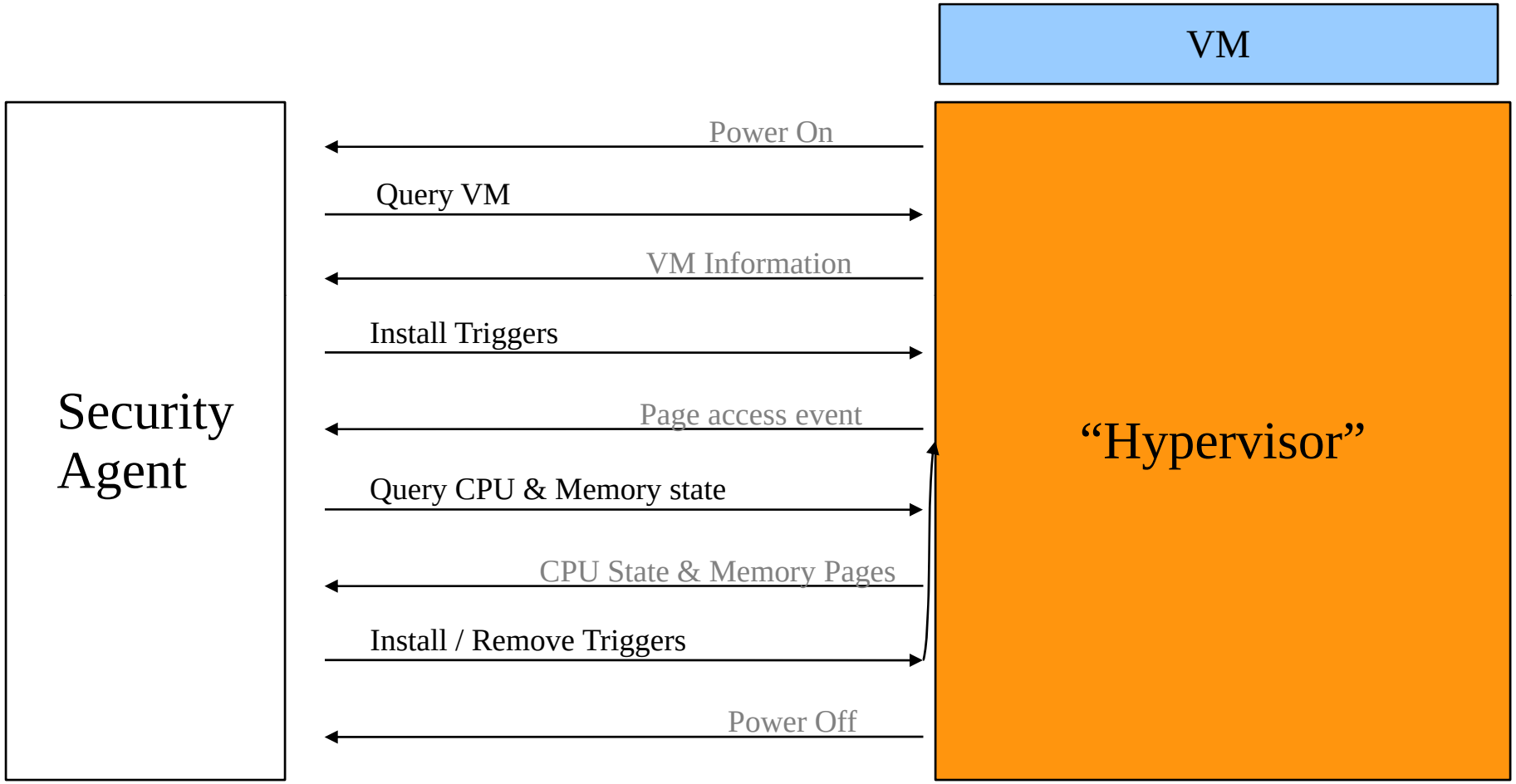
Security APIs (VMsafe)

Goals

- Better than physical
  - Exploit hypervisor interposition to place new security agent
  - Provide security coverage for the VM kernel (and applications)
- Hypervisor as a Base of Trust
  - Divide responsibilities between the hypervisor and in-VM agent
  - The hypervisor covers the VM kernel, the rest is done from within the VM
  - Insure in-VM security agent execution and correctness
- Security as an infrastructure service
  - “Agent less” security services for VMs
  - Flexible OS independent solutions

Introspection

Verify-Before-Execute Flow



Sample Introspection Agents

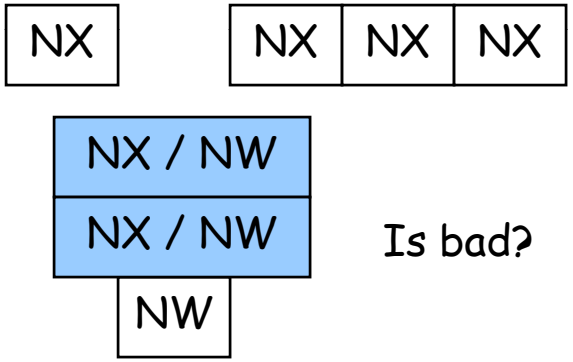
Verify-Before-Execute

Utilize memory introspection to validate all executing pages

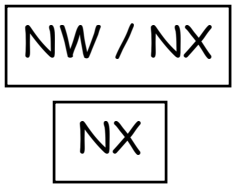
Flow

1. Trace all pages for execution access

1. On execution detection
   - Trace for page modification
   - Verify if page contain malware
   - Remove execution trace



1. On modification detection
   - Trace for execution
   - Remove modification trace



Security APIs – Use cases

VM Kernel coverage

- Detect infection in early boot process
  - Device opt ROM attacks
  - Boot loader
  - Boot records
  - OS image
- Detect code injection due to kernel vulnerabilities
- Detect self modifying code in kernel
- Lock kernel after initialization

Introspection Case Study - Microsoft Patch Guard

Goal

- Prevent patching of (x64 based) kernels
- Force ISV to behave nicely
- Prevent Root-kits ??

Implementation

- Obfuscated Invocation
- Obfuscated Persistence
- Evolving (Thanks to the awesome work from uninformed.org)

What's The Problem?

- Circumventable
- Complicated
- Only for x64 based Windows Systems

Kernel Security Demo

"MyPatchGuard"

- Secure & Isolated Agent
- Inline enforcement using memory write triggers.
- Protect Windows XP guest syscalls table
- Simple.

Security APIs – Use cases cont'

Watch dog services

- Liveness check for in-VM security agent
  - Detect agent presence
  - Verify agent periodic execution
  - Protect agent code and static data

TPM vs. Introspection

TPM

- Root of trust rely on hardware
- Passive device
- Platform and software stack decide what to measure
- Need software update to change measurement coverage
- Can not detect compromise in software stack since verification

VM Introspection

- Root of trust rely on hypervisor
- Introspection agent have the initiative
- Security vendor / policy dictate what to measure
- Coverage is content, and can change independently of VM
- Designed to continuously scan VMs and to detect compromise

VMsafe – Network Introspection

Capabilities

- Place an inline network agent on any VM virtual nic
- Allow reading, injecting, modifying, and dropping packets.

Benefits

- Efficiently monitor inter-VM network communication
- Integrated support for live migration.

Virtualization only applications

- Correlate VM internals with network policy. (using CPU/ Memory inspections one can learn OS version, patch level, configuration etc)
- Build a trusted distributed firewall.

Talk Overview

Setup

- Virtualization 101
- Talk Focus

VM Introspection

- Capabilities
- Sample Use Cases (and demos)

Magics

- Retrospective Security

Misc & QA

Retrospective Security

Motivation

- Detect whether you have been attacked in the past
- Detect if you might be still compromised by a past attack

Method

- VMware Record & Replay allow for a deterministic replay of VM using recorded logs
- Potentially the recordings have captured an attack
- The security API's are detached from the recorded VM (unlike in-VM agent) and can attach to a replay session

Retrospective Security

Demo

What is it good for?

- Run more aggressive policies that will not be acceptable in production environments
- Discover 0days used to exploit your system
- Learn how the malware / attacker have navigated your system
- Use data tainting technique to detect any side effects that still exist on your system
- Possibly clean the finding from last step on your production VM.
- Learn about the scope of the damage done to your system, i.e. what is the extent of data leakage

Security vs. Hardware Virtualization

1st Generation – SVM, VT-X

- VMM no longer need to run the VM kernel under binary translation
- Security Trade off – Code Breakpoint, Guest code patching (while translating), Control flow visibility

2nd Generation – NPT, EPT

- VMM no longer need to have software based MMU
- Security Trade off – Tracking LA->PA mapping is becoming expensive, resulting with inability to operate on linear addresses.

3rd Generation – IO MMU, VT-D

- VMM can assign physical devices to VMs without worry of VM escape or hypervisor corruption
- Security Trade off – Interposition on the pass-thru device is eliminated

Conclusion

Questions?

Contact

odedh@vmware.com