SAND2010-5159P

Virtual Machine Monitors as Secure Service Platforms

Jason Trent

August 4, 2010

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

LOCKHEED MARTIN

Sandia National Laboratories

Welcome

- Categories of hypervisors
- Type I versus Type II hypervisors for VM IaaS
- Challenges to VM IaaS
- Potential VM services
- Virtualization as a secure platform
- Requirements for building a secure virtual platform
- Interesting academic architectures

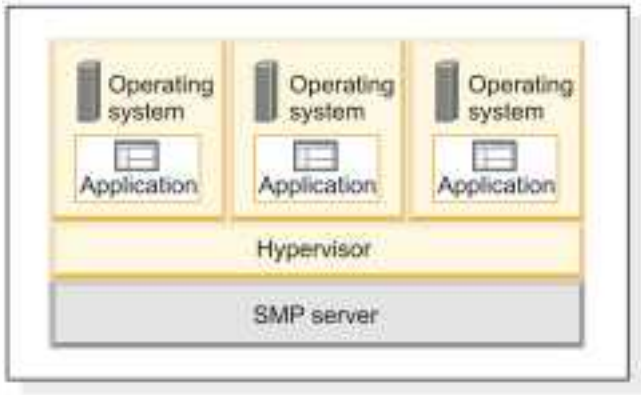
VM Infrastructure as a Service

"Adding services via a virtual machine is analogous to adding network services via a firewall. Both virtual machines and firewalls intercept actions at a universal, low-level interface, and both must overcome performance and semantic-gap problems. Just as network firewalls have proven useful for adding network services, we believe virtual machines will prove useful for adding services for the entire computer."

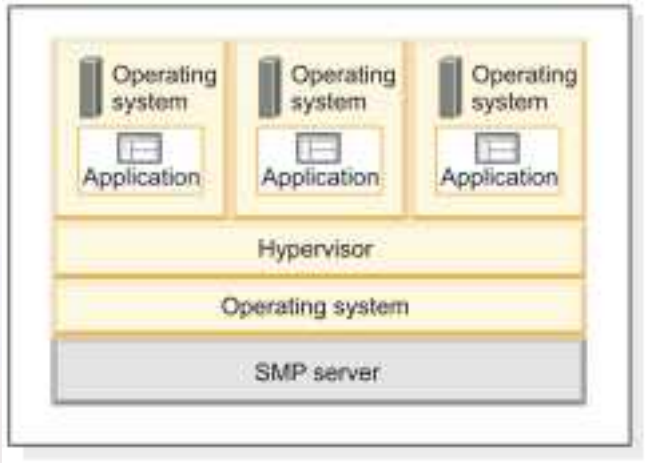
- Chen, P.M., Noble, B.D., "When Virtual Is Better Than Real," *In Proceedings of the 2001 Workshop on Hot Topics in Operating Systems (HotOS)*, pages 133– 138, May 2001.

Two Categories of Hypervisors

- **Type I Hypervisors**
  - Bare Metal
  - Examples:
    - VMware ESXi
    - Hyper-V
    - “Thin” hypervisors



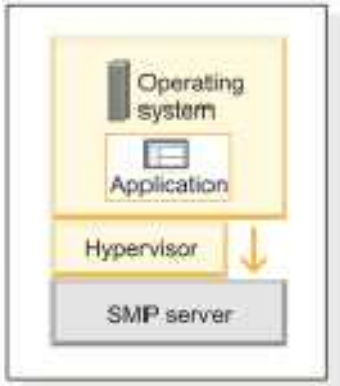
- **Type II Hypervisors**
  - Hosted
  - Examples:
    - VMware Workstation
    - VirtualPC
    - Parallels



- **IBM, “IBM Systems Virtualization,” 2005.**

“Thin” Hypervisors

- Subclass of Type I hypervisor
- Single guest
- Guest allowed to interact with some hardware directly
- Hypervisor virtualizes as little as possible to maintain itself, its security model, its features, and its running guest
- Examples: Blue Pill, Vitriol, BitVisor, MAVMM



When One Is Better Than Two

- Most security tools and research today are built with hosted hypervisors
- These require that your VM is an image and runs in a hosted OS. This is good for large servers and honey nets, etc…
- However, for end-point workstations this is less desirable
- Building on top of bare metal hypervisors allows an pre-existing OS to run inside a VM-based security tool
- This likely means only one OS running at a time
- Most principles still apply when using a type two hypervisor

Challenges to VM IaaS

- **Challenges**
  - ✓ Performance [1]
  - ❑ Semantic gap [1]
  - ❑ Introspection subversion [2]
  - ❑ Hypervisor trust

1. Chen, P.M., Noble, B.D., "When Virtual Is Better Than Real," *In Proceedings of the 2001 Workshop on Hot Topics in Operating Systems (HotOS)*, pages 133– 138, May 2001.
2. Bahram, S., Jiang, X., Wang, Z., Grace, M., Li, J., Xu, D., "DKSM: Subverting Virtual Machine Introspection for Fun and Profit," NC State University, TR-2010-4, February 2010.

Potential VM Services

- **Secure logging/reporting**
- **Intrusion prevention and detection**
- **Monitoring guest components**
  - SSDT/IDT hook detection
  - Kernel integrity enforcement
- **Debug registers –**
  - Strict monitoring, control, and usage of
- **…**

Virtualization as a Platform

- **Static "thin" security hypervisors**
  - BitVisor
  - rkAnalyzer (based on BitVisor)
  - MAVMM
- **Extensible "thin" security hypervisors**
  - ?
- **…DANGER…**
  - Run-time loadable "ring -1" code
  - Easier exploitation by malware
  - Larger attack surface
  - Sounds a lot like attacking a kernel

VMM Assurance

- ***"… a VMM is a simple-enough mechanism that we can reasonably hope to implement it correctly."*** [1]
- ***"We also assume that SecVisor does not have any vulnerabilities."*** [2]
- **Et cetera…**

1. Garfinkel, T., Rosenblum, M., "A Virtual Machine Introspection Based Architecture for Intrusion Detection," In *Proceedings of the Networked and Distributed Systems Security Symposium,* February 2003.
2. Seshadri, A., Luk, M., Qu, N., Perrig, A., "SecVisor: A Tiny Hypervisor to Guarantee Lifetime Kernel Code Integrity for Commodity Oses," In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP 2007),* October 2007.

Virtualization as a Secure Platform

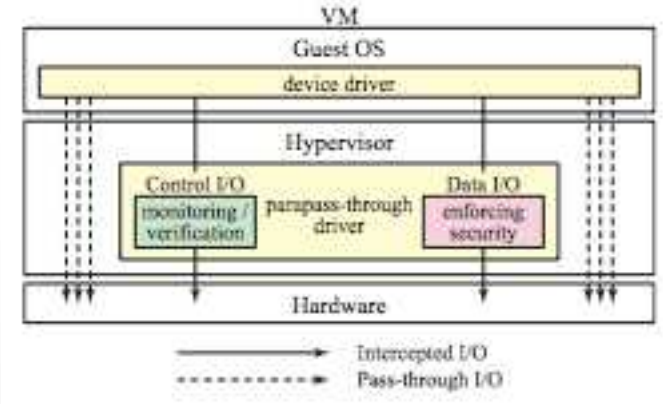
- **Can't assume the hypervisor is secure**
  - Do everything we can to secure it
- **Detect something wrong?**
  - Bail out (ala PatchGuard)
- **Attestation for known, good initial state**
- **How do we trust loadable modules**
  - Only load trusted, signed modules (e.g. Driver Signing)
- **Sounds a lot like securing a kernel**

Designing a Secure Virtual Platform

- **Employ hardware protection**
  - Paging –
    - Extended Page Tables (EPT) on Intel
    - Rapid Virtualization Index (RVI) (aka Nested Page Tables) on AMD
    - Shadow paging
  - DMA –
    - Virtualization Technology for Direct I/O (Vt-d) on Intel
    - AMD I/O Virtualization (AMD-Vi aka IOMMU) on AMD
  - TPM –
    - “Thin hypervisor” starts before OS
    - Load-time hypervisor integrity
    - Measure similar to Secure Boot

Designing a Secure Virtual Platform

- **Software protection**
  - Best practices, code auditing, etc…
  - Control particular driver interactions
  - Hypervisor run-time integrity
  - HyperSafe approach



[1]

1. **Takahiro Shinagawa et al., "BitVisor: A Thin Hypervisor for Enforcing I/O Device Security," *Proceedings of the 2009 ACM SIGPLAN/SIGOPS Virtual Execution Environments (VEE '09)*, March 2009.**

HyperSafe

- **Non-Bypassable Memory Lockdown**
  - Paging-based protections
- **Ensure lifetime W⊕X for all hypervisor code pages**
  - i.e. After initial setup no page that has ever been writable will be allowed to become executable
- **Ensure no pages are double-mapped to violate this property**
- **Write-protect page tables**
- **Hypervisor uses page-faults in hypervisor to enforce**

HyperSafe continued…

- **Restricted Pointer Indexing**
  - Used for control-flow integrity (CFI)
  - Modification to the compilation
- **Pre-compute indirect control-flow and use function entrance and exit wrappers to enforce this control flow**
- **Mark data pages which contain the control flow tables as read-only**

- **Wang, Z., Jiang, X., "HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity," In *Proceedings of the Thirty First IEEE Symposium on Security & Privacy*, Oakland 2010.**

Building an Extensible Hypervisor

- **Requirements:**
  - Kernel best practices
    - NX, measured launch, integrity checking, etc…
  - Utilize hardware
    - EPT/RVI
    - Vt-d/AMD-Vi
    - TPM
  - Driver control or device virtualization
    - Required for interposition of devices
    - Must be able to know the state of hardware
    - Doesn't necessarily need full driver

Building an Extensible Hypervisor

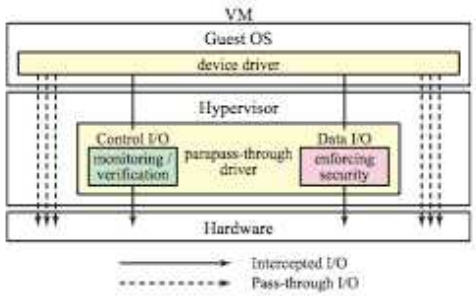
- **Requirements:**
  - Module loader
    - Signing enforced by loader
    - Modules not allowed to allocate executable memory
  - Symbol resolution
    - Imported internal symbols resolved for module
    - Exported symbols from module maintained
  - Inspection/interposition API
    - Inspect guest memory
    - Assist bridging semantic gap
    - Interpose on data transfer (i.e. to/from hard disk, network, etc…)

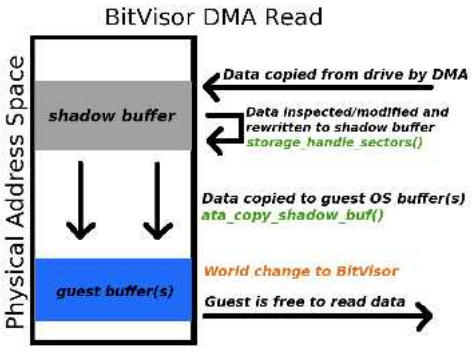
FLOW: A VM Service Case Study

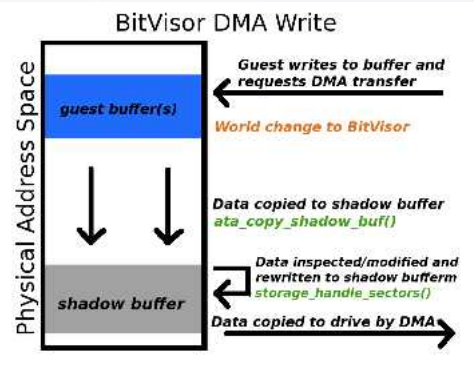
- FLOW is a disk transfer inspector built on top of the BitVisor hypervisor
- Interposes on all disk reads and writes
- May sanitize DMA contents before data is written to disk or used by the OS
- Does NOT currently reconstruct files or higher-level semantics (i.e. it operates only on DMA transaction buffers)

FLOW: Implementation

- BitVisor uses the built-in ability to capture all disk reads and writes to implement encryption
- Single choke point, storage_handle_sectors(), for hard disk data interposition
- Modified to implement a simple signature scanner/sanitizer
  - This is where an interface could be added to allow additional computation without modifying original code



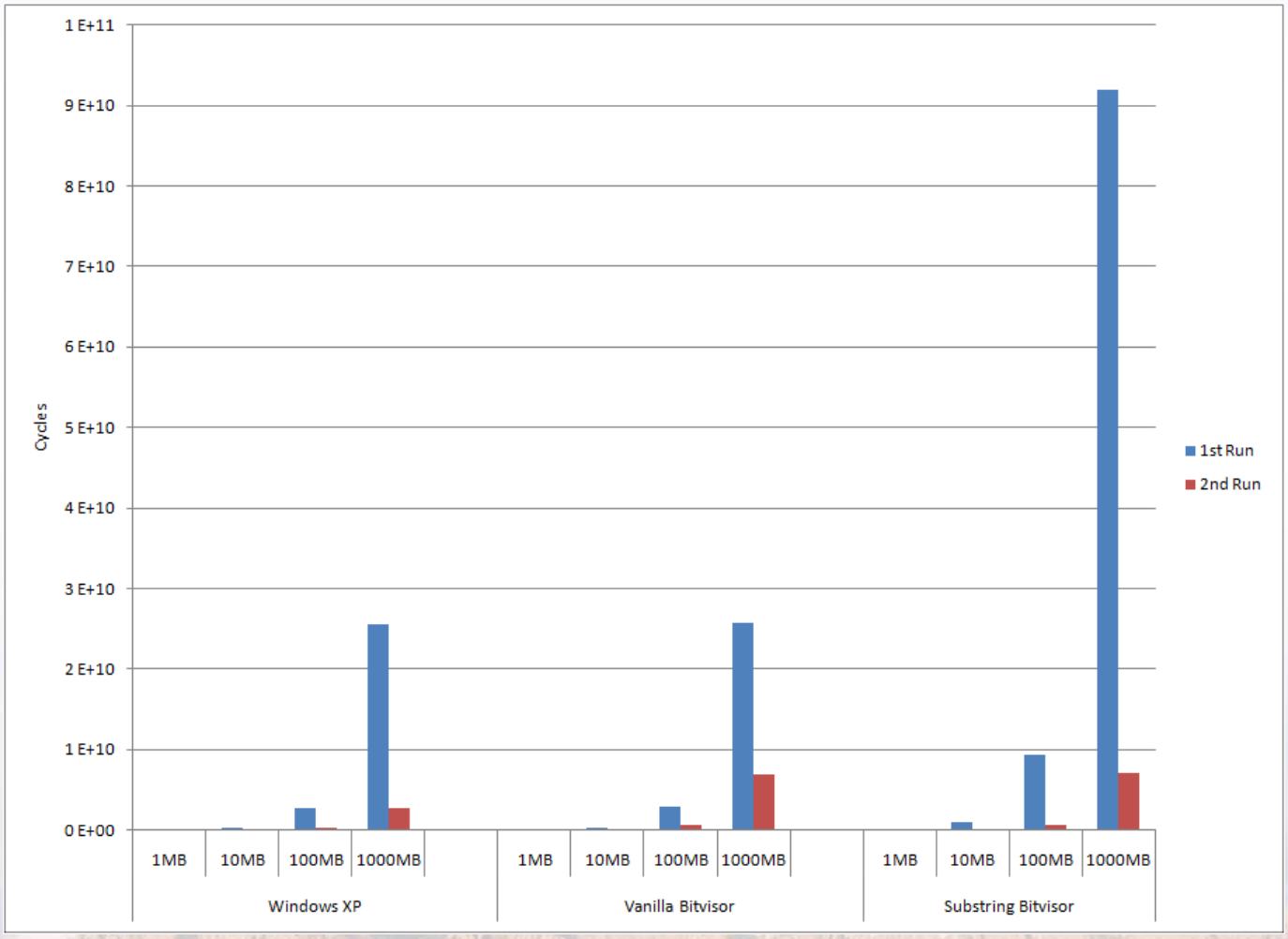




FLOW: Timing Analysis

- **Created files of random data**
- **Timed transfers of size: 1 MB, 10MB, 100MB, and 1GB**
- **Read file twice**
  1. Initial read (no cache)
  2. Second read (with cache from first read)
- **Tested three environments:**
  - Plain Windows XP
  - Windows XP inside vanilla BitVisor 1.1
  - Windows XP inside BitVisor with substring signatures

FLOW: Performance



Sandia National Laboratories

FLOW: Outcome

- Inspecting data does add overhead, in addition to the hypervisor
- However, this overhead is insignificant relative to the cost of performing the disk I/O
- In the past, similar results have been shown with network traffic
  - I/O inspection adds minor overhead, but provides very powerful positioning to respond to malicious/undesirable data
  - The hypervisor design should allow the analysis performed on data to be extensible

Parting thoughts

- Can't implement service on top of hypervisors that are assumed to be secure
- Must work toward protected environment
- There is still a lot of research to be done in hypervisor security
- Hypervisor enables a more closed eco-system that may benefit from restrictions that are less practical in the kernel
- Developing extensible hypervisors provides a quick means to add services

Interesting VM Architectures

- Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D., "Terra: A Virtual Machine-Based Platform for Trusted Computing," in *Proceedings of the 9th ACM Symposium on Operating Systems Principles*, pp. 193–206, 2003.

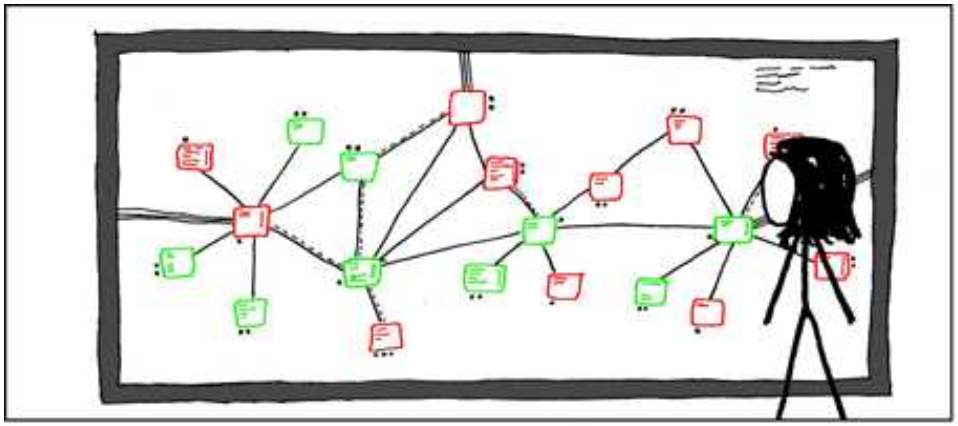
- Sailer, R., Valdez, E., Jaegar, T., Perez, R., Doorn, L., Griffin, J., Berger, S., "sHype: Secure Hypervisor Approach to Trusted Virtualized Systems," *IBM Research Report, RC23511, February 2005.*

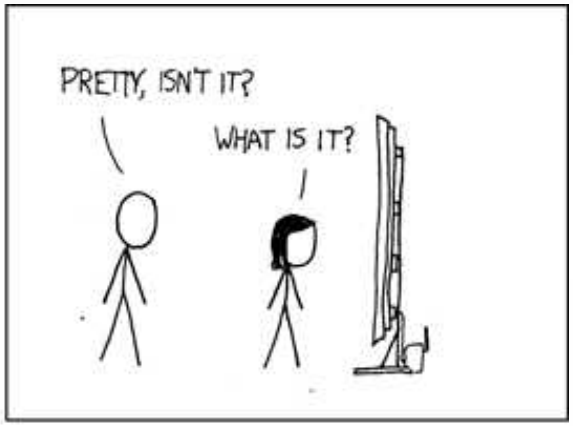
- Payne, B., Carbone, M., Sharif, M., Lee, W., "Lares: An Architecture for Secure Active Monitoring Using Virtualization," *In Proceedings of the IEEE Symposium on Security and Privacy*, 2008.

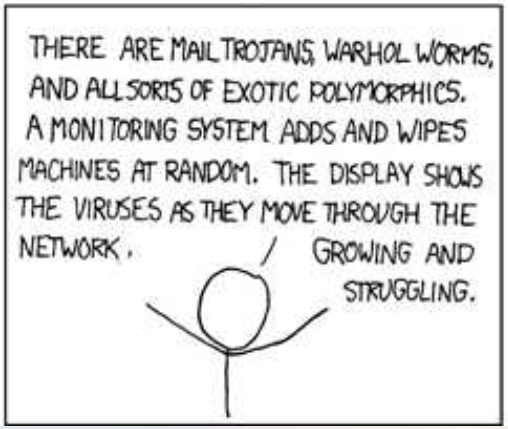
- Rutkowska, J., Wojtczuk, R., "Qubes OS Architecture," Invisible Things Lab, January 2010.

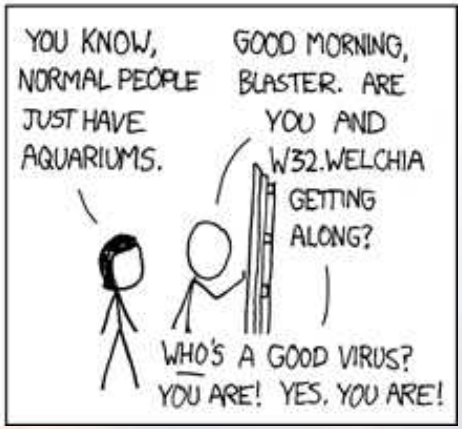
- Douglas, H., "Thin Hypervisor-Based Security Architectures for Embedded Platforms," Royal Institute of Technology, Sweden, February, 2010.

Questions









- Munroe, R., "Network," http://xkcd.com/350/, November 2007.