Using Virtual Machine Introspection for Deep Cybersecurity Education

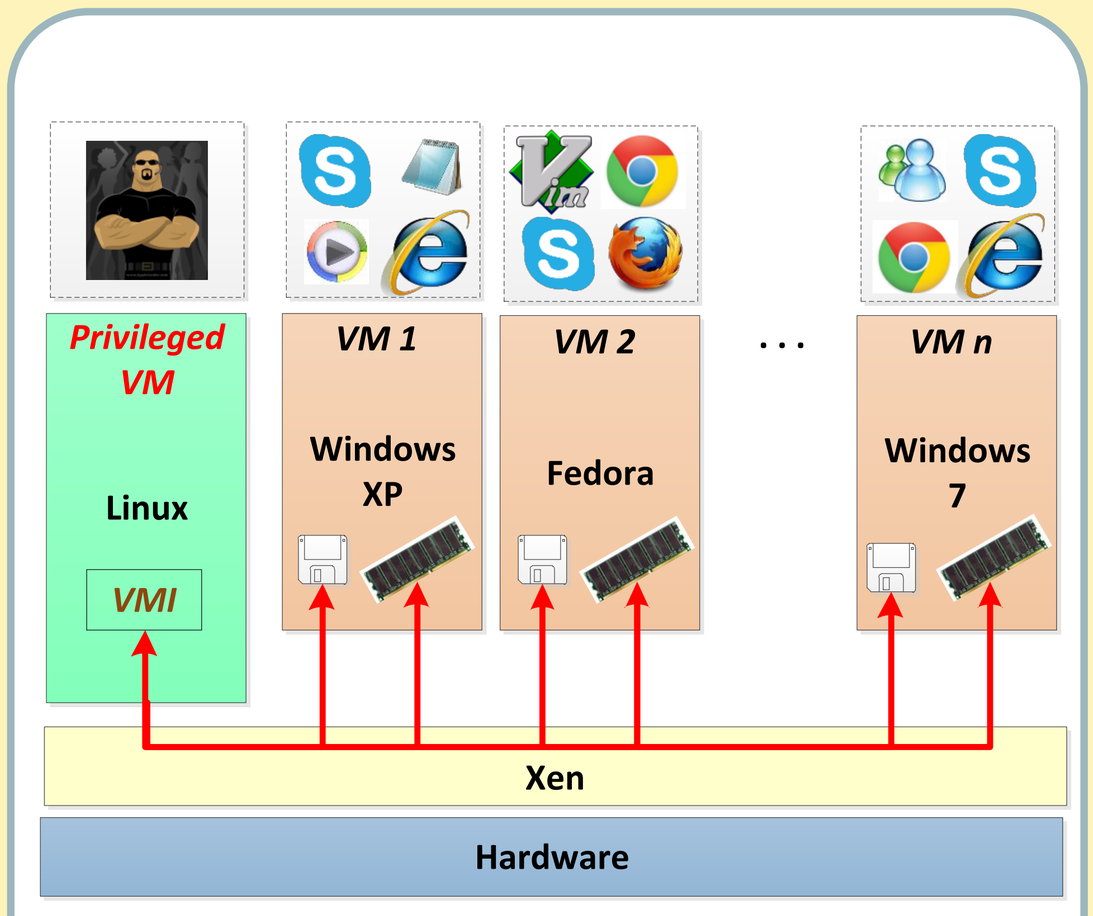
Zhiqiang Lin
University of Texas at Dallas

Irfan Ahmed
University of New Orleans

Playground for physical memory manipulation of a virtual machine

The objective of this project is to develop a virtual machine introspection (VMI) toolkit in support of hands-on computer security education.

- A code (for offense/defense) executed in a VM makes changes in the current state of memory
- The changes are essential for students to understand the underlying concept
- Unfortunately, these changes are mostly transparent to students during hands-on



Virtual Machine Introspection (VMI)

- Capability to examine the system resources (memory/disk) of a guest VM
- Directly from outside of the guest VM
- VMI tools run on a privileged VM

Approach – VMI Toolkit

- Utilize VMI to develop a toolkit for studying both user and kernel level attacks/defenses
- The toolkit provides conceptually different approach for many hands-on exercises
- provides a playground for memory manipulation that let students make direct changes in memory contents and then observe their effect
- particularly useful to understand lower level internals of software, including kernel and user applications, and offensive and defensive approaches in cyber security

Example - DKOM

- FU rootkit hides a process from user applications
- The process list is maintained by kernel as a linked list
- The toolkit allows students to perform main functionality of the rootkit
- Modify pointers of the list directly in the memory to delink the node of the process to be hidden

Example – Code-reuse Attacks

- Return oriented programming (ROP)
- The toolkit automatically identifies gadgets in a live physical memory of a VM
- Setup the return addresses in the stack of a program to demonstrate the ROP attack
- Allows students to observe the impact of the attack

Example – Buffer Overflow

- a vulnerability to overwrite more data into a buffer than the buffer can hold.
- The toolkit provides a detailed view of stack content
- Overwrites the buffer and its adjacent memory locations including return address

Example – PatchGuard (for Kernel code)

- List kernel modules, and extract their code from memory
- periodically check the integrity of kernel modules
- Allows students to modify kernel code to raise alert

Interested in meeting the PIs? Attach post-it note below!

National Science Foundation
WHERE DISCOVERIES BEGIN

The 3rd NSF Secure and Trustworthy Cyberspace Principal Investigator Meeting
January 9-11, 2017
Arlington, Virginia