Thin Hypervisors and VMI (Virtual Machine Introspection) Security Projects

236349

Winter 2017-2018

Contact Person:
Dr. Sara Bitan

email: sarab@cs.technion.ac.il

Phone: 077-8875854

Background:

Stealth control and monitoring are security requirements in many scenarios. For example, malware researchers who analyze samples, run them, in a sandbox to prevent possible malware outbreak. However, sophisticated malware often modifies its behavior when it detects execution in sandbox, to look benign.

VMI – Virtual Machine Introspection mechanisms consists of modules in a hypervisor, that inspect the state of a guest machine. From the nature of VMM, it stems that all the guest state information – memory, registers, virtual I/O, is open to the inspection and analysis. VMI can be very useful for intrusion detection, malware analysis and research.

The goal of these projects is to develop stealth VMI mechanisms.

Project 1: Stealth Thin Hypervisor

Description:

A thin hypervisor is a small size hypervisor (several K bytes of code), that can intercept and handle system event. Our goal is to develop a stealth thin hypervisor, i.e. that cannot be detected from a guest operating system,

Platform:

SimpleVisor https://github.com/ionescu007/SimpleVisor will be used as the base to this project.

Project Requirements:

- Install a thin hypervisor before the OS is loaded into memory, i.e. during UEFI execution.
- Test the thin hypervisor on a common pc platform (TBD)
- Define a set of tests designated to detect execution in a virtual machine running Windows 10
- Run the test in Windows 10 guest OS over the thin hypervisor, and provide the results

Pre-requisites:

Kernel programming knowledge.

Deep understating of operating systems.

Project 2: VMI

In this project we will use a thin hypervisor as a guest system monitoring platform.

Hyperplatform is a system monitoring platform for the Windows OS, based on VT-X and EPT (Extended Page Table). It uses VM-exit handler to intercept and monitor events on the Guest OS.

Currently there three VM-exit handler using Hyperplatform, MemoryMon, GuardMon and EoPMon.

- MemoryMon detects execution of Kernel memory which is not backed up by image file.
- GuardMon detects access to system register, e.g. CR0, CR4, GDT and IDT, from kernel memory not backed up by image
- EoPMon detects Elevation of Privileges. It can detect a process with stolen system token.

Platform:

Hyperplatfrom https://github.com/tandasat/HyperPlatform .

Project Requirements:

- Define two system events typical to malware
    - OS – Windows 10
- Design and implement a vm-exit handler that monitor this event
- Infect a guest OS with malware using the above event
- Monitor a virtual machine from a thin hypervisor, and detect the events.

Pre-requisites:

Kernel programming knowledge.

Deep understating of operating systems.

Deep understanding of virtualization (specifically VT-x) and the extended page tale mechanisms.

Familiarity with malware.

References

1. A Virtual Machine Based Introspection, https://suif.stanford.edu/papers/vmi-ndss03.pdf
2. EoPMon GitHub, https://github.com/tandasat/EopMon
3. GuardMon GitHub, https://github.com/tandasat/GuardMon
4. HyperPlatform GitHub, https://github.com/tandasat/HyperPlatform
5. MemoryMon GitHub, https://github.com/tandasat/MemoryMon
6. Monitoring and Controlling Kernel-mode Events by HyperPlatform, Recon 2016, Presentation slides, https://recon.cx/2016/resources/slides/RECON-0xA-HyperPlatform-Satoshi.pdf
7. Monitoring and Controlling Kernel-mode Events by HyperPlatform, Recon 2016, Talk Video, https://www.youtube.com/watch?v=G-X6g4zkNtE
8. Monitoring and Controlling Kernel-mode Events by HyperPlatform, Igor Korkin's Blog, http://igorkorkin.blogspot.co.il/2016/06/monitoring-controlling-kernel-mode.html
9. SimpleVisor GitHub, https://github.com/ionescu007/SimpleVisor