Hypervisor Internals for Vulnerability Researchers

Training Class Description

Overview

This training class is a quick start on vulnerability discovery in arbitrary virtualization systems, authored and taught by a professional zero-day vulnerability researcher and reverse engineer.

Abstract

Virtualization software invisibly permeates our lives. You deal with it transparently each time when you work with a public cloud instance that may be backed by Microsoft Hyper-V or KVM, or a corporate workstation hosted on a VMWare ESXi server. Even some modern smartphones rely on a hardware-based hypervisor under the hood for enhanced security and privilege separation.

As of 2020, virtualization systems are one of the hottest targets in offensive vulnerability research. Top cash rewards are offered for zero-day security bugs in popular virtualization systems by certain software vendors' Security Bounty Programs[1], third party exploit brokers, and international hacking competitions[2]. One reason for that is the omnipresence of virtualization systems, which is coupled with their business-critical reliability and security requirements. A single virtual machine escape exploit targeted to a popular corporate hypervisor or a public cloud hypervisor can have an enormous impact on information privacy by allowing unauthorized remote access to co-located virtual instances, the virtualization server itself and the infrastructure connected to it, while a single persistent DoS exploit can incur dramatic business losses in unreceived profit due to availability issues with the customer-facing infrastructure.

Another reason is the high complexity and technical advancedness of this class of systems. A typical virtualization system consists of numerous subsystems that provide support for multiple

[1] Microsoft Hyper-V Bounty https://www.microsoft.com/en-us/msrc/bounty-hyper-v
[2] pwn2own competition 2020 rules https://www.zerodayinitiative.com/Pwn2OwnVancouver2020Rules.html

operating systems and hardware device models, of which the hypervisor is only one core part. It may be backed by hardware virtualization capabilities or purely software-based, executing on bare metal or an underlying operating system, working together with emulated, virtualized, or paravirtualized hardware device modules, guest additions or services, and other subsystems, all of which constitute a valid attack surface for a virtual machine escape or an information leakage attack.

System architecture details and internals of virtualization systems are mostly undocumented and unregulated. The books "Theory of Virtualization Systems" and "Design and Implementation of Hypervisors" do not exist yet. Development of virtualization systems is a living process, spearheaded by the open source community on the one side, and system architects of proprietary software products on the other. Because of that, design and implementation properties of various virtualization systems vary greatly, while sharing plenty of technical details, and indeed entire subsystems, that may be influenced, borrowed, or inherited from competitors.

Despite implementation-level differences, all modern virtualization systems share a common abstract model, which is suggested and enforced by their deployment purposes and responsibilities. The goal of this training is to introduce the students to efficient vulnerability discovery in arbitrary virtualization systems through the perspective of their common abstract model of internals, implementation-specific technical case studies, and the author's many-year experience in operating systems introspection and vulnerability research. Major implementations in scope are VirtualBox, VMWare Workstation, and a bit of Microsoft Hyper-V.

The structure of this training is a mix of theoretical and hands-on experience. Each training day will be divided into four 1.5-2-hour topical blocks. Each topical block will consist of a theoretical lecture, a practical exercise/lab/CTF task, and finally, a solution or a walkthrough. The author's personal cognitive optimization and time management techniques will be employed for better learning efficiency.

This is a deep technical class of intermediate to advanced level, focused on system internals for vulnerability discovery. Students will be expected to have solid knowledge of x86_64 assembly and C/C++, fundamental Operating Systems concepts, some prior experience with software vulnerabilities, and confident mastery of a Linux command line and building environment.

Materials of this training class are based on independent technical research and reverse-engineering performed by the author in 2016-2018, and illustrated with zero-day vulnerabilities discovered by the author herself.

Learning Objectives

- Review essential OS theory and hardware theory which is relevant to hypervisor security research.
- Comprehend the abstract design model of a virtualization system, and how to apply it to generalized vulnerability discovery.
- Identify common attack surfaces, attack vectors, and classes of vulnerabilities in virtualization systems.
- Get a high-level overview of implementation specifics of dominant virtualization systems.
- Investigate in-depth internals of selected virtualization subsystems through open source implementations and vulnerability cases.
- Familiarize with the historical record and technical details of known security bugs in virtualization systems.
- Learn to set up a research platform for mainstream virtualization systems (compilation, debugging, fuzzing environment).
- Create a vulnerability proof-of-concept from a binary security patch.
- Review essential research publications and trends.

Daily Plans

Day 1: Foundation.
Top level abstract models. Relevant OS theory. Source-level case studies. Userland-reachable attack surface. Bugs in guest additions. VirtualBox specifics.

Day 2: Advanced Topics.
Hardware device models. Relevant hardware theory. Binary-level case studies. Kernel-accessible attack surface. Bugs in virtualized and emulated devices. Binary patch analysis. VMWare Workstation.

Day 3: More Advanced Topics.
Hypervisor monitor model. Relevant OS & hardware theory. Deep, unique, and hypervisor-specific bugs. Review of research publications. Microsoft Hyper-V.

To Do: Details

Who Should Attend

This class is primarily intended for professional security and vulnerability researchers coming from other specialization areas who wish to re-focus on virtualization systems, familiarize with popular hypervisors from the security research perspective, or extend and enhance their in-depth understanding of modern computer systems in general.

Materials of this class constitute essential knowledge for aspiring zero-day vulnerability researchers who are considering or have chosen Hyper-V, VMware Workstation, VirtualBox or other virtualization system as their research target.

Security engineers and software developers of low-level software and firmware systems, C-level technical staff of cloud solutions providers, and newcomers to vulnerability research will benefit from this class.

Prerequisites

- x86_64
- C/C++
- Operating Systems concepts
- Linux command line and build environment.

Hardware and Software Requirements

- Productivity-grade laptop with hardware support for nested virtualization (see Notes below).
- Ubuntu Linux 18.04 LTS x64 (either VM or native) - **REQUIRED**.
- Professional disassembler.
- IDE with C/C++ syntax intelligence.

Notes:

1. Essential labs will be based on a self-compiled VirtualBox on Ubuntu Linux 18.04 LTS x64 with a Linux x64 guest OS. Nested virtualization is only required to run a VM inside a VM.

2. VirtualBox can only support nested virtualization on AMD hardware.[3] VMWare Workstation and Parallels for Mac have full support for nested virtualization.
3. Recommended hardware/software options:
    a. MacBook Pro with Mac OS X + Parallels Workstation + VM Ubuntu Linux 18.04 LTS x64.
    b. Laptop with Ubuntu Linux 18.04 LTS x64.
    c. Laptop[4] with Windows 10 x64 + VMWare Workstation + VM Ubuntu Linux 18.04 LTS x64.
    d. AMD-based laptop[4] with any OS supported by VirtualBox + Oracle VirtualBox + VM Ubuntu Linux 18.04 LTS x64.

Students must test their hardware/software setup for compatibility with Notes pt.1 prior to attending the class.

Class Dates & Locations

- 31 January - 02 February 2020 (3 days)
  Tel Aviv, Israel
- 17-19 February 2020 (3 days)
  Berlin, Germany
- 3-5 March 2020 (3 days)
  Goa, India

Class Times: 12:00 to 20:00.
Lunch at 13:30, coffee break around 18:00.

Registration

Registration fee for any 3-day class: 4000,- EUR.
Welcome Discount: 25% (first 3 seats in each class).
Early rate: 3500,- EUR (until 25 January).

To book a seat, send an email request to class@alisa.sh.
All trainings will be organized by An Exponential Limited, Hong Kong CR no.2898486.

[3] https://docs.oracle.com/cd/E97728_01/F12469/html/nested-virt.html
[4] Hardware support for nested virtualization is required for this option.

About the Author

Alisa Esage (Alisa Shevchenko) is an internationally recognized computer security researcher, hacker, and an entrepreneur. As a technical expert, Alisa is specialized on target-invariant zero-day vulnerability discovery and exploit development, reverse engineering, and low-level system internals. Alisa has a Zero Day Initiative's Silver status of 2018, the 1st award in "Hack the Smart City" competition of 2014, and a track record of zero-day security bugs discovered in nearly all dominant software systems. Alisa is the author of the Research Notes project on undocumented system internals.

More info: alisa.sh.
Mail list: club.alisa.sh.
Twitter: @alisaesage.