kaspersky.academy

Malware Reverse Engineering

Curriculum

Victor Chebyshev, Boris Larin
Kaspersky Lab

17.06.2019

kaspersky.academy

Malware Reverse Engineering

The ability to dissect, analyze executables and scripts, as well as understand what they do, is becoming an important skill not only for an information security engineer but for almost any IT related specialist.

The hands-on, practice-oriented format of this course will allow students to rapidly obtain skills needed for static and dynamic analysis.

As a result, students will gain a deeper and more thorough understanding of applications and operating systems, they would learn how to analyze and reverse engineer software, avoid common software bugs, exploited by attackers and build better, more secure applications.

Course Duration

44 instructor-led hours (+ 70 hours of individual study)

Objectives & Learning Goals

This course is meant as a practical course on malware reverse engineering and analysis for beginners.

It will start with an introduction to the main concepts and terms needed, as well as an assembly language crash course for those with no prior experience. We will learn what approaches and tools can be used and how to perform static and dynamic analysis of malicious executables for multiple platforms – Windows, Linux, MacOS, and Android.

The course will also cover some of the more advanced topics on software vulnerabilities and exploits analysis, reverse engineering byte-code and script languages, automating reverse engineering tasks, unpacking and deobfuscating.

Methodology

The course will be organized as follows: each three-hour session will consist of a lecture and a practical part. In the practical part of the course, students will solve problems related to the topic of the lecture and get practical assignments.

Evaluation and grading

Evaluation is performed based on class work (50%), homework (25%), midterm and final exam (25%).

kaspersky.academy

Course Outline

1. Introduction to the main concepts and terms
   a. Course program and plan;
   b. Setting up needed software and safe environment;
   c. Modern types of threats and malware classification;
   d. Types of analysis (Basic static, static, dynamic / behavior, advanced dynamic);
   e. Disassemblers and decompilers;

2. x86 and x64 Architecture and Assembly
   a. Processor architectures, CISC vs RISC;
   b. Fundamental data types;
   c. Endianness;
   d. x86 and x64 architecture;
   e. Memory organization – memory models, paging and virtual memory;
   f. Register set, main purpose hardware registers;
   g. Assembly language – instruction set, opcodes, mnemonics, operands and examples;
   h. Stack and heap;
   i. Function calls – calling conventions, stackframe, epilogue\prologue;

3. Static Analysis
   a. Basic static analysis. Approaches and mindset;
   b. Strings, entropy and hash analysis;
   c. Portable Executable header analysis;
   d. PE resources, overlay, imports, compiler and protection analysis;
   e. PE signature, publisher information and file icon analysis;
   f. AV scanning, Virustotal and web research;
   g. Advanced static analysis;

4. Dynamic Analysis
   a. Dynamic Analysis: Why? When? How?;
   b. Using system monitoring utilities to capture file system, registry and network activity;
   c. Monitoring process activity;
   d. Monitoring APIs;
   e. Monitoring network;

5. Malware behavior
   a. Windows malware techniques;
   b. Malware persistence;
   c. -Anti-analysis - obfuscation, anti-debugging, anti-emulation, etc.;
   d. Packers, crypters and protectors. Unpacking malicious samples.
   e. Debugging windows applications using x64dbg and Windbg.

6. Byte-code based and scripting languages
   a. Native code vs Byte-code;
   b. Virtual Machines, Binary Translation, Code Emulation, JIT;
   c. Main analysis techniques and tools.
   d. Analysis of .NET malware
   e. Fighting obfuscation techniques

7. Software vulnerabilities and exploits
   a. Classification;
   b. Types and causes for binary vulnerabilities;
   c. Common exploitation technics;

   d. Shellcode analysis;
   e. Modern Operating System security features;
   f. Bypassing OS exploit preventions and mitigations.

8. Exploits Analysis
   a. MS Office Exploits;
   b. Web browser Exploits;
   c. Flash Exploits;

9. Non-windows malware - Linux
   a. Statistics, attack vectors;
   b. Operations system security basics;
   c. Static analysis: ELF file format, IDA, HIEW;
   d. Dynamic analysis: file behavior, strace, remote gdb, unpacking.

10. Non-windows malware – Mac OS
   a. Statistics, attack vectors;
   b. Operations system security basics, internal AV, single source application distribution;
   c. Static analysis: MACH-O file format, objective-c constructions;
   d. Dynamic analysis: ptrace, remote GDB.

11. Non-windows malware – Android
   a. Statistics, attack vectors: unknown sources, exploit vector;
   b. Operations system security basics: Sandbox, SafetyNet, rooting, device admin tricks;
   c. Static analysis: APK file format, decompilation, pseudo Java.

kaspersky.academy

Practice Sessions

All practical exercises can be combined into three thematic units:

1. Windows Malware Reverse Engineering
2. Exploits and Vulnerabilities Reverse Engineering
3. Non-Windows Malware Reverse Engineering

Windows Malware Reverse Engineering

Setting up a safe environment

- Setting up host machine
- Setting up guest virtual machine
- Setting up Virtualbox

Practical Labs, discussed in this course can be potentially dangerous to your host machine. As it can infect and corrupt files on your computer, start spreading via network and be very hard to remove we first need to properly configure host and virtual machine, as well as the hypervisor that is running the VM. Once all of this is done, we can carry on our analysis in a controlled environment with no risk of infection.

Assembly

- Assembly Samples

In this session we will statically analyze a few simple samples by reading their disassembly code using IDA Pro. As a result of this lab we will gain deeper understanding of what the malware does by analyzing its code.

Static analysis

- String, Entropy, Hash Analysis
- Portable Executable Analysis
    - Portable Executable Header
    - Debug Information
    - Compiler and Protection
    - Resource
    - Import
    - Digital Signature, Publisher and Icon Information
- Advanced Static Analysis

Here, the students will perform basic static analysis using multiple tools – HIEW, CFF Explorer, DiE, Exeinfope, Resource Hacker, sigcheck, strings and others. A lot of the malware indicators can be found statically in the Portable Executable file itself. Understanding those indicators, and knowing how to extract them is a crucial skill for malware analysis process. In the last part of the session we will show how advanced static analysis can benefit an analyst and help gain deeper knowledge of the program behaviour.

Dynamic analysis

- Monitoring Process Activity
    - Failcryptor
    - Inject sample
- Monitoring APIs

    - Zbot
- Monitoring Network
    - Banking Trojans
    - Setup
    - File Transfer

During this session students will observe various ways to observe behavior of the sample dynamically, with the help of different tools used by malware analysts. We will show how you can detect malware activity by observing its process, API and network activity.

Malware Behaviour

- Persistence
    - Winlogon Events
    - Browser Helper Objects
- Unpacking
    - UPX
    - WinUPack

In this lab, we will study how to detect different malware persistence techniques using Process Monitor, Regshot and Autoruns utilities.

In the second part of this lab students will manually dynamically unpack two popular packers – UPX and WinUpack using x32dbg and Scylla.

Exploits and Vulnerabilities Reverse Engineering

.NET Malware analysis

During this session, students will practically use knowledge about .NET executable files reverse engineering basics. They will analyze samples with modern reverse engineering tools both statically and dynamically. Students will learn how to work with packed .NET samples, how to debug and unpack them.

Shellcode analysis

During this session, students will be introduced to analysis of shellcodes, their tricks and how to deal with them. They will get familiar with debugging using WinDbg and will analyze most common examples of shellcodes with IDA Pro.

Microsoft Office Exploit Analysis

During this session, students will learn how to analyze Microsoft Office documents, fight real world obfuscations, and analyze all stages of modern Microsoft Office exploits.

Web browser Exploits

During this session, students will learn how to replay malicious web traffic in analysis purposes, how to debug and deobfuscate malicious web scripts. Besides that, students will get familiar with analysis of exploits for web browsers during analysis of Internet Explorer “god-mode” exploit.

kaspersky.academy

Flash Exploits

During this session, students will learn how to analyze Adobe Flash files, will get familiar with Adobe Flash exploits and how to analyze them.

Non-Windows Malware Reverse Engineering

Linux malware reversing basics

Practical workshop will be performed using student's notebooks, using virtual machines in VirtualBox.

During this session, students will practically use knowledge about ELF files reverse engineering basics. They will personally see interesting points in the file, analyze the file statically using HIEW and IDA. Students will learn how to deal with packed ELF files, learn how to identify file functionality.

Mac OS malware reversing basics

During this session, students will practically use knowledge about Mach-O files reverse engineering basics. They will personally see interesting points in the file, analyze the file statically using HIEW and IDA.

Android malware reversing basics

During this session, students will practically use knowledge about APK files reverse engineering basics. Students will learn how to deal with packed APK files, learn to identify entry point if the package, learn how to unpack files, packages structure, how to decompile DEX files.

kaspersky.academy

Bibliography

Core materials:

- Michael Sikorski, Andrew Honig – Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software
- Dennis Yurichev – Reverse Engineering for Beginners (https://beginners.re/)
- Daniel Barrett, Richard Silverman, Robert Byrnes - Linux Security Cookbook
- Jonathan Levin - Android Internals::A Confectioner's Cookbook
- Bruce Dang, Alexandre Gazet, Elias Bachaalany - Practical Reverse Engineering: x86, x64, ARM, Windows Kernel, Reversing Tools, and Obfuscation

Additional materials:

- Amit Singh - Mac OS X Internals: A Systems Approach
- Pavel Yosifovich, Alex Ionescu, Mark E. Russinovich, and David A. Solomon – Windows Internals, Seventh Edition, Part 1

Prerequisites

Good knowledge of operating system concepts, as well as familiarity with high level and low level system programming. C\C++\Assembly and any scriptable language (Python, Ruby, Javascript, etc.) experience is highly recommended.

Knowledge of cryptography and network fundamentals would be a plus.

kaspersky

www.kaspersky.com/
www.securelist.com