CFRS 775-001: Kernel Forensics and Analysis

Computer Forensics Program (http://cfrs.gmu.edu)
Department of Electrical and Computer Engineering (https://ece.gmu.edu/)
Volgenau School of Engineering (http://volgenau.gmu.edu/)
George Mason University (http://www.gmu.edu/)
Spring 2017 Calendar (https://registrar.gmu.edu/calendars/spring-2017/)

Instructor
Name: T.Roy
Title: Adjunct Professor
Email: troychou@gmu.com
Office Hours: Via email throughout the semester. If required, instructor will schedule a phone and skype call with individual students.
Office Location: Virtual

Course Details
Course Number and Section: CFRS 775 001
Credit Hours: 3
Meeting days and times:

#	Date	Day	Hours
1	Jan 28, 2017	Saturday	0900 - 1800
2	Jan 29, 2017	Sunday	0900 - 1800
3	Feb 04, 2017	Saturday	0900 - 1800
4	Feb 05, 2017	Sunday	0900 - 1800
5	May 13, 2017	Saturday	0900 - 1800

Building and room: GMU Fairfax Campus, Nguyen Engineering Bldg., Room 4457

Prerequisites
CFRS 761-001 Malware Reverse Engineering
Working knowledge of Windows
Proficiency in C/C++ programming
Familiarity with X86/X64 assembler is preferred but not required.

Course Description

To achieve maximum stealth and obtain unabated access to the system, rootkits execute in kernel mode. In order to identify rootkits it is critical to understand how the kernel works and how the mechanisms provided by the kernel are exploited by rootkits for malicious purposes. Artifacts left behind by rootkits in various places in the system, as a part of the hooks they place, can be detected by various forensic analysis tools. The course starts off by introducing students to the Windows kernel

development and debugging environment. It then dives into CPU architecture followed by kernel components, algorithms and data structures. Once students understand how the kernel works, the course shifts focus to malicious activity in the kernel like call flow diversion, data structure manipulation, covert communications followed by some of the security mechanisms that have been added to the kernel in recent version of Windows. The course concludes with a study of recent Windows kernel rootkits and discussing live and post-mortem forensics tools that help identify indicators of compromise in the kernel. Students will attend lectures, work on programming, debugging and reverse engineering assignments and complete a rootkit analysis project.

Course Objectives

This course introduces students to the internal working of the Windows kernel, describes the different ways in which rootkits exploit the kernel, and the tools and techniques for detecting presence of malicious activity in the kernel. Upon attending this course students will have good understanding of how the Windows kernel works, be able to develop kernel modules for Windows and be able to perform forensic analysis of systems to identify rootkits.

Grading

Criteria	Percentage
Assignments	30%
Final	30%
Project Report and Presentation	30%
Class Participation	10%

Class Sessions and Topics

#	Dates	Topics
1	Jan 28	Windows kernel environment Windows kernel development and debugging environment. Windows Driver Kit (WDK). Code, Build, Deploy, Debug and Test kernel driver in a virtual machine.
2	Jan 28	Windows kernel debugging Discussing the kernel debugger architecture, debugger components and their usage for live and post-mortem debugging as well as for kernel reverse engineering.
3	Jan 28	Hardware Support Focus on those features of 32-bit and 64-bit Intel/AMD CPUs that are

		important for kernel functionality and how they can be used for malicious purposes. Usage of debugger to peer into these CPUs features and understand their usage by the Windows kernel.
4	Jan 29	Kernel software development Coverage of common tasks that are required in most kernel drivers including communication with user mode applications, memory allocation, registry and file system access.
5	Jan 29	Windows kernel internals I Introduction to kernel code execution environment and kernel memory management.
6	Jan 29	Windows kernel internals II Introduction to kernel objects, handle management and driver architecture.
7	Feb 04	Kernel hooking techniques How malware and anti-malware hook into the Windows kernel to gain execution and perform code flow diversion.
8	Feb 04	Kernel data structure modification How rootkits use direct kernel object manipulation (DKOM) to escalate privileges and hide their presence from the rest of the operating system.
9	Feb 04	Covert network communications Overview of the networking components in the Windows kernel, they APIs they provide and how they are used by rootkits to communicate with command and control servers.
10	Feb 05	Kernel security mitigations Overview of the security mitigations added by Microsoft in recent version of Windows, the types of attacks they mitigate and how some of them are bypassed by malware.
11	Feb 05	Kernel forensics Overview of the various live and post-mortem forensics tools available for the Windows platform, their usage in identifying kernel subversion and how some of these can be bypassed using anti-forensic techniques.
12	Feb 05	Kernel rootkits case study Discuss the real-world application of the offensive techniques covered throughout the course and study how they are exploited by contemporary rootkits.
13	May 13	Final Exam covering ALL course topics: 0900 – 1130 hours Individual Project Presentations: 1300 – 1800 hours

Assignments

There will be a total of six (6) assignment comprising of development environment setup, investigating internals of the kernel with a debugger and developing kernel mode modules for Windows. Students will have 2 weeks to complete each assignment. All assignments must be performed on Windows 7 SP2 (latest updates rollup) 64-bit edition. Assignments involve code development (C and x64 assembler) and usage of kernel debugger.

#	Due Dates	Topics
1	Feb 18	Kernel Development Environment Develop, build, deploy, test and debug a simple kernel mode driver.
2	Mar 04	Kernel Debugger Usage Use the kernel debugger to display values of low level hardware registers and data structures.
3	Mar 18	Kernel Programming Environment Develop a kernel mode driver that maintains a count of the number of times it has been loaded.
4	Apr 01	Kernel Mode Code Signing Implement a driver that can load and execute on the target 64-bit system without a kernel debugger being attached.
5	Apr 15	Kernel Mode Code Subversion Implement a kernel module that installs an inline hook in any Microsoft provided 64-bit kernel module.
6	Apr 29	Covert Network Communication Develop a kernel module that uses the WSK API to perform DNS lookups.

Tests

There will a single final test consisting of multiple choice questions, free form questions and verbal Q&A. All topics discussed in sessions 1 through 12 would be covered in the final test.

Project

The objective of this project is to apply the knowledge gained through this course to analyze real world malicious software.

Students will perform research on detection of artifacts left behind in system by a contemporary Windows kernel mode rootkit and publish a comprehensive report on the topic along with a 15-minute presentation followed by a Q&A session.

Projects must be performed individually.

If you choose to do a different project, something you feel would more useful to you, it must be first approved by the instructor. The last date for alternative project approval is Feb 05, 2017.

#	Milestone	Due Date
1	Project Proposal	Feb 18, 2017
2	Project Report	Apr 29, 2017
3	Project Presentation	May 13, 2017

Text Books

Required Text:
The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System (2nd Edition) by Bill Blunden, Jones & Bartlett Learning 2012.
(http://www.amazon.com/The-Rootkit-Arsenal-Evasion-Corners/dp/144962636X)

Recommended Reading:
Windows Internals: Part 1 and Part 2 (6th Edition) by Mark Russinovich, David Solomon & Alex Ionescu, Microsoft Press 2012
(http://www.amazon.com/Windows-Internals-Part-Developer-Reference/dp/0735648735)
(http://www.amazon.com/Windows-Internals-Edition-Developer-Reference/dp/0735665877)

Equipment Requirements

Students must attend each class with their own laptop with the required tools (listed below) fully setup. The instructor will conduct live demos to help understand the theory discussed in the class and students are expected to be able to follow along and replicate the steps on their own systems. Students must also use their own systems to perform the assignments and the project.

Hardware	▪ Hardware virtualization (VT-x) capable CPU(s) ▪ Minimum 8GB of RAM (for running one guest VM) ▪ Minimum 40 GB free disk space ▪ Working USB Port ▪ Working Wireless LAN
Software	▪ Host OS Windows 8 or above Enterprise 64-bit Edition Evaluation version of Windows is fine. ▪ Virtualization (Hyper-V is a built-in feature of enterprise edition) Enable this via Control Panel -> Programs and Features -> Turn Windows Features on or off -> Hyper-V. You choose to use any other virtualization software, but please don't expect any support from the instructor. ▪ Guest OS Windows 7 SP2 (64-bit version). Do not use Windows 8, 8.1 or Windows 10. Evaluation version is fine. ▪ Administrator access required on both host and guest OSs. ▪ Windows Driver Kit (Windows 7 SP1 v7.1.0) https://download.microsoft.com/download/4/A/2/4A25C7D5-EFBE-4182-B6A9-AE6850409A78/GRMWDK_EN_7600_1.ISO

	▪ Debugging Tools for Windows v10.0.14321.1024 x64 http://www.codemachine.com/downloads.html ▪ Favorite text editor Visual Studio 2015 Community Edition will work fine. ▪ Volatility Framework Standalone (v2.6) http://www.volatilityfoundation.org/26 All other software will be provided by the instructor.
Internet	Internet access will be required for all labs to download symbol files from Microsoft's symbol servers.

Honor Code

Students are required to be familiar and comply with the requirements of the GMU Honor Code. The Honor Code will be strictly enforced in this course.

Students with disabilities

Students with disabilities who seek accommodations in a course must be registered with the GMU Office of Disability Services (ODS) and inform the instructor, in writing, at the beginning of the semester. See http://ds.gmu.edu or call 703-993-2474 to access the ODS.