NYU Tandon School of Engineering
Center for Cyber Security (CCS)

Special Topics Course:

The Art of Binary Exploitation: Mobile and Embedded Systems

3 Credits

OVERVIEW

This special topics course covers the art of binary exploitation of software and embedded firmware (for mobile and embedded architectures) while exploring the history of the fundamental exploitation primitives across various architectures. Topics: software reverse engineering, ARM, binary exploitation, software exploitation, cybersecurity, protection methodologies.

PREREQUISITE(S):

Graduate Status | Weekly Lecture Hours: 1.5 | Weekly Lab Hours: 1.5 | Weekly Recitation Hours: 0

MATERIALS:

Course Lectures will consist of over 900 slides, 15-20 guided hands-on lab sessions (with 100+ page lab manual). Students will be provided with remote access to their own preconfigured Virtual Lab environments.

INSTRUCTOR:

Stephen A. Ridley

GOALS & LEARNING OBJECTIVES

Participants will get hands on experience performing each type of exploitation while also learning the history of the exploitation primitive itself: the discoverer, prior art, and about how to evade related protections. Using embedded architectures as a case study the student will learn about the modern protection mechanisms of operating systems and chip architectures and how they can be circumvented.

Week 1: Foundations of Mobile and Embedded System Design

Students will learn about the history of the more popular embedded architectures with the focus on ARM as a basis to compare with Desktop/Server architectures. In the process of exploring this, students will review (and/or get familiarity with the debugging and disassembling tools we will be using in this course, GDB, IDA, Ghidra, Capstone Engine, etc.)

Week 1 Labs and Assignments: Students will spot vulnerabilities in C source code. Students will trace through execution of various ARM applications to learn how static and dynamic analysis can compliment one another in ways that would otherwise take much longer when only using one analysis technique.

Week 2:

Students will learn about dynamic linking with a focus on why shellcode evolved the way it did historically. Students will learn about shellcode “loaders” and how they are used to exploit processes.

Week 2 Labs and Assignment: Students will write their own shellcode from scratch for the purpose of exploiting several specific apps.

Week 3:

Students will use the shellcode they hand-crafted in the previous unit to exploit a stack overflow. Students will learn about NOP sleds, bouncepoints, and trampolines while simultaneously learning about how to subvert some protection mechanisms such as stack cookies.

Week 3 Labs and Assignment: Students will exploit an ARM stack overflow by hand, learning how to manipulate the size of payloads, and experiencing some of the runtime nuances of stack-overflow exploitation.

Week 4:

Students will be introduced to the fundamentals of Return Oriented Programming in this unit when leveraging it to defeat the NX “no execute stack” protection mechanism of ARM chips.

Week 4 Labs and Assignment: Students will target and exploit an application leveraging the “gadgets” found in the ARM application (with more detailed ROP gadget hunting methodologies expounded in later units).

Week 5:

In a continuation of further ROP exploration, students will be introducted to more advanced stack exploitation techniques the mirror the reality of exploitation of real-world services (handling multi-threaded conditions, remote-services, post-exploitation process continuation, and compound protection mechanisms). Students will learn about overwriting stack cookies, info disclosure, partial and full pointer overwrites, exception-handler overwrites.)

Week 5 Labs and Assignment: Students exploit network based services that use XN and Stack Cookies leveraging partial pointer overwrites and simple ROP techniques.

Week 6:

Thorough exploration of the various uses of ROP techniques which comprises the majority of modern exploitation techniques. Students will learn about all the various protection mechanisms that necessitate the use of ROP exploitation techniques (code signing, ASLR, cookies, environment uncertainty). Discussion of some of the more advanced methods of finding and chaining gadgets will be discussed, but students will learn the fundamentals of manually finding gadgets and crafting ROP payloads.

Week 6 Labs and Assignment: Students will find and exploit a vulnerable application by building a custom payload by manually stringing together a bunch of ROP gadgets..

Week 7:

Advanced ROP. Continuing with some of the material from the previous unit, students will learn about the nuances of handcrafting ROP payloads (binary target version differences, execution environment uncertainty, et al).

Week 7 Labs and Assignments: Students will exploit a vulnerable application by handcrafting a ROP payload (with given gadgets) to obtain a modern full “connect-back” rootshell in the exploited process.

Week 8:

Students will be introduced to the core primitives of Heap exploitation by exploring the core implementation differences of various heap implementations (tcmalloc, dlmalloc, pools, thread-cache freelists, et al.). Students will also be introduced to the core concepts of bespoke remote heap-manipulation in target applications.

Week 8 Labs and Assignments: Student will exploit a network server, associating “protocol events” with process execution and consequently in-process dynamic memory layout.

Week 9:

Advanced Heap

Continuing with heap exploitation, students will learn about Singly Linked List Exploitation and how it relates to the fundamentals of heap exploitation, then see (first hand) the way that some heap-overflow mitigations were designed to protect from simple conditions. Using a specific modern allocator implementation as a case-study (TCMalloc) students will see how to evade the protections and ultimately successfully exploit modern heap implementations.

Week 9 Labs And Assignments: Students will exploit a network server that uses a real heap implementation (TCMalloc) leveraging a (Write-Me-Where primitive) by associating network protocol events with memory allocations that ultimately will all them to manipulate remote target system memory enough to ultimately groom the target into ideal exploitation state allowing complete compromise with their customized payload.

Week 10:

Students will combine all the cumulative subjects of previous weeks in this course to see how hybrids of those techniques can be leveraged to exploit "application specific" vulnerabilities (i.e. vulnerabilities in the specifically coded business logic of applications). Students will also explore how "application specific exploitation" tends to comprise the majority of modern exploitation, and discuss why this is. Students will dive deep into the construction of Object-Oriented Languages (such a C++) to discuss virtual table pointer overwrites.

Week 10 Labs and Assignments: Students will exploit VTable overwrites and explore techniques for investigating these kinds of vulnerabilities.

Week 11:

MultiHeap and Real-World Exploitation

Students will combine all the techniques for no-execute stack, ROP, simple and advanced stack overflows, Object Oriented Programming nuances, and Application Specific Exploitation, to build a simple and powerful technique for bypassing the most common of modern protection mechanisms.

Week 11 Labs and Assignments: Students will use a heapsprage or similar technique to overwrite a C++ object to create a fake vtable entry that redirects the stack into the heap, thereby "stack flipping” and bypassing most of the modern protection mechanisms on modern embedded and mobile operating systems.

Week 12:

ASLR and Defeating ASLR

We will discuss the many implementations of Address Space Layout Randomization and how (combined with other protections like XN) is did so much for making simple exploitable conditions VERY difficult to exploit. We will review some of the academic work that is being done to further protections like this (pointer encryption ARM8.3, Windows Control Flow Guard, etc.) but also examine how ASLR-related implementations gave birth to many of the above. Finally we will explain how ASLR can be bypassed and investigated.

Week 12: Lab and Assignments Students will find and exploit an information disclosure vulnerability which will help identify the layout of memory in the remote process, to inform their manual crafting of their ROP payload which will successfully evade XN and ASLR using their own manually created exploit and payload..

Week 13

Course Examination:

A comprehensive examination will be given to students that will cover all course topics from the weekly lectures and lab exercises.

Week 14:

Team/Project Presentations

Students will give lectures on their projects and will deliver a written paper to the class for review and comment, OR, students will document their exploration of a real-world target. From research/discovery through (un)successful exploitation, with clear articulation of what real-world challenges were encountered. Projects will be done individually or in groups as per agreement from the instructor.

Week 15:

Team Projects and Presentations 2

Students will give lectures on their projects and will deliver a written paper to the class for review and comment, OR, students will document their exploration of a real-world target. From research/discovery through (un)successful exploitation, with clear articulation of what real-world challenges were encountered. Projects will be done individually or in groups as per agreement from the instructor.