Modern Binary Exploitation - Course Syllabus

Course Information

Course Title: Modern Binary Exploitation
Course Number: CSCI 4968
Credit Hours: 4
Semester / Year: Spring 2015
Meeting Days: Tuesday/Friday 2-4PM
Room Location: Walker 5113
Course Website: http://security.cs.rpi.edu/courses/binexp-spring2015/
Prerequisites (one of the following or permission of instructor):

- CSCI 2500 - Computer Organization
- ECSE 2660 - Computer Architecture, Networks, and Operating Systems

Instructor

Name: Bülent Yener
Office location: Lally 310
Email Address: yener@cs.rpi.edu

Teaching Assistant(s)

TAs: RPISEC
TA Office Location: Sage 3101
TA Office Hours: Wednesday 7-10PM
TA Email Address: binexp_ta@cs.lists.rpi.edu

Course Description

Cybersecurity is one of the fastest growing fields in computer science, though its study is rarely covered in academia due to its rapid pace of development and its technical specificity. Modern Binary Exploitation will focus on teaching practical offensive security skills in binary exploitation and reverse engineering. Through a combination of interactive lectures, hands on labs, and guest speakers from industry, the course will offer students a rare opportunity to explore some of the most technically involved and fascinating subjects in the rapidly evolving field of security.

The course will start off by covering basic x86 reverse engineering, vulnerability analysis, and classical forms of Linux-based userland binary exploitation. It will then transition into protections found on modern systems (Canaries, DEP, ASLR, RELRO, Fortify Source, etc) and the techniques used to defeat them. Time permitting, the course will also cover other subjects in exploitation including kernel-land and Windows based exploitation.

Course Text(s)

Suggested Textbooks:

- Hacking: The Art of Exploitation, 2nd Edition by Jon Erickson
  - ISBN 978-1593271442
- The Shellcoder's Handbook: Discovering and Exploiting Security Holes, 2nd Edition by Chris Anley et al
  - ISBN 978-0470080238

Student Learning Outcomes

Upon successful completion of this course, students will:

1. Possess the skills necessary to carry out independent vulnerability research against binary applications.
2. Have an intimate understanding of executable formats, program control flow at the assembly level, and other low level concepts.
3. Understand classic and contemporary vulnerabilities and exploitation techniques.
4. Apply both source code auditing and binary reverse engineering to the vulnerability discovery process.
5. Be capable of exploiting vulnerabilities found in real world software as defined by MITRE's Critical Vulnerabilities and Exposures (CVE) system.

Course Assessment Measures

- **Labs:** Accompanying each lecture will be a lab that will solidify understanding of the topic of the lecture. Each lab will contain at least 3 problems, a C, B, and A level problem. There may occasionally be an A+ problem for extra credit.

- **Term Projects:** There will be two projects over the course of the semester. In these projects students will be exploiting a known vulnerability in a realistic piece of software as selected by the instructors. Students will be expected to analyze the vulnerability and produce a working exploit with a writeup detailing the vulnerability and how it can be leveraged to gain privileged information. Students may work in groups of up to two for projects.

See the Grading Criteria and Course Calender sections for further details.

Grading Criteria

Students will be able to calculate their standing at any time during the class by using the below percentages to calculate their grade.

- **Labs:** 60%, 10 labs equally weighted at 6%
    - Labs will typically consist of 3 problems, a C, B, and A problem with each problem harder than the last. Lab grading works as follows.
        - Students who complete only the C problem will receive a C
        - Students who complete both the C and B problems will receive a B
        - Students who complete all three lab problems will receive an A
        - There may be one or two A+ problems throughout the semester that can be used to increase the letter grade of any previous lab
    - If a student does not complete the first problem before the end of lab, they will receive a letter grade reduction for the assignment.
    - In order to receive full credit for a problem the student must submit their exploit code, the flag, and may be asked to explain their work upon submission to be checked off.
    - Labs problems will be introduced by the end of lecture. The first problem will be due in person **by the end** of the associated lab period. All other problems become homework, and are due **at the start of class, exactly one week after the associated lab period**.
    - Labs submitted late will receive a letter grade reduction and MUST be submitted no later than the class following their original due date, anything later will not be accepted.

- **Term Projects:** 40%, two projects equally weighted at 20%
    - Project specific grading breakdowns will be given when they are assigned.
    - Project checkpoints exist to keep you on pace with projects, failing to make checkpoints will result in a -5% reduction to the term project grade.
    - Projects submitted late will receive a -10% reduction per day late, and will not be accepted for credit after 5 days.

There will be servers dedicated to hosting the problems to be completed by students for both the lab and projects. The submitted exploits/solutions can be developed in any practical manner, but ultimately must work on the course servers to receive credit.

Grades and course progress will be made available to students throughout the semester. A grade can only be appealed within 5 days of the grade being made available to students.

Questions regarding a grade on an assignment should first be directed at the class TAs.

Course Calendar

Date	Title	Topics	Content
1/27	Syllabus and Review	Linux, C, x86	Lecture
1/30	Introduction to Reverse Engineering	Tools and the VM	Lecture
2/3	Extended Reverse Engineering	GDB & IDA	Lecture
2/6	Reverse Engineering Lab		Lab
2/10	Intro to Memory Corruption	ELF, the stack, calling conventions, buffer overflows	Lecture
2/13	Memory Corruption Lab		Lab
2/17	******* **NO CLASS** *******	**President's Day**	-
2/20	Shellcoding / Code Injection	Writing shellcode, developing scenario relevant payloads	Lecture
2/24	Shellcoding Lab		Lab
2/27	**Mid-term Project Assigned** Format String Vulnerabilities	Format strings, DTOR/GOT overwrites	Lecture
3/3	Format String Lab		Lab
3/6	******* **NO CLASS** *******	**ISTS12 @ RIT**	-
3/10	DEP and ROP	Data Execution Prevention, writing ROP, ret2libc	Lecture
3/13	ROP Lab		Lab
3/17	**Mid-term Project Checkpoint One Due** Secure Systems and Game Console Exploitation	OpenBSD, SELinux, GRSEC Game Console Exploitation	Lecture
3/20	Project One Lab		Lab
3/24	******* **SPRING BREAK** *******		-

3/27	******* **SPRING BREAK** *******		-
3/31	**Mid-term Project Due** Address Space Layout Randomization (ASLR)	Overview, info leaks, partial overwrites, ASLR closure	Lecture
4/3	ASLR Lab		Lab
4/7	Heap Exploitation	Heap structure and concepts, corruption, use after free	Lecture
4/10	Heap Exploitation		Lab
4/14	Stack Cookies & Misc Concepts	Bypassing stack cookies, signed/unsignedess issues, uninitialized data, etc	Lecture
4/17	Misc Concepts Lab		Lab
4/21	C++ Differences & Concepts	C++ basics, structures, vTables, Exceptions	Lecture
4/24	**Final Project Assigned** C++ Exploitation Lab		Lab
4/28	Kernel Exploitation	Basic kernel exploitation	Lecture
5/1	**Final Project Checkpoint One Due** Kernel Exploitation Lab		Lab
5/5	Final Project Lab		Lab
5/8	**Final Project Checkpoint Two Due** Exploitation on ARM, 64bit, Windows	Branching out, how exploitation differs on ARM, 64bit, and Windows	Lecture
5/12	Automation & The Future of Exploitation	Fuzzing, taint analysis, dynamic instrumentation, SMT/SAT solvers	Lecture
5/15	**Final Project Due**		

Attendance Policy

Although attendance for lecture is not mandatory, it will be *exceptionally* difficult to keep up without attending due to the subject matter and progression of the course. **Attendance for labs is required** as the first problem of each lab set must be turned in in person during lab.

See Grading Criteria for more details

Academic Integrity

Student-teacher relationships are built on trust. For example, students must trust that teachers have made appropriate decisions about the structure and content of the courses they teach, and teachers must trust that the assignments that students turn in are their own. Acts that violate this trust undermine the educational process. The Rensselaer Handbook of Student Rights and Responsibilities defines various forms of Academic Dishonesty and you should make yourself familiar with these. In this class, all assignments that are turned in for a grade must represent the student's own work. In cases where help was received, or teamwork was allowed, a notation on the assignment should indicate your collaboration.

Submission of any assignment that is in violation of this policy will result in a penalty of **a zero for the assignment** for all parties involved. **Repeated offenses will result in a failing grade for the course.**

If you have any question concerning this policy before submitting an assignment, please ask for clarification.

Other Course-Specific Information

The labs and interactive exercises used in lecture will be hosted in a variety of different series on an internal 'Wargame' server meant purely for teaching security concepts in a safe and educational environment. Students enrolled in the course will have SSH access to the server and are expected to use it as an aide to the learning process. Attacking parts of the server infrastructure beyond the scope of what is assigned or attempts to disrupt services for the course or other students is **not allowed** and will be considered a violation of academic integrity.

Due to the experimental nature of the course and assignments being offered, the schedule, pacing, and other elements of the course may be modified depending on the rate at which students are progressing. Any changes made to the curriculum will be clearly defined and communicated to the students as well as being documented accordingly.