Systems Security and Binary Code Analysis

Lecture 1: Course Overview

Zhiqiang Lin

Department of Computer Science
University of Texas at Dallas
Outline

1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Reverse Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
Outline

1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revere Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
The **goal** of this class is to understand the state-of-the-art of techniques in both offense and defense in computer systems, and know

- How to perform the reverse engineering of x86 binary code with program analysis (e.g., static and dynamic analysis, symbolic execution)
- How to identify the vulnerabilities (e.g., buffer overflow, integer overflow) in binary and exploit them by developing robust shell code (e.g., ROP)
- How to design defenses using compiler extension, OS kernel, virtualization (i.e., hypervisor), and hardware support (e.g., SGX)
This course is taught in a **regular-class** style. Each student will be expected to
This course is taught in a regular-class style. Each student will be expected to

1. **Attend** all the lectures
2. **Finish** the assigned readings
3. **Participate** a capture the flag (CTF) hacking contest
4. **Five** hands on assignments and one research project.
5. No midterm, and but there will be a final exam
Course History

Prior Courses

1. CS/SE 6V81: Advanced Digital Forensics and Data Reverse Engineering (Fall 2011)
2. CS/SE 6301: Systems Security and Binary Code Analysis (Spring 2012, Fall 2013)
3. CS/SE 4v95: Offense based cyber security (Fall 2014)
Feedbacks – From Students

Positives

- This course was great and gave us insight into a lot of system security
- Overall the course was fun to learn
- Overall I liked the course and got to know many important things in system security
- ...

Improvements

- Engage with more hands-on project
- The project needs to have more checkpoints
- Need more practical sessions
Feedbacks – From Students

Positives

- This course was great and gave us insight into a lot of system security
- Overall the course was fun to learn
- Overall I liked the course and got to know many important things in system security
- ...

Improvements

- Engage with more hands-on project
- The project needs to have more checkpoints
- Need more practical sessions
- ...

1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revert Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
1 Background

2 Course Content
   - Basic Computer Systems Knowledge
     - Binary Code Analysis / Revere Engineering
     - Vulnerability, Exploits, and Defenses
     - Virtualization Based Systems Security
     - Hardware Supported Systems Security

3 Homework and Project

4 Course Policy
Review of Basic System Knowledge

Topics

1. Low level code (x86 disassembly)
2. Programming and Program Execution
   - Stack, Heap, Global
   - Control flow
3. Operating Systems
   - Process Creation/Execution/Exit
   - Virtual Memory
   - Networking, I/O
Outline

1 Background

2 Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revere Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3 Homework and Project

4 Course Policy
Binary Code Analysis: Tools

Topics

1. Static analysis
   - IDA Pro, BinNav
   - BAP

2. Dynamic analysis
   - PIN
   - QEMU

3. Symbolic execution
   - FuzzBall, Fuzzgrind
   - S2E
Binary Code Analysis: Techniques

Topics

1. Data Flow Analysis
   - Data Dependency
   - Taint Analysis
   - Point-to analysis (alias analysis)

2. Control Flow Analysis
   - Control flow graph
   - Call graph
   - Control dependency
   - Strong control dependency
1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Reverse Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
Vulnerability

Topics

1. Control flow hijacking
   - Buffer Overflow
   - Integer Overflow
   - Format String

2. Non control data attack

3. Web/Android App vulnerability is not covered
Exploits

Topics

1. Shell Code
2. Code Injection
3. Return-into-libc
4. Reliable Shell Code
   - Address leakage attack
   - Blind-Hacking
   - Return-oriented programming
Defenses

Topics

1. Compiler extension (canaries)
2. Safe Libraries (libsafe, libsafeplus)
3. OS support: ASLR, DEP, Re-Randomization
4. Control flow integrity (CFI)
   - Strict-CFI
   - Weak-CFI
1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revers Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
Virtual Machine Introspection

Topics

1. Challenges
2. Approaches
3. Applications
   - Rootkit detection
   - Malware analysis
   - Intrusion detection
   - Guest OS Management
Outline

1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Reverse Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
Trusted Computing w/ SGX

Topics

1. Trusted computing w/ hypervisor, hardware support
2. Enclave
3. SGX Instructions
4. SGX Programming
5. SGX Applications
   - Outsourced computation protection
   - Client side software protection
Outline

1. Background

2. Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revere Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3. Homework and Project

4. Course Policy
Homeworks

Readings

- You will be asked to read the assigned book chapters or other reading materials.
- Questions in the project assignment and exams will be asked based on these materials.
- Please be aware they may not be covered in the lectures.
Course Assignments/Projects

Course Assignments

1. Using BinNav for static program analysis
2. Using PIN/QEMU for dynamic program analysis
3. Analyzing software vulnerabilities and developing (ROP) exploit to compromising vulnerable software
4. Using Crash/Volatility to analyze memory dumps
5. Developing SGX programs

Research Project

In addition to the five written assignments, you also need to finish a research project. Topics need to be discussed with the instructor offline.
Outline

1 Background

2 Course Content
   - Basic Computer Systems Knowledge
   - Binary Code Analysis / Revere Engineering
   - Vulnerability, Exploits, and Defenses
   - Virtualization Based Systems Security
   - Hardware Supported Systems Security

3 Homework and Project

4 Course Policy
Grading Policy

- 30% Course Assignment
- 30% Research Project
- 30% Final exam
- 10% Course Participation
Grading Policy

- 30% Course Assignment
- 30% Research Project
- 30% Final exam
- 10% Course Participation

Exceptional work will be rewarded appropriately (e.g., guaranteed for A+)
Prerequisites

System Skill Set

This is a highly technical class. We expect students to have a strong technical background before taking this course. Students who have not taken a security class before or whom are otherwise unfamiliar with computer security will likely not be able to complete this class. Specifically, students should satisfy at least three of the following:

1. Assembly code (Intel X86 preferred)
2. Knowledge of Computer Security basics
3. Proficiency in programing development (gcc/gdb) (CS 3376 C/C++ Programming in a UNIX Environment)
4. Proficiency in a scripting language (python preferably)
5. Familiarity with operating system kernel/internals (windows or linux) (CS 4348 Operating Systems Concepts)
6. Comfortable with command line operation of Windows AND Linux
Course Prerequisites

UTD: Must take any **three** of the following class:

1. CS 3340 Computer Architecture
2. CS 3376 C/C++ Programming in a UNIX Environment
3. CS 4348 Operating Systems Concepts
4. CS 4393 Computer and Network Security
5. CS 4394 Implementation of Modern Operating Systems

Otherwise, please drop (you can feel free to audit if you want)
Required textbooks


Other reference materials

1. Intel Software Guard Extension References [SGX]
2. Recent academic papers

They all can be read online from our campus library link.
Leave vs. Stay

```plaintext
char code[] = "\xb0\x01\x31\xdb\xcd\x80";
```

```
08048080 <_start>:
8048080:   b0 01   mov $0x1,%al
8048082:   31 db   xor %ebx,%ebx
8048084:   cd 80   int $0x80
```
char code[] = "\\xb0\\x01\\x31\\xdb\\xcd\\x80";

08048080 <_start>:
8048080:  b0 01 mov $0x1,%al
8048082:  31 db xor %ebx,%ebx
8048084:  cd 80 int $0x80

char code[] = "\\x31\\xc0\\xb0\\x46\\x31\\xdb\\x31\\xc9\\xcd\\x80\\xeb"\\n    "\\x16\\x5b\\x31\\xc0\\x88\\x43\\x07\\x89\\x5b\\x08\\x89"\\n    "\\x43\\x0c\\xb0\\x0b\\x8d\\x4b\\x08\\x8d\\x53\\x0c\\xcd"\\n    "\\x80\\xe8\\xe5\\xff\\xff\\xff\\x2f\\x62\\x69\\x6e\\x2f"\\n    "\\x73\\x68\\x58\\x41\\x41\\x41\\x41\\x42\\x42\\x42\\x42";
Other Policy

Late Policy

No late submission. The points will be automatically decreased 1 points per week-day until all points in that assignment become 0.
Other Policy

Late Policy
No late submission. The points will be automatically decreased 1 points per week-day until all points in that assignment become 0.

Collaboration Policy
Students are encouraged to collaborate, particularly on the discussion on the homework and course projects. However, each individual student must finish the project by him/her-self.
Late Policy
No late submission. The points will be automatically decreased 1 point per week-day until all points in that assignment become 0.

Collaboration Policy
Students are encouraged to collaborate, particularly on the discussion on the homework and course projects. However, each individual student must finish the project by him/her-self.

Cheating Policy
Strictly follow the university policy on cheating and plagiarism.
Late Policy
No late submission. The points will be automatically decreased 1 point per week-day until all points in that assignment become 0.

Collaboration Policy
Students are encouraged to collaborate, particularly on the discussion on the homework and course projects. However, each individual student must finish the project by him/her-self.

Cheating Policy
Strictly follow the university policy on cheating and plagiarism
Who should take

1. Security analyst
2. Vulnerability researchers
3. Incident responders
4. Penetration testers
5. Security professionals
6. Forensics professionals
Who should not take

1. Students who have not taken a security class before
2. Students who are afraid of low level stuff (e.g., machine code, reverse engineering, shell code).
3. Lazy people who do not do the assigned readings or homeworks
4. If programming intimidates you, then reverse engineering is probably not for you.

Again, tests and assignments will cover the reading materials that are not mentioned in the lectures.