CS 6V81-05
System Security and Malicious Code Analysis –
Course Overview

Zhiqiang Lin

Department of Computer Science
University of Texas at Dallas

January 18th, 2012
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Outline

1 Overview
   - Course Goals
   - Course Style

2 Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3 Course Project

4 Course Policy

5 Homework
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Course Goals

The **general goal** is to understand the state-of-the-art

---

**Offense/Attack**
- Find memory vulnerability
- Develop exploits
- Create malware

---

**Defense/Protection**
- Find vulnerability
- Stop exploits
- Analyze malware
The **general goal** is to understand the state-of-the-art.

**Offense/Attack**
- Find memory vulnerability
- Develop exploits
- Create malware

**Defense/Protection**
- Find vulnerability
- Stop exploits
- Analyze malware
- Ensure availability
Mission

Break the system to gain resources with respect to **confidentiality** (something you are not supposed to access) and **integrity** (something you are not supposed to modify)

Techniques

- Find memory vulnerability
  - Memory vulnerability
    - Buffer overflow, integer overflow, and format string
    - Logic vulnerability or new web vulnerability (SQL injection)
- Develop exploits
  - Memory exploits, shell code, ROP, heapspray
- Create malware (Obfuscation/Packing)
  - Packing (encryption)
  - Translation, virtualization
Defense/Protection

Mission
Protect information and system resources with respect to confidentiality, integrity, and availability (defending such as Denial of service attack)

Techniques
- Find vulnerability (Penetration Testing)
- Stop exploits
  - Architecture, hardware
  - Operating System, Loader, Linker
  - Compiler
- Analyze malware (Reverse Engineering)
  - Unpacking (decryption)
  - De-transformation, De-virtualization
- Ensure availability, preventing DoS
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
## Course Style

This course is taught in both a **seminar** and a **regular-course** style. Each student will be expected to:
Course Style

This course is taught in both a seminar and a regular-course style. Each student will be expected to

1. **Read** one of the most recent papers given by the instructor
   *Thoroughly*
Course Style

This course is taught in both a seminar and a regular-course style. Each student will be expected to:

1. **Read** one of the most recent papers given by the instructor
   *Thoroughly*

2. **Present** the key techniques in the paper within 12 minutes, with additional 3 minutes question
This course is taught in both a seminar and a regular-course style. Each student will be expected to

1. **Read** one of the most recent papers given by the instructor

   *Thoroughly*

2. **Present** the key techniques in the paper within 12 minutes, with additional 3 minutes question

3. **Attend** all the lectures
This course is taught in both a seminar and a regular-course style. Each student will be expected to

1. **Read** one of the most recent papers given by the instructor *Thoroughly*

2. **Present** the key techniques in the paper within 12 minutes, with additional 3 minutes question

3. **Attend** all the lectures

4. **Scribe** one lecture (write notes)
Course Style

This course is taught in both a seminar and a regular-course style. Each student will be expected to

1. **Read** one of the most recent papers given by the instructor *Thoroughly*

2. **Present** the key techniques in the paper within 12 minutes, with additional 3 minutes question

3. **Attend** all the lectures

4. **Scribe** one lecture (write notes)

5. **Perform**
   - An individual research project, or as
   - A team for an engineering project (5K LOC), with 3 members
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Outline

1 Overview
   - Course Goals
   - Course Style

2 Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3 Course Project

4 Course Policy

5 Homework
Understanding OS Kernels

Topics

1. Process Management
   - Creation/Running/Exit
   - Process address space
   - Context switch

2. Virtual Memory
   - Paging

3. File System and Disk Data Management
   - EXT2/EXT3, NTFS
   - Proc
Understanding Program Runtime

Topics

1. Unveiling Program Execution
   - Stack, Heap, Global
   - Control flow

2. System Loader
   - ld-linux.so, ld_preload
   - DLL Injection

3. Dynamic Linker
   - PLT/GOT
   - Online-patching
Understanding Program Analysis

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

3. Path-(in)sensitive

4. Context-(in)sensitive
Outline

1 Overview
   • Course Goals
   • Course Style

2 Course Content
   • System and Software Security Foundations
   • Vulnerability Analysis and Exploit Generation
   • System Defense
   • Reverse Engineering

3 Course Project

4 Course Policy

5 Homework
Vulnerability

Topics

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

2. Data manipulation
   - SQL Injection, XSS (web application)
   - Logic-vulnerability (shopping free attack)
   - Browser-logic vulnerability (GUI interface)
Exploits

Topics

1. Shell Code
2. Code Injection
3. Return-into-libc
4. Reliable Shell Code
   - HeapSpray (ASLR)
   - Return-oriented programming
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Architecture, OS Perspective

**Topics**

1. Address Space Randomization (ASR), DEP, NX-bit
2. Instruction Set Randomization (ISR)
3. Data Randomization (DR)
4. Operating System Interface Randomization, RandSys
5. N-Variant System, Reverse Stack Execution
6. System Call Interposition
7. Library Extension (Libsafe/LibsafePlus/LibsafeXP)
8. Virtual Machine Introspection
Compilation Extension, Code Transformation, Runtime Verification

### Topics

1. Bounds Checking, Type Checking
2. Diehard (heap protection), exTerminator
3. Binary Rewriting, SFI/XFI/CFI/DFI
4. Program Shepherd
5. Sandboxing, NativeClient
6. Taint Checking, Blocking Bad Input
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Compilation Extension, Code Transformation, Runtime Verification

Topics

1. Binary Code Analysis (Reverse Engineering)
2. Dynamic Binary Analysis
   - Run-time Instrumentation
   - Automatic
3. Symbolic Execution
4. Malware Packing/Unpacking
   - Binary Code Transformation
   - Virtualization
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Course Projects

Sample Projects

- Implement a dynamic binary code analysis plug-in, using a most recent PIN, with a data flow analysis (taint analysis at byte level) capability
- Roughly code size 3K LOC

Apply your plug-in for

- Vulnerability analysis
- Penetration testing
- Reverse engineering
Outline

1 Overview
   - Course Goals
   - Course Style

2 Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3 Course Project

4 Course Policy

5 Homework
Grading Policy

- 20% In-Class Presentations
Grading Policy

- 20% In-Class Presentations
- 10% Class participation
Grading Policy

- 20% In-Class Presentations
- 10% Class participation
- 20% Scribe a lecture (write a note)
Grading Policy

- 20% In-Class Presentations
- 10% Class participation
- 20% Scribe a lecture (write a note)
- 50% Class Project
Grading Policy

- 20% In-Class Presentations
- 10% Class participation
- 20% Scribe a lecture (write a note)
- 50% Class Project
- **Exceptional work** will be rewarded appropriately
Prerequisites

System Skill Set

Solid programming/development skills (Assembly, C, C++, Unix). "Operating System", "Compilers", and "Computer Security", are the least prerequisites for this class. I am training “academic hackers”

UTD

- CS 3340 Computer Architecture
- CS 3376 C/C++ Programming in a UNIX Environment
- CS 4348 Operating Systems Concepts
- CS 4393 Computer and Network Security
- CS 4394 Implementation of Modern Operating Systems
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
Leave vs. Stay

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"
  "\x16\x5b\x31\xc0\x88\x43\x07\x89\x5b\x08\x89"
  "\x43\x0c\xb0\x0b\x8d\x4b\x08\x8d\x53\xc0\xcd"
  "\x80\xe8\xe5\xff\xff\xff\xff\x2f\x62\x69\x6e\x2f"
  "\x73\x68\x58\x41\x41\x41\x41\x42\x42\x42\x42";
Other Policy

Late Policy

No late submission
Other Policy

Late Policy
No late submission

Collaboration Policy
Encouraged, but limit the team member to at most three students.
Late Policy
No late submission

Collaboration Policy
Encouraged, but limit the team member to at most three students.

Cheating Policy
Strictly follow the university policy on cheating and plagiarism
Other Policy

Late Policy
No late submission

Collaboration Policy
Encouraged, but limit the team member to at most three students.

Cheating Policy
Strictly follow the university policy on cheating and plagiarism
Outline

1. Overview
   - Course Goals
   - Course Style

2. Course Content
   - System and Software Security Foundations
   - Vulnerability Analysis and Exploit Generation
   - System Defense
   - Reverse Engineering

3. Course Project

4. Course Policy

5. Homework
Homework-0: Due next Monday

**Paper Presentation Sign-Up**
Selecting the paper (date implicitly selected) to present

**Scribe Sign-Up**
Selecting the date for to-be-scribed lecture. Two students are allowed to scribe the same lecture if there is no slot.

**Engineering Project**
- Forming your team with at most 3 members
- Starting to get familiar with PIN

**Research Project**
- Decide the research project
- Talk to the instructor