Course Description

In this course, students will be introduced to the foundations of software security. We will be exploring different classes of software vulnerabilities, analyzing the fundamental problems behind these vulnerabilities, and studying the methods and techniques to discover, exploit, prevent and mitigate these vulnerabilities. Topics of interest include buffer overflow, integer overflow, type confusion, use-after-free, etc. Throughout the course, we take a defense-in-depth mentality and see how systems can be protected. Students are expected to have a solid understanding of assembly language, C/C++ and operating system. Pre-req: CSE 361

Textbook

There is no textbook for the class. We will use research papers for some of the topics. However, the following references can be helpful.

- Hacking the art of exploitation
Grading

There is no exam in this class, evaluation will be based on class participation and final projects.

- Research Discussion: 40%
- Projects: 60%

Schedule

Slides, Lab Assignments and QA are in the WUSTL backboard system.
We will cover topics from buffer overflow to automatic vulnerability discovery in various platforms.

**System Knowledge**
- Class Intro and Principle of Software Design
- Low level details - Application Memory Layout
- Low level details - User vs Kernel
- Low level details - instrumentation tools
- Low level details - vulnerability analysis tools

**Software Vulnerabilities**
- Stack overflow with studio
- Stack overflow Discovery and prevention
- Heap overflow with Studio
- Heap overflow discovery and prevention
- Return to lib C with studio
- Return to lib C discovery and prevention
Software Security

- Format string vulnerability discovery and prevention
- Command injection Vulnerability with ShellShock Studio
- Command injection discovery and prevention
- Integer overflow with studio
- Integer overflow discovery and prevention
- Type confusion with studio
- Type confusion discovery and prevention
- Concurrency bug with Dirty COW Studio
- Concurrency bug automatic discovery and prevention
- Side channel attack with studio
- Side channel discovery and prevention

Trustworthy Computing

- ARM TrustZone
- TrustZone attack and defense discussion
- Intel SGX
- SGX attack and defense discussion

Research Paper

Detecting Format String Vulnerabilities with Type Qualifiers

TBD

Projects
Software Security

The class project can be original research or survey on an existing topic in network security. Class project report should follow the IEEE conference template. The length of survey should sufficient to provide insight into a topic (It often requires more than 6 pages for individual projects). The class project can also be improvement on an existing security tool. The development should be source controlled using tools, such as git.

Students are expected to spend at least six to eight hours on the class labs and projects every week. Some of the ideas for projects are listed below:

- Building an analysis module that will find any of vulnerable patterns we discuss in class
- Build improve current defense against one of the vulnerabilities
- YOUR IDEA

Ethics

With greater power, comes greater responsibility. In this course, we will be learning about and exploring some vulnerabilities that could be used to attack systems. Students are expected to behave responsibly and ethically. You may not attack any system prior approval of the site owners, and may not use anything you learn in this class to disrupt services or harm others. If you have any doubts about whether or not something you want to do is ethical and legal, you should check with the course instructor.