Course Title

CSC 595 Topics in Computer Science: Program Analysis

Course Description

Software debugging and testing are critical aspects of software development. Nowadays many techniques have been developed to automate or aid the process of software debugging and testing. As the core of these techniques, program analysis automatically analyzes the code of a target program to discover properties of the program, such as correctness and robustness. Tremendous effort has been invested by both the industry and academia to develop program analysis techniques for finding and addressing software bugs, which are frequently exploited by real-world attacks. Major software vendors such as Google, Microsoft, Facebook (Meta), and Adobe, have incorporated intensive use of program analysis tools as a critical part of their software development life cycle.

The purpose of the course is to introduce the concepts and techniques of program analysis, particularly for addressing software debugging and testing problems, such as finding and addressing software bugs. Students will also gain hands-on experience applying program analysis to automatically test software and address software bugs in complex real-world programs.

Prerequisites

CSC 373/406 or consent of the instructor

Instructor

Professor: Zhen Huang
Office: Room 735, CDM Center
Email: zhen.huang@depaul.edu
Phone: (312)362-8239
Homepage: `http://facsrv.cs.depaul.edu/~zhuang28/`

Office Hours

My office hours are held via zoom meetings in two sessions:
11:00am – 12:00pm on Tuesdays
10:00am – 12:00pm on Thursdays
The link to the zoom meetings is posted on the course website.

Textbooks

Anders Mller and Michael I. Schwartzbach (2022). *Static Program Analysis*

Learning Outcomes

After taking the course, students will be able to

- formulate a software development problem in a way so that it can be solved using program analysis
- implement and use program analysis techniques, such as control flow analysis and data flow analysis, to address software debugging and testing problems
- use and extend state-of-art program analysis tools to solve software development problems

Weekly Schedule

Week 1	Introduction of program analysis. Review of x86-64 assembly language.
Week 2	Control flow analysis. Basic blocks. Dominators and post-dominators.
Week 3	Control dependency graph. Data flow analysis. Reaching definition analysis.
Week 4	Liveness analysis. Data flow framework.
Week 5	Angr binary analysis framework. Solver engine. Program state. VEX intermediate representations.
Week 6	Angr's control flow graph and data flow graph. Developing control dependency graph using Angr.
Week 7	Common software vulnerabilities. Vulnerability repair and mitigation.
Week 8	Security Workaround for Rapid Response (SWRR). Identifying error handling code with program analysis.
Week 9	Final exam. Binary analysis using RVM.
Week 10	Identifying error handling code in binary. Generating and instrumenting SWRRs using RVM and Talos.
Week 11	Final project presentations.

Assessment

Course assessments consist of homework assignments, a project, and a final take-home exam. The course grade will be computed as follows:

- Homework Assignments: 30%
- Final exam: 25%
- Project: 30%
- Final Presentations: 15%