

## **ECE568 Advanced Microprocessor Architecture and Design**

### **Spring 2007 - Architecture Support for Software Protection**

**Instructor:**

Gyungho Lee

Dept. of Electrical and Computer Engineering

University of Illinois at Chicago

email: [ghlee@ece.uic.edu](mailto:ghlee@ece.uic.edu)web: <http://arch.ece.uic.edu/>**Course Objective:**

To gain knowledge on the fundamentals of secure computing and to study the issues and the recent trends in software protection for trusted computing from processor architecture perspective.

**Course Description:**

Security issues in computing have been extensively studied last forty years. Numerous, both theoretical and experimental, research works exist on Trusted Computing Base (TCB) that is a set of hardware and software mechanisms which guarantees that regardless of any program executed on the system, security will not be violated. TCB proposals have been based on access control or data dissemination control using the concept of "reference monitor" and algebraic manipulation of multi-level trust values for secure information flow. However, the early TCB proposals have not been adopted in mainstream computing because of high performance overhead and programming restrictions they impose. Instead, the advent of desktop computing and the Internet has caused wide deployment of unsecured computing with "flat" memory model to the public. With the flat memory model and lack of stringent type checking, the programs are prone to errors causing vulnerability in security. With ever increasing instances of exploiting security holes in software, many approaches have recently been proposed to make software more secure. One common aspect of all these approaches is that they are in essence an effort to retrofit the current systems with a simplified version of old trusted computing proposals.

This course considers software protection for trusted computing from the computer architecture perspective:

1. Review of "old" theoretical models for trusted computing and information flow and their implementation cases: (4 to 6 lectures)
2. Study of software protection issues and protection schemes: (6 or 8 lectures)
3. Introduction to instruction validation at run-time: (3 or 4 lectures)

**Web:** All the lecture notes and reference papers including reading assignments will be posted at <http://arch.ece.uic.edu/~ece568/> Check the website regularly.

## Course Outline

1. Introduction
  - a. Issues
  - b. Informal Design Principles
2. Trust/Information-flow model
3. Cryptography
  - a. Stream/Block Cipher
  - b. Symmetric/Asymmetric Encryption
  - c. (AES accelerator: later by Lan Li)
4. Architectures for Trusted Computing
  - a. Original – Data Mark Machine
  - b. Example proposals
5. Control-Flow Altering Attacks
  - a. Buffer Overflow
  - b. Others – Format String, Racing, mimicry, etc.
6. Intrusion Detection - Securing Program Behavior
  - a. Pattern/Case Based
  - b. Model Based
7. Run Time Validation of Program Behavior
  - a. Validating Instruction Instance – control flow and data flow
  - b. PC-encoding
  - c. Enhancing Branch Prediction (later by Yixin Shi)

**Note** Some of the materials regarding run-time validation of program behavior are confidential and the property of the instructor's.

## Grading

will be based on two aspects:

- Presentation of paper reading – students will be asked to read a few recent papers, summarize them, and present the summary in class.
- Term project – each student (or a group of two students) is expected to work on an aspect of software protection as the term project for the class: the project report, in the form of IEEE conference proceedings (5 to 6 pages in double column in 10pt.), is due by the end of the semester. One page (about 200 ~ 300 words) for the term project topic is due by Feb. 26 (via email)