CMSC 838g
Software Security

Michael Hicks
CS, MC2 @ University of Maryland
1/28/2014

When is a program secure?

- When it does **exactly** what it should
  - All the right things, none of the wrong ones
  - How do we know which is which?
    - Someone tells us (do we trust them?)
    - We decide ourselves (how do we check?)
    - We write the code ourselves (how much of the software you use have you written?)

- **Perfect security does not exist**
  - Must trade off performance, cost, usability, functionality
  - When is software “secure enough”?

Software Security

- Conventional security: program is a black box
  - Encryption
  - Firewalls
  - System calls/privileged mode
  - OS-provided process isolation
- Major limitations — doesn't address:
  - Downloaded and mobile code
  - Buffer overruns and other software flaws
  - Application-level security policies
  - System-wide security guarantees
- **Open the black box: consider the software**

CMSC 838: Software Security

- In this class we will look at methods for improving software security
  - Novel programming languages for ensuring “security by construction”
  - Program analysis techniques for finding flaws or proving their absence in existing code
    - Means to (formally) state what security is
    - Testing, dynamic analysis, and static analysis to check that it holds
  - Hardware/Systems elements to help
- We will study software both theoretically (formalism and proof) and empirically (by building/measuring things)

Syllabus

- Course content
  - Largely based on paper reading, both classics and more recent results
- Graded activities
  - on-line discussion and off-line paper reviews (25%),
  - attendance and class participation (15%),
  - final project writeup and presentation (30%),
  - final exam (30%)
    - with final proj, constitutes COMP credit
- Prereqs:
  - None, but CMSC 631 will help. Will provide lecture material to catch up those without this background.

Paper reading

- Read the paper in several passes (1-3 hours total):
  - **Quick skim** for the main ideas: read the intro, then scan over the section headings for the rest, to see what it's about. Then read the conclusions
  - **Revisit the details**, trying to understand, roughly, the approach they are taking and how the evaluate their results
  - **Deep dive** into main technical results (if needed)
- Let your reading be
  - **Critical**: Are they living up to what they claimed?
  - **Comparative**: Relation of ideas to other work?
  - **Creative**: Can you do it better? Apply it elsewhere?

Before class: Commenting on papers

- Each paper: **post** an *insightful* **comment** or **question**
  - **By 9am the morning of the class**
  - That is distinct from prior comments/questions
  - Good questions can point out things not made clear in the paper, missing connections to other work, etc.
- Grading comments (I will justify my score):
  - 0, for showing no evidence of having read the paper
  - 1, for minimal effort or a not-insightful summary
  - 2, for actively insightful
- Aim for 1-2 paragraphs, but go longer or shorter if you think it's warranted
  - Save some (more) questions for class

During class: Discussion

- I will take attendance
- I will spend 5-10 minutes summarizing each paper
- The rest of class is open discussion
  - Goal is to understand the paper better: use your classmates' expertise/insights to help!
  - Be prepared with topics to discuss (e.g., from reading blog, and your own ideas)
  - I may call on you to answer pointed questions if the discussion dries up
- Keep in mind topics you are particularly interested in, and write down ideas for further research

Scribes

- During each class we will have a scribe take notes about the discussion
  - I will assign a scribe just before class
- Scribe reports should have the following format:
  - **Summary**: one or two paragraphs summarizing the papers
  - **Questions**: questions about parts of paper that were not clear, and their resolution (e.g., on technology, approach, etc.)
  - **Criticisms**: problems with the algorithm, evaluation, applications, claims, etc.
  - **Ideas**: suggestions for improvement, new applications, and other work

Project

- A substantial effort to carry out
  - Novel research
  - A meaningful implementation (e.g., of an existing approach)
  - A comprehensive, well-done survey
- By yourself or with a partner
- Proposal
  - With goals, approach, timeline
  - Draft, basics due the week before Spring Break
  - Will provide feedback
  - Final version due the Friday after Spring Break

Final Exam

- Comprehensive coverage of the papers we read
- Likely will be a take-home exam