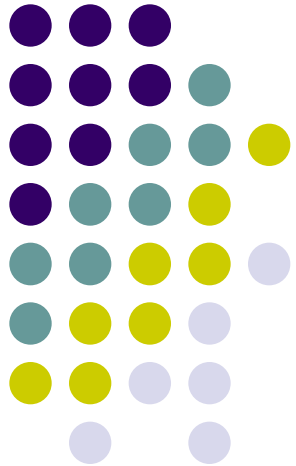
IS 2620

Developing Secure Systems

Introduction

Jan 8, 2013

James Joshi,
Associate Professor

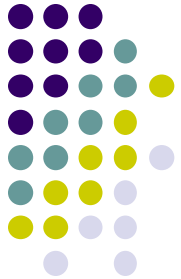


Contact

- James Joshi
- 706A, IS Building
- Phone: 412-624-9982
- E-mail: jjoshi@mail.sis.pitt.edu

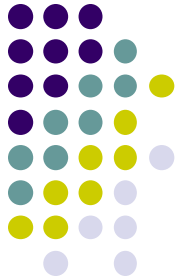
- Web: http://www.sis.pitt.edu/~jjoshi/courses/IS2620/Spring13/
  Office Hours: By appointments
- GSA: will be announced later

Course Objectives



- To learn about how to design/implement secure and high assurance information systems
  - Understand and analyze code for vulnerabilities
  - Secure programming (e.g., C, C++, Java)
- Understand the principles and practice towards designing secure information systems
  - Life cycle models/ security engineering principles
  - Usability issues
- To learn about the tools and techniques towards assurance (validation/verification/testing)
  - Use of tools to detect coding/design flaws;
  - architectural risk analysis

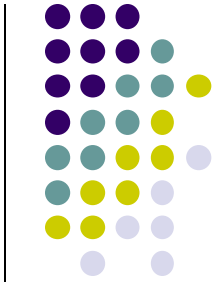
Course Coverage



- Secure programming
  - Coding practices and guidelines
    - Code analysis;
      - Buffer overflows
      - Input validation
      - Cross-site scripting Mobile Code

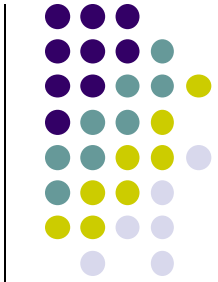
      Race conditions
      SQL injection
      Safe Languages
- Secure software development process
  - Security Engineering/Lifecycle models
    - E.g. Capability Maturity Models and Extensions
    - Building security In
  - Secure Design/Implementation Principles
    - Systems / software &Formal methods and testing
      - UMLSec, Model Checking (code, protocols)
- Miscellaneous issues (recent papers/articles)

Pre-requisite



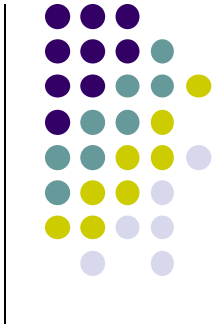
- IS 2150/TEL 2810 Introduction to Computer Security
  - OR some background in security
- Following courses are preferred but not required:
  - IS 2170/TEL 2820 Cryptography; TEL 2821 Network Security
  - IS 2511 or 2540
- Talk to me if you are not sure of the background

Course References



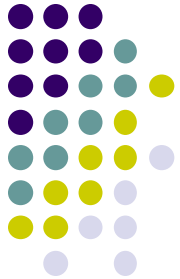
- Building Secure Software: How to avoid the Security Problems the Right Way, John Viega, Gary McGraw, Addison-Wesley, 2002
- Enterprise Java Security: Building Secure J2EE Applications, Marco Pistoia, Nataraj Nagaratnam, Larry Koved, Anthony Nadalin, Addition-Wesley, 2004
- Secure Systems Development with UML, Jan Jurjens, Springer-Verlag, 2005.
- Securing Web Services with WS-Security: Demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption – Jothy Rosenberg, David Remy, 2004, Sams Publishing, 2004.

Course References



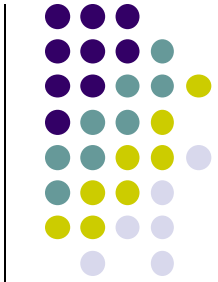
- High Assurance Design: Architecting Secure and Reliable Enterprise Applications, Clifford J. Berg, Addison-Wesley, 2006.
- Core Security Patterns: Best Practices and Strategies for J2EE?, Web Services, and Identity Management, Christopher Steel, Ramesh Nagappan, Ray Lai; Prentice-Hall
- How to Break Software Security - James Whittaker, Herbert Thompson, Addition Wesley, 2003
- Secure Coding in C and C++, Robert C. Seacord, Addition-wesley, 2006
- Computer Security: Art and Science by Matt Bishop (ISBN: 0-201-44099-7), Addison-wesley 2003.
- Papers; MSDN, US-CERT

Grading (Tentative)



- Assignments/Presentation/Exam: 60-**70**%
  - Read/Review and/or present research papers or articles
  - Assignments and lab exercises
  - One exam (15% - 20%)
- Project : 40-**30**%
  - Development-oriented project (e.g. Creating Secure Social Network; Secure Mobile Apps, etc.)
  - Research paper for conference
  - Team oriented and in some cases in collaboration with PhD students
  - Start early on

Course Policy



- Your work MUST be your own
  - Zero tolerance for cheating/plagiarism
  - You get an F for the course if you cheat in anything however small – NO DISCUSSION
  - Discussing the problem is encouraged
- Homework
  - Penalty for late assignments (15% each day)
  - Ensure clarity in your answers – no credit will be given for vague answers
  - Homework is primarily the GSA's responsibility
- Check webpage for everything!
  - You are responsible for checking the webpage for updates