CISC 327
Software Quality Assurance

Lecture “review3”
Review for Mini-Exam #3
Likely topics/questions on mini-exam #3

• From Lecture 17:
  – Mutation testing
    • Is mutation testing systematic?
    • What is the system (if there is one)?
    • How is mutation testing different from other white-box methods?
Likely topics/questions on mini-exam #3

• From Lecture 18:
  – Maintenance and continuous testing methods
    • Corrective, perfective, adaptive maintenance
    • The relevant exam question will involve Bogosys
Likely topics/questions on mini-exam #3

• From Lectures 18–19 (Maintenance, continuous testing, regression testing)
  – Corrective, perfective, adaptive maintenance
  – EVIL TIME from Lecture 18 won’t be on the exam
  – Regression testing
    • Know the 3 kinds of tests in a regression test suite
    • Know why some regression tests get “retired”
Likely topics/questions on mini-exam #3

• From Lecture 19-1:
  – general idea of backdoors, including Thompson’s compiler backdoor
  – know some of the many password attacks, e.g.: “man-in-the-middle”; guessing common passwords; timing attacks
  – buffer overruns:
    • what are the necessary elements of an attack?
      – overflowing a buffer
      – knowing or guessing the buffer’s location in memory, so that the return address can be overwritten with a pointer to the “payload”
    • why does address space layout randomization help?
Likely topics/questions on mini-exam #3

• From Lecture 19-1:
  – Morris worm: depended on buffer overruns, so it needed to know where buffers would be stored in memory
    • requires a relatively **homogeneous** network
  – the worm also exploited the fact that `finger` did not follow the Principle of Least Privilege
Likely topics/questions on mini-exam #3

• From Lecture 19-1:
  – early Macintosh and macro viruses:
    • extremely virulent, thanks to the willingness of the early Mac OS and early Word, Excel, etc. to automatically run whatever code they found
Likely topics/questions on mini-exam #3

• From Lecture 19-1:
  – Heartbleed and information leaks
    • not a buffer overrun, but related
    • OpenSSL bug that leaked extremely private information
    • potentially addressed through information-flow type systems
Likely topics/questions on mini-exam #3

• From Lecture 19-2:
  – Language-based security
    • why is C so vulnerable to buffer overruns?
    • why are Java, Python, Haskell much less vulnerable?
Likely topics/questions on mini-exam #3

• From Lecture 19-2:
  – Language-based security
    • why is C so vulnerable to buffer overruns?
    • why are Java, Python, Haskell much less vulnerable?
      – memory safety / type safety
Likely topics/questions on mini-exam #3

• From Lecture 19-2:
  – Language-based security
    • why is C so vulnerable to buffer overruns?
      – no array bounds checking
      – casts between pointers and non-pointers
      – casts between different pointer types
    • why are Java, Python, Haskell much less vulnerable?
      – memory safety / type safety
Likely topics/questions on mini-exam #3

• From Lecture 19-2:
  – Language-based security
    • why is C so vulnerable to buffer overruns?
      – no array bounds checking:
        breaks memory & type safety
      – casts between pointers and non-pointers:
        breaks memory & type safety
      – casts between different pointer types:
        breaks memory & type safety
    • why are Java, Python, Haskell much less vulnerable?
      – memory safety / type safety
Likely topics/questions on mini-exam #3

- From Lecture 19-2:
  - more generally, however, there are few clear connections between implementation language and quality
Likely topics/questions on mini-exam #3

• From Lecture 19-3:
  – SQL code injection
    • how it works
    • how to stop it
  – URL manipulation
    • how it works
    • how to stop it
Bonus question

• involves Bogosys, security, and PDF files