A Strategy to Include Defensive Programming Tactics in the Undergraduate CS Curriculum at UMBC

By
John Simmons
Brian Roberts
Doug Cress
Problem

• There are too many security flaws in computer programs
  – Millions of hosts are hacked each year by buffer-overflows both remote and local
  – Ordinary people, companies, & governments suffer from these AVOIDABLE flaws
Proposed Solution

• Software Developers are rarely concerned with security because it is always included as an afterthought

• In most of the educational institutions security and defensive programming are not even mentioned in the software development classes

• Our solution:
  – Start mentioning them!
Proposed Solution

- Best way to kill a weed is to get at the root
- To improve the security of software we must start at the root of a software developer’s education – The undergraduate CS degree
- We felt the best way to accomplish this goal was to come at the students early and often, instructing them in Defensive Programming Tactics
New & Significance

• The problem of poorly secured software is a significant problem
• Any solution that seeks to overcome this significant problem is therefore also significant
• Our solution is new, because no one has ever tried it before at UMBC
• Our solution is also beneficial to the UMBC community because it helps satisfy criteria for the NSA Centers Of Academic Excellence in Information Assurance Education initiative.
General Research Track

• Analyze the current curriculum
  – Choose the set of classes to be influenced
  – Examine current and past class resources to determine best fit for our improvements

• Interview faculty members responsible for those classes
  – Solicited reaction to our suggested improvements
General Research Track

• Refine and develop materials to be used by instructors
  – Create html pages
  – Develop PowerPoint presentations
  – Create new project descriptions

• Present developed materials to instructors for feedback and delivery
Chosen Classes

• CMSC 201 – Computer Science I for majors
  – Basic C class teaching good programming practices
• CMSC 202 – Computer Science II for majors
  – Basic C++ class teaching more advanced topics
• CMSC 341 – Data Structures
  – Examining and implementing various ADT’s
• CMSC 345 – Software Design & Development
  – Introducing professional software development
Why we choose those classes

- All four of those classes are required for a major in CS
- The first three classes are required for a minor in CS
- Therefore we have the impact on the greatest number of students by affecting the classes they all have to take
Why we choose those classes

- These classes all have a focus on basic programming skills
- None of them mentioned security at all
- Starting with 201 the classes build to 345
  - Students gain more skill in programming itself
  - Students gain more knowledge about computers in general through their other classes
- Therefore they’re capable of more sophisticated instruction as they move through the curriculum
General Included Topics

• The potential harm of programming errors and poorly secured code
  – Physical damage that has resulted due to unreliable code
  – Widespread malicious cyber-events against commercial companies, government agencies, and educational institutions
  – Financial impact of stolen credit cards or more serious bank account fraud

• Poorly commented and non-standardized code and how it can contribute to security problems.
General Included Topics

• Common insecure practices that might be encountered at a student’s current instruction level
  – I/O data not validated
    • Can result in unexpected behavior in programs and invalid results
  – Segmentation faults due to accessing memory outside the bounds of an array
  – Data printed from the wrong memory addresses
    • Can potentially access and send sensitive information
  – Use of insecure functions such as strcpy() and gets()
  – Mishandled exceptions
General Included Topics

– Object Oriented programming issues
  • Class Destructors that don’t clear out sensitive data before the memory is deallocated
  • Lack of properly designed access control mechanisms utilizing public/private/protected/package modifiers
  • Read-only accessor functions that return pointers instead of copies of the data referenced
CMSC 201

- Influenced format and why
- Details of suggested improvements
- Reactions by instructors
CMSC 202

- Influenced format and why
- Details of suggested improvements
- Reactions by instructors
CMSC 341

- Influenced format and why
- Details of suggested improvements
- Reactions by instructors
CMSC 345

• Influenced format and why
• Details of suggested improvements
• Reactions by instructors
Overview of Changes

• 201 – 16 topics for inclusion
  – Approx. 32 slides worth of material
• 202 – Changed html pages and added slides
  – Approx. 9 slides worth of material
• 341 – Changed a project description & added 6 slides
• 345 – Added an entire lecture of 20 slides & provided improvement 4 detailed suggestions for the class project
Future Work

• Our research project suffered from a lack of time
  – To properly evaluate our ideas we would need several years to observe students and their reactions to our teaching methods
  – Additionally we would need to find a way to measure the effect of our ideas on the world software scene
    • Surveys, contests, testing within the classes
Future Work

- Create a secure programming class
- Look for opportunities for including security issues in all the other classes
Bibliography

• Current and past class materials for: CMSC 201, 202, 341, & 345

• All the instructors for all the classes
  – Mrs. Evans, Ms. Block, Mr. Raouf, Mr. Frey, Dr. Oates, Dr. Peng, Mr. Edleman, Ms. Mitchell, Dr. Pinkston

• NSA criteria for COE qualification