PA193 Secure coding principles and practices

Overview of the subject

Zdeněk Říha & Petr Švenda

CROCS
Centre for Research on Cryptography and Security

PA193 Secure coding principles and proposal

- Relatively new subject
  - Introduced in September 2013
- Secure coding
  - How to write code in a secure way
  - So that the program cannot be attacked/exploited
  - ≠ Programming of security applications
- 2/2/2
  - Lecture: 2 hours weekly
  - Seminar: 2 hours weekly (2 seminar groups)
  - Homework: about 3-6 hours weekly

Aims of the subject

- To learn how to program in a way that the resulting application is more secure
  - Free from security related bugs
  - Cannot be attacked/exploited
- To understand security consequences of decisions made by programmer
- Many issues are independent on programming language
- Most examples are based on C/C++ and Java

Position of PA193 among other subjects

- PV079 – Applied cryptography
  - Practical aspects of cryptography
- PV181 – Laboratory on Security and Applied Cryptography
  - Using common crypto libraries and smart cards
- PA018 Advanced Topics in Information Technology Security
  - Practical project
- PA168 Postgraduate seminar on IT security and cryptography
  - Discussions on current issues of IT Security
- PB173 Domain specific development in C/C++
  - Group focused on implementation security and applied crypto

Requirements

- Basic knowledge of (applied) cryptography and IT security
  - symmetric vs. asymmetric cryptography, PKI
  - block vs. stream ciphers and usage modes
  - hash functions
  - random vs. pseudorandom numbers
  - basic cryptographic algorithms (AES, DES, RSA, EC, DH)
  - risk analysis
- Practical experience in programming with C/C++ language
- Basic knowledge in formal languages and compilers
- User-level experience with Windows and Linux OS

Organization

- Lectures + seminars + homeworks + project + exam
- Homeworks
  - assigned every week/seminar
  - individual work of each student
  - expected workload: 3-6 hours
- Project
  - groups of 2-3 students
  - topic assigned in first half of semester
  - project defense in last seminar of the term
  - expected workload: 20 hours/project/participant

Grading

- Points
  - Homework (30)
  - Project (30)
  - Written exam (90)
- Grading
  - A ≥ 90% of maximum number of points
  - B ≥ 80% of maximum number of points
  - C ≥ 70% of maximum number of points
  - D ≥ 60% of maximum number of points
  - E ≥ 50% of maximum number of points
  - F < 50% of maximum number of points

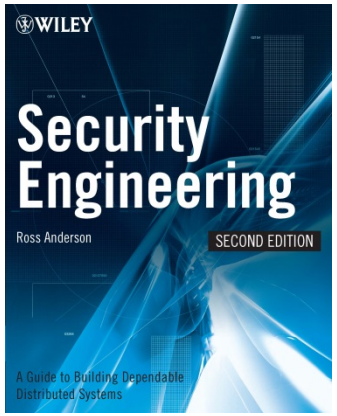
Attendance

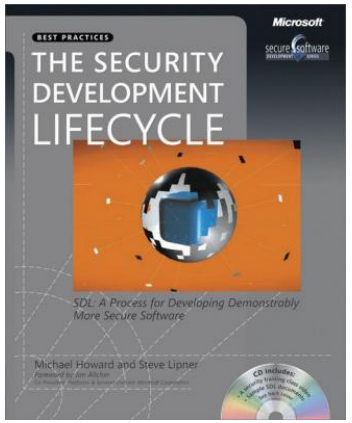
- Lectures
  - Attendance not obligatory, but highly recommended
  - Not recorded
- Seminars
  - Attendance obligatory
  - Absences must be excused at the department of study affairs
  - 2 absences are ok
- Homeworks and projects
  - Done during students free time (e.g. at the dormitory)
  - Access to our lab is possible

Course resources

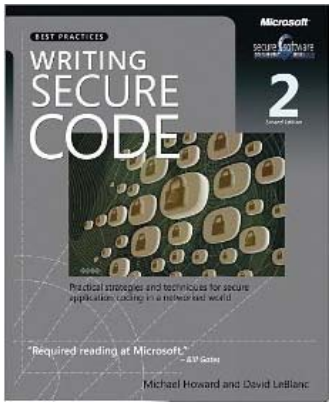
- Lectures (PDF) available in IS
  - IS = Information System of the Masaryk University
- Homeworks/assignments available in IS
  - Submissions also done via IS
- Additional tutorials/papers/materials from time to time will also be provided in IS
  - To better understand the issues discussed
- Recommended literatures
  - To learn more …

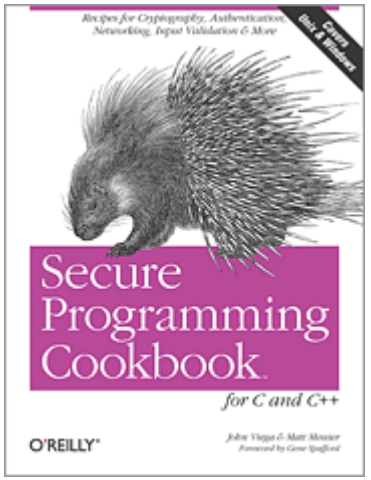
Recommended literature





- Ross Anderson - Security engineering, Wiley
- Michael Howard, Steve Lipner - Secure Development Lifecycle, MS Press
- John Viega, Matt Messier - Secure programming cookbook, O'Reilly
- Michael Howard - Writing secure code, MS Press





Plagiarism

- Homeworks
  - Must be worked out independently by each student
- Projects
  - Must be worked out by a team of 3 students
  - Every team member must show his/her contribution
- Plagiarism, cut&paste, etc. is not tolerated
  - Plagiarism is use of somebody else words/programs or ideas without proper citation
  - IS helps to recognize plagiarism
  - If plagiarism is detected student is assigned -5 points
  - In more serious cases the Disciplinary committee of the faculty will decide

Topics covered (1)

1. Language level vulnerabilities: Buffer overflow, type overflow, …
2. Defence in depth, …
3. Input processing (all input is evil …)
4. (Automatic) Code checking
5. Security testing: blackbox vs. whitebox testing, fuzzing, …
6. Access control, privilege separation, …
7. Automata based programming, securing API, …

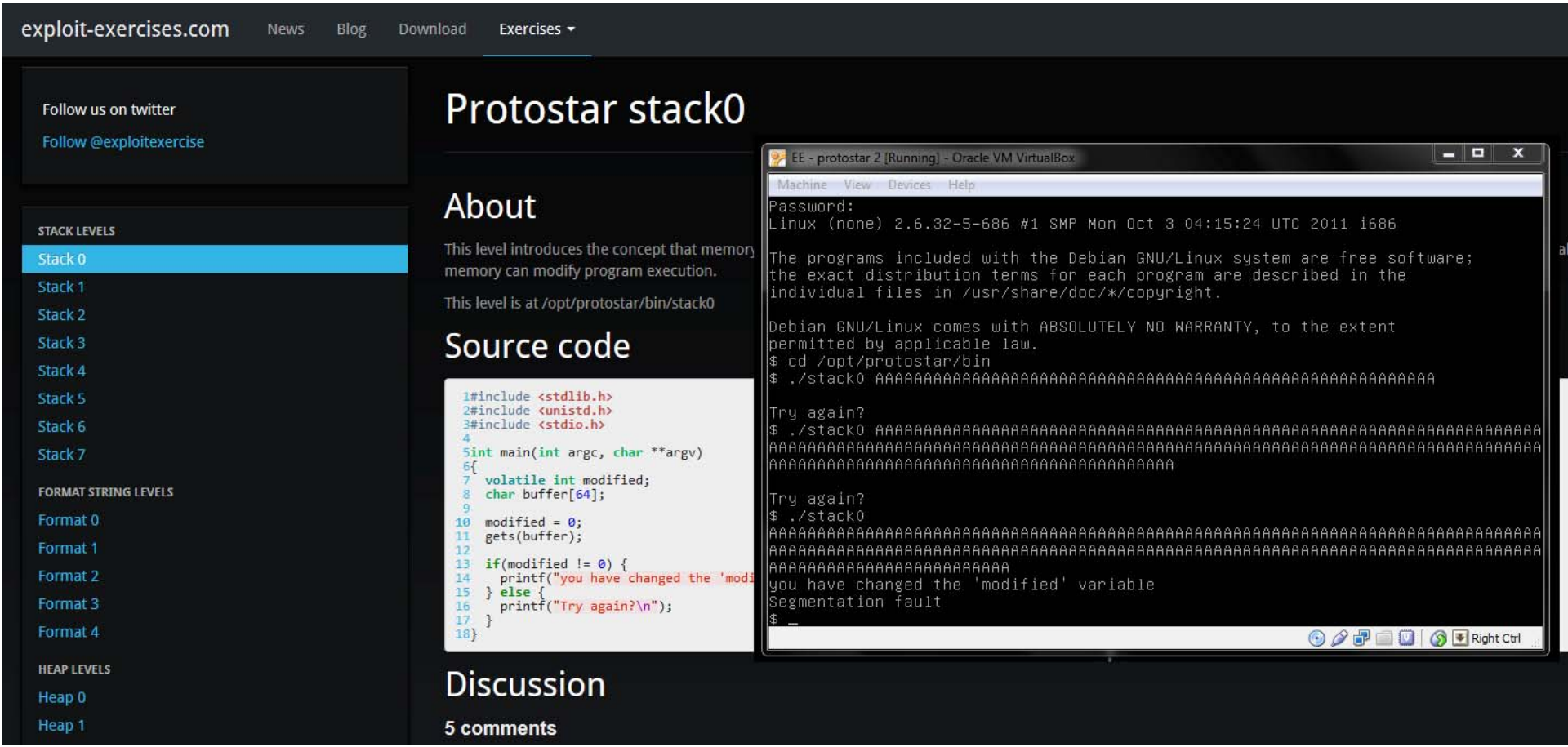
Topics covered (2)

8. Integrity of modules, parameters, temp files, …
9. Concurrent issues: IPC, race conditions, Valgrind, …
10. (Pseudo)random numbers, their generation and usage, …
11. Security primitives: secure channel, secure storage, key management, …
12. Security code review

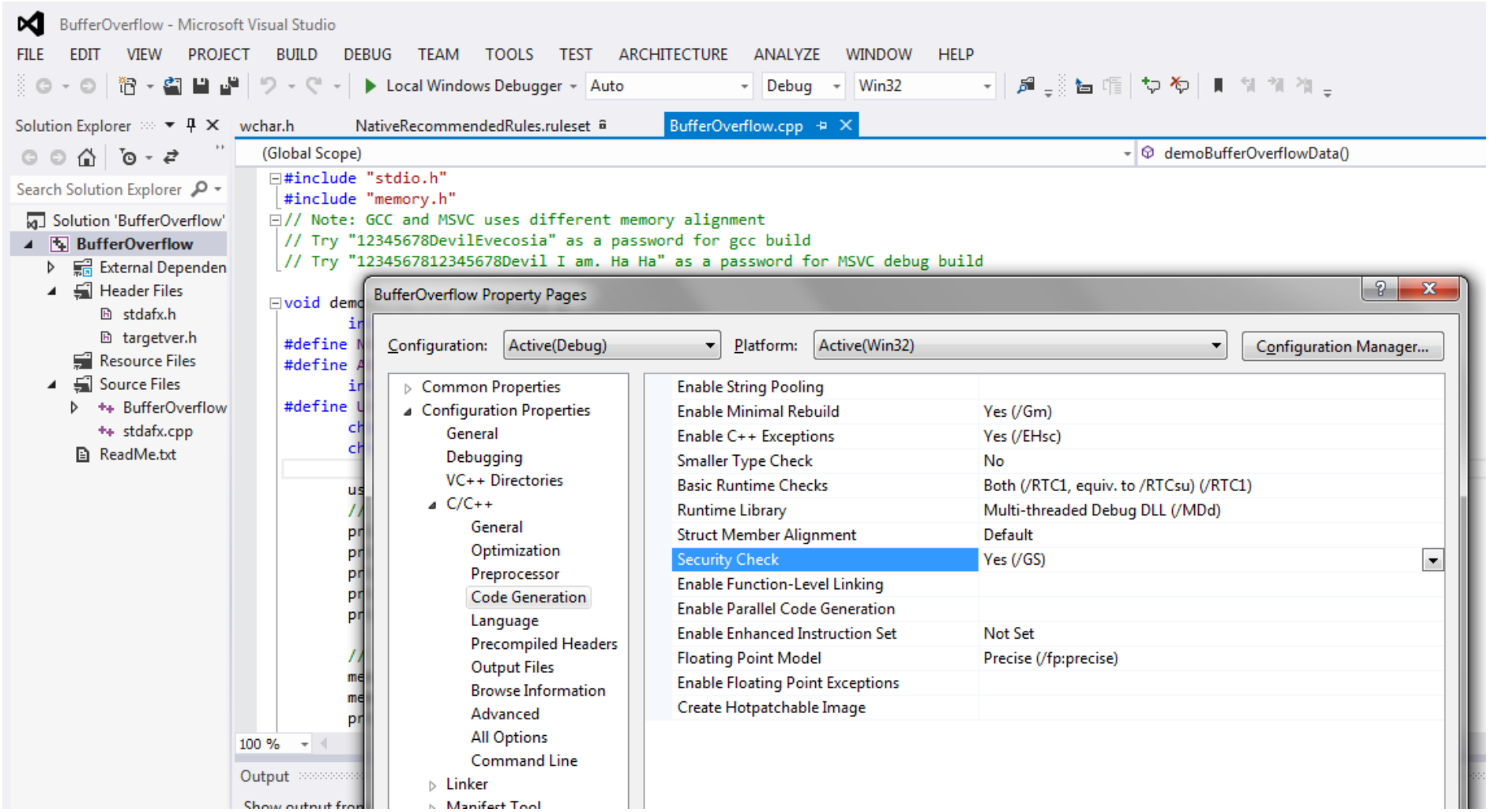
Labs - organization

- Dedicated teaching room close to security laboratory (G201)
- Pre-prepared environments (Windows, Linux)
  - compilers, analyzers...
- Virtual images for selected exercises
  - can be used also outside laboratory
- Necessary software available for students
  - freeware tools preferred for easy home-use

Protostar virtual image with exercises



Compiler settings for /DEP and /ASLR



Deeper look into disassembly