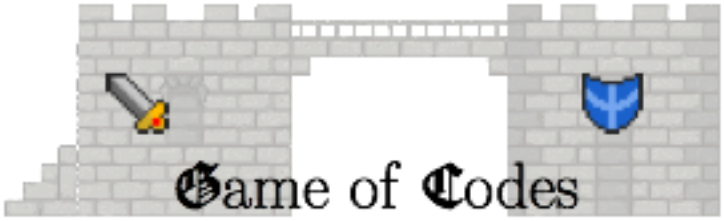
Secure Coding Project



Phase 2: Black-Box Testing

REMEMBER: This is a competition between teams!

- All students will be provided with link to (private) online survey
  → vote for top 3 best applications/presentations
  - 1st team you vote for gets 5 points
  - 2nd team you vote for gets 3 points
  - 3rd team you vote for gets 1 point

- For each team member that does not vote, their team will get -5 points

- Voting will occur after all the presentations are done in each phase

- Prizes will be handed out during the last Secure Coding lecture

- Prizes will be handed out to the top 3 teams

Phase 2: Black-Box Testing

- The bank for which you are developing the Internet banking web application is not convinced that your solution is secure
- Therefore, it hires other experts who will test your application for vulnerabilities
- You are also given one application to test, developed by one of your competitors
- **Your goal:** perform systematic black-box testing of:
  - Web-application from your competitors' team **AND**
  - **Your own web-application**
  - Argue whether your app is more secure
- **Focus:** application testing in a black-box manner. Problems with the configuration of the VM are not awarded any points.
- **Deadline:** 10:00 AM, 24th November 2015

Phase 2: Black-Box Testing

- **Target of black-box test:** VM delivered by your competitors.
  - Import VM and change the network settings from NAT to Host-only Adapter or Bridged Adapter, to see the VM from your host.
  - Access web-application through browser of your host OS
- NOTE: Use target as a black-box, don't try to break the password of the OS user (no points will be awarded for this)

- **Documentation to use for testing:** slides with use-case description provided by other team

- **Testing tools:**
  - Any tool installed on your own Samurai WTF VM
  - Any other tool you know and want to use (including tools from Kali Linux)
  - Develop your own custom scripts/plug-ins (highly appreciated)
- **Requirements:**
  - Each team-member must use at least 2 different testing tools! This means that there should be 8 different tools used per team. It is OK if you find different vulnerabilities with different tools.
  - Use a systematic approach, via OWASP testing guide
  - Don't test superficially! Use advanced features of tools (extend if possible)

Deliverables for Phase 2

- Deadline is (Tuesday) 24th November 2015 at 10:00 AM

  → You can hand in via **e-mail** or USB stick before this date at the office MI 01.11.040

  or on that date in the classroom (lecture will start at 10:00 AM sharp)

  →There will be a 1% grade penalty for every minute after the deadline

- E-mail / USB stick must contain 3 files:

  1. **BBTestingReport-TeamX-Phase2.pdf** (where X is your team number)
  2. **OWASP-Checklist-TeamX-Phase2.xls (Excel 97-2003)**
  3. **Presentation-TeamX-Phase2.pdf** (**NO .PPT or .pages files**)

     The structure of the report and presentation are presented on next slides

- Teams and students who present are picked randomly on the presentation date

  → If the picked student is not present s/he gets zero points for this phase (except if s/he can provide written evidence that the absence was justified)

Structure of Phase 2 Presentation

Presented in **MAXIMUM 7 MINUTES** during the lecture

1. First slide: team number and team members (full-names)

2. Second slide: enumeration of most important vulnerabilities in competitor's app
   - Mention name, impact and likelihood of each vulnerability
   - Impact and likelihood are measured as: low, medium or high
   - Use CVSS v3.0 template presented in next slide for computing the vulnerability score https://www.first.org/cvss/specification-document
   - **Remember:** video of tutorial 1 on Moodle

3. At this point in the presentation you either give a live-demo or video demo of the vulnerabilities or present them in more detail on the following slides

CVSS v3.0 template

	Enter your data below
Access Vector	
Access Complexity	
Privileges Required	
User Interaction	
Scope	
Confidentiality	
Integrity	
Availability	

Structure of Phase 2 Checklist

The checklist will not be presented during the lecture

- Custom version of the OWASP testing checklist https://docs.google.com/spreadsheets/d/1RZYzw7OXjjmiG0U1F8gYCV_SGrZY3guSncRwluwmiGM/edit?usp=sharing
- Fill out columns D, E, F and G in the spreadsheet from previous URL

- **Testing columns** (D and E) should be filled out with color-coded values: Secure (green), Vulnerable (red), Not Tested (yellow), NA for Not Applicable (gray)
  - Insert note/comment for each cell that is Secure (green) or Vulnerable (red) with reference to the page in the BB testing report document where testing for this vulnerability is performed (example notes are provided for a few cells of the spreadsheet linked above)

- See the description of tests in the OWASP Testing Guide https://www.owasp.org/index.php/OWASP_Testing_Project

Structure of Phase 2 Black-Box Testing Report

The report will not be presented during the lecture

NOTE: In the report you MUST talk about both your app and the competitor's app

1. First Page: Course title, Phase title, team number and team members (full names)

2. Second Page: executive summary (1 page description of important findings, conclusions)

3. Third Page: table of contents

4. Fourth Page: time tracking table (info / student / task at hour granularity. No more than 2 hours tasks!)
   Tasks should be evenly distributed – project management part (this will be graded)

5. Fifth Page: overview of most important observations with impacts, likelihood and risk estimation. If necessary add a table with a legend for special symbols or abbreviations used in the document

6. Next few pages: compare the 8 different tools used by your team. Which vulnerabilities can each tool find? Which vulnerabilities can they not find?

Structure of Phase 2 Black-Box Testing Report (continued)

Remaining pages: describe identified **vulnerabilities** mentioning the following for each one

- **Observation:** describe which part of the application is vulnerable and why?
- **Discovery:** how was this vulnerability exactly discovered? Which tools were used and which steps were performed? Which inputs did you give to the tools? Which were the outputs?
- **Likelihood:** what is the likelihood that this vulnerability is exploited? Which assumptions must hold and which skills must an attacker have?
- **Impact:** what is the potential impact of an exploit of this vulnerability? What could happen?
- **Instantiate the CVSS v3.0 template** for this particular vulnerability.
- **Comparison with your own application:** how and why is your app better/worse

NOTE: Each vulnerability should be described on a new page!