Semestral Project

PV286 – Secure Coding Principles and Practices

Spring 2023

CROCS

Centre for Research on Cryptography and Security

Project introduction

- Teams of three people
- Programming language from the following set
  - C, C++, Rust, Go, C#, Java
- Four phases (~3 weeks each)
- Up to 45 points awarded
  - Bonus points possible for exceptional contribution
- Questions
  - By email to xdufka1@fi.muni.cz or your project feedback person (to be assigned)

Project idea

- Project topic assigned in two weeks
- Implementation of an application that includes:
  - Input parsing (CLI arguments, files)
  - Data manipulation and data structures
  - Output formatting
- Allowed to use only standard library
- Focus on the security of the implementation
  - Use defensive programming
  - Write tests (consider using TDD)
  - Use code analysis tools

Project phase outputs

- Phase I – deadline 3rd week
  - Teams of 3 people, programming language, GitHub repository
- Phase II – deadline 6th week (10 points)
  - The first part of the implementation, report
- Phase III – deadline 9th week (15 points)
  - The final implementation, recording of a project presentation
- Phase IV – deadline 14th week (20 points)
  - Report of analysis of another team's project

Phase I

- Form teams of 3 people
- Agree on your programming language
  - C, C++, Rust, Go, C#, Java
- Create a **private** repository on GitHub
- Write an email to xdufka1@fi.muni.cz containing:
  - Team member names + GitHub usernames
  - Selected programming language
  - Link to the GitHub repository
- Deadline: 26. 2. 2023

Phase III

- Finalize the implementation
  - Include all test vectors in GitHub Actions
  - Test it for correctness and try to fix potential security issues
  - Release the final version/build on GitHub
- Prepare and record a presentation of your project (10 minutes)
  - Code overview
  - Description of your testing and analysis
  - Application demonstration (building and running)
- Deadline: 9. 4. 2023
  - Submit presentation slides and the recording to IS
    - Submission from this phase will be made available to reviewing teams
  - Setup your code for a review (see the following two slides)
    - You will be requested to add access to your repository to reviewing teams later on

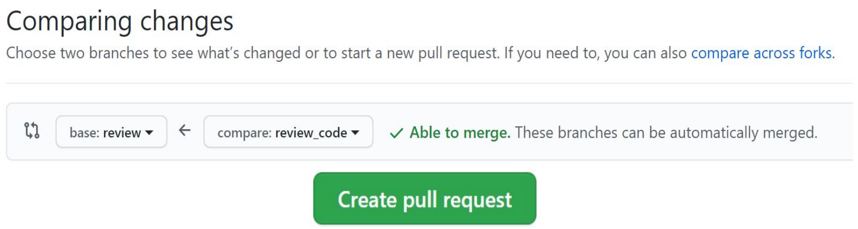
Phase III – Review setup

```
# Create review branch without code
git checkout -b review
git rm -r --cached .
git commit -m "Create review branch"
git push --set-upstream origin review

# Create branch for pull request into the review branch
git checkout -b review_code
git add .
git commit -m "Add review code"
git push --set-upstream origin review_code
```

Phase III – Review setup

- Create pull request from `review_code` to `review` branch

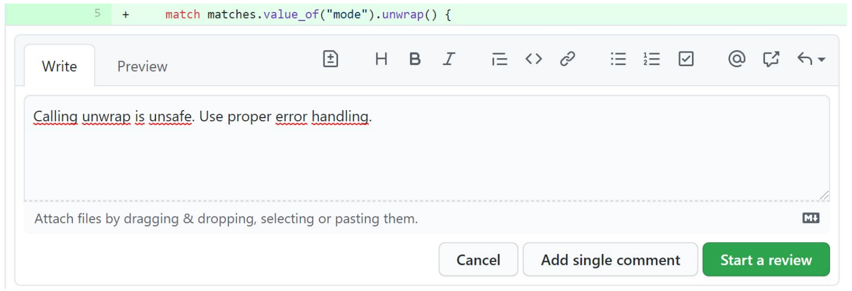


Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

✓ Able to merge. These branches can be automatically merged.

- Reviewing team will comment in the pull request



```
+        match matches.value_of("mode").unwrap() {
```

Calling unwrap is unsafe. Use proper error handling.

Attach files by dragging & dropping, selecting or pasting them.

Phase IV

- Analyze the assigned implementation
  - Overall code quality
  - Static analysis (at least 1), dynamic analysis (at least 1), fuzzing (at least 1)
  - Use at least 5 different tools in total
- Provide comments on the code in GitHub `review` branch
  - Consider using Conventional Comments
- Prepare pull requests fixing at least 1 discovered issue
- Write 3-4 page report of your analysis covering
  - Overall code quality
  - Used analysis tools
  - Discovered issues and fixes
  - Summary and score of the implementation based on your analysis (5 highest, 0 lowest).
- Deadline: 14. 5. 2023
  - Submit the report to IS