FAF.DAS16.2 Fall 2022 PBL Project Guideline

Original author: Mihail Gavrilița
Updated by: Vasile Drumea

Preliminaries

Nowadays a lot of projects are developed with little to no focus on the security aspects. Many developers take on projects that collect personal or sensitive information and forget to make sure that these are not leaked or misused.

With strict personal data laws that are getting implemented in more and more countries, there is a big risk of having serious problems in case the project does not correspond to the regulations. Besides unauthorised access to information in other projects, simply denying the service for a an arbitrary amount of time can cost the company money.

It is thus important to design secure systems from the beginning, rather then trying to retrofit security features in an existing application after encountering problems. This project aims to develop an application while approaching the security at all the stages of the development process.

Team formation

The first stage of this PBL Project is to form the teams. In order to make it somehow efficient, the team formation will be organized following the steps enumerated bellow.

- The students who voluntarily want to be leaders will have the chance to select teammates.
- Based on the number of student leaders:
  - if there are to many there will be a poll in order to select a top;
  - if there are to few leaders then the other students will randomly be assigned to teams;
  - if there are exactly how many leaders we need we proceed to teammates selection;
- Each leader will select a teammate when her/his turn comes.

After the teams are formed, each one should prepare a project proposal for the project they are going to work on. It is up to the students to come up with the topic/idea and in case there are questions the students should feel free to reach out to the mentors.

Getting Started

For starters, the students need to contact the university/company mentors. Their task is to provide guidance and help with any encountered questions/issues. A good idea for the teams is to choose a team leader who would interface with the mentors and would create a favourable working environment.

The students need to define communication channels between team members and the mentors. Also, the team should take care of task management. Ideally each member should manifest proactively and try to stay include in the work process. Any communication between students, or the mentors should be based on mutual respect.

1st Midterm Requirements

For the first midterm the students will be working on the following things:

1. Crystallizing the topic of the project:
   - Problem description/analysis,
   - Project subject,
   - Objectives.
2. Defining the requirements of the project:
   - Functional (What the project does?): Functions of the system,
   - Non-Functional (How the project does what it does?): Characteristics like performance, UX, system limitations etc.
   - CIA: Confidentiality, Integrity, Availability (These are also NF Requirements).
3. Creating a high level architectural model of the software (Recommended set of diagrams):
   - Use Case: how many are needed to cover all actors and use cases.
   - Sequence: one for each relevant functionality of the system.
   - Activity: same as $\uparrow$.
   - State Machine: in case you have entities with state transitions.
   - Class: in case you have data models bonded by relations.
   - Component/Deployment: how many the project needs.
4. Performing threat modelling:
   - Potential threats sorted by priority/impact:
     - Identify interfaces and data flow in order to know where the system might be attacked or have data interceptions.
     - The threats can appear at different levels of the system so please use logical groupings of the threats.
     - Map vulnerable cases to use cases.
   - In order to perform a risk assessment you can use one of the existing methodologies:

    - Attack tree,
    - DREAD,
    - STRIDE,
    - PASTA.
- Determine countermeasures and mitigation measures.

First, they will need to choose a topic for the project. A good topic would be one that allows to easily illustrate security related approaches in software development. For some hints on choosing a topic in general see the Annex at the end of the document.

Next the students will need to define the project's requirements. Usually, requirements come from the client the application is created for, but in this case the students need to defined an appropriate set of functional and non-functional requirements. Other sources of requirements might include community best practices, different standards or legislation. Special attention must be paid to explaining the security requirements that have been defined.

To better understand the application, it needs to be modeled. The students need to search and choose an instrument that would allow them to describe the architecture of the system, from a high level, and show how parts of it interact with each other. These models can be hand drawn pictures or diagrams in a well known standard such as UML.

Also, if needed, additional details on how the components could be split into sub-networks (e.g. public, DMZ and private networks) with emphasis on the security elements that make the system secure/protected.

Finally, the meat of the project, besides of course implementing the app, is the threat model. The instructions on this item are purposefully vague so that the students have the opportunity to freely research the topic. They'll need to read into how such analysis is performed and what can be the result of it. After this step, additional security requirements might appear, so those need to be added to the list of initial requirements.

In case you're wondering how the midterm will take place, it will go according to the list below:

- ~~It will start on 19.10.2022 at 17:00 in room 401.~~
- The teams will be assigned time slots so each team should be present only during their presentation.
- Each team will have 5 minutes to make a presentation with slides, if needed (remember, the slides are for the presenter/-s and the speech is for the audience). After that, 2-3 minutes for other things and questions.
- Each team leader (only the leader this time :)) should submit the slides and report v1.0 on ELSE at the corresponding assignments.

The re-evaluation will happen on 24th of November at 8:15AM in 118. The student will have to present the same slides and report. Also, will be taking into consideration the corrections.

2nd Midterm Requirements

For the second midterm the teams will have to make sure to cover the following:

1. Defining and prioritizing the defense mechanisms that need to be implemented in the application:

   - Having identified possible threats, the team needs to come up with mitigation strategies and possible defences.
   - Based on the value these could provide, or the severity of the impact of the threat, the countermeasures can be prioritized.

2. Developing the Minimal Viable Product (MVP):

   - There are no architectural constraints, but the developed project needs to correspond to the initial objectives & requirements.
   - All the components or architectural levels should be reinforced by security measures according to the performed threat modeling.

The students will need to create a list of defense mechanisms aimed at protecting the application from the vulnerabilities found during threat modelling. But, before starting to implement them, a logical order of implementation must be defined. Some mechanisms protect from critical threats, others provide defenses against an array of vulnerabilities. By starting from the most urgent mechanisms, the application becomes more secure in less time.

Of course, who needs security mechanisms if there is nothing to protect, right? Continuing the project, the teams will need to develop the idea into an MVP. They'll have to make sure to focus only on the most important features of the application. The aim of this project is the security of a system, not its features.

Lastly, they'll need to implement the defense mechanisms defined for the application. It is recommended to start with the most important/most urgent mechanisms to cover as much ground as possible. A system is never truly secure, but the hope is that at one moment, after implementing the most important defense mechanisms, it will become secure enough.

The date of midterm 2 by the timetable is 8th of December (Thu) also in 118. The students who manage to do the Coursera course can skip the presentation for midterm 2, but the course needs to be finished by 4th of December 23:59:59. In case the students did the course and also would like to do the presentation they should let the mentors know about this by the deadline of the course so that the midterm 2 presentations could be organize accordingly. In this case, the course completion will be considered as a bonus for the presentation.

Exam Requirements

For the exam the students need to prepare the following:

1. The report of the project and application in final states;

2. Preparing each other for the exam from the theoretical side of things.

After receiving feedback from the second midterm, it is time to finalize the project. The students will have to make sure to follow the instructions from mentors on how to enhance the application, what needs to be added and what needs to be re-done in the report. After applying all the needed changes, the team leader will upload the final version of the report on ELSE, together with a link to publicly available repository/-ies with the implementation. After this, the mentor will mark the final project with a specific grade, called in the $mentor\_grade$.

For the exam the students will have to do the following:

- Make a presentation of at most 7 minutes and after that have a Q&A session of at most 3 minutes;
- Upload the report on ELSE;
- Needless to say that all the team members should be aware of all the aspects of the project.

The report will have to be uploaded until the end of 25th of January of earlier. Bye!

Project Documentation

The deliverable components for this project are the presentation slides, the report and the developed software. At the presentation, the team should try to explain what they did, how did they do it. Also, motivations for the taken decisions should be provided.

In order to keep the presentations interesting the presenters should bring arguments and cover only the important stuff. The presentations should mimic the report structure.

The credo for the report of the project is ”High quality content, with low levels of bullshit”. There are no page limits, neither upper nor lower bound, so no one should fill the reports with water, but better use it to plant some trees.

For the structure of the report, one can take inspiration from the previous sections where the tasks were described. Also, it should contain a title page and a references section. Not stating the sources is considered plagiarism and *will not be tolerated.*

This project requires the use of LaTeX for text editing, and a safe choice which also provides collaboration capabilities is Overleaf). Also, Grammarly could be used as it is a free tool that will help fix not only spelling but also bad grammar and awkward sentence formulations.

If the project can be demoed, demo it at the end of the presentations. The source code must be version controlled and available to the exam committee at the day of presentation. Any good code repository has a good README file, obviously.

Evaluation

The final grade for this project will be calculated as follows:

$$\begin{aligned} final\_grade &= 30\% \times individual\_midterm1\_grade \\ &+ 30\% \times individual\_midterm2\_grade \\ &+ 40\% \times final\_exam\_grade \end{aligned}$$

Additionally the mentor grade will be given individually and will impact the 3 grades enumerated above. The mentor grade is not included in the formula and is up to the mentors to use it when grading. The midterm presentations are going to be team presentations which will receive a team grade. Individual midterm grades will then be calculated as follows:

$$individual\_midterm\_grade = team\_midterm\_grade + individual\_correction$$

For both Midterm 1 and Midterm 2 presentations, each team must submit a list of corrections in which they appreciate each member’s contribution to the project. Below are 2 examples of such a list:

Team 1	
Surname0 Name0	0
Surname1 Name1	0
Surname2 Name2	0

Team 2	
Surname3 Name3	0
Surname4 Name4	+1
Surname5 Name5	+1
Surname6 Name6	-2

The students will submit 1 list of corrections per team which will be the cumulative sum of the individual corrections. The possible correction values are `[-2.5, -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5]`, and their sum should always be equal to `0` (i.e. if some student gains points another should lose some). After the presentation, a student's individual midterm grade will be calculated by the formula above.

If, for example, both Team 1 and Team 2 have received an `8` for their presentations, their individual marks will be as follows:

Team 1	
Surname0 Name0	$8 + (0) = 8$
Surname1 Name1	$8 + (0) = 8$
Surname2 Name2	$8 + (0) = 8$

Team 2	
Surname3 Name3	$8 + (0) = 8$
Surname4 Name4	$8 + (+1) = 9$
Surname5 Name5	$8 + (+1) = 9$
Surname6 Name6	$8 + (-2) = 6$

As such, these corrections could be used to appreciate teammates that had a significant input to the project during the half-semester. Conversely, the corrections could be used to penalize teammates that did not prove to be helpful to the team during the project.

Good Luck! Don't disappoint!!

Annex

The following text was taken from "FAF-17x PBL Project Guideline", authored by Alex Burlacu. If it helped polish up the project topic, don't forget to say thanks to him.

Any project that you design should solve a problem, don't forget that. That means, whatever you do, have a couple of arguments of why you're doing it. With sound arguments you could even make the committee believe a programming language that you designed is indeed a suitable Bachelor thesis project, in case you thought about it.

- **Topic** – Topic must convey the concrete deliverable, in technical terms, that is the result of the Bachelor thesis. An example of a fairly bad name is: "Platform for restaurants" or "Digital whiteboard". These names are bad because they do not explain exactly what will be made during the work on the Bachelor thesis project. On the other hand, names like: "A client-server based system for tracking inventory" or "Mobile-based secure chat application with support for short video messages" are considerably better, for the reasons outlined above;

- **Subject** – The subject should summarize what exactly will be done in the project. Think of it as a more extended version of the topic, as if someone asked "OK, tell me more" or "What should I expect of it?". Remember, it's not a sales pitch and thus be honest and don't promise the sea and the salt. Use this as an opportunity to define well what to expect of this project, who are the target auditory and what is its scope. A bad example would be: "Creation of a service to help people save money", where's for the same project, a better subject would be: "Design and development of a service and a web-based client that will provide analytics on weekly and monthly income/expenses, divided by categories, and also will make custom recommendations on how to save money based on expense patterns of users";

- **Objectives** – Objectives are not test scenarios, and neither a task list. On the other hand, objectives do describe what are the system capabilities. Therefore, a good list of objectives define what functionalities the system has, and not how they are realised, and neither how it will be developed. Lists containing things like stages of a project life cycle, tools and languages that are to be used in the project, and generic lines like "The system will use artificial intelligence" or "Build a platform for X", are not to be present in an objectives list. If you still want to brag about what a cool thing will it contain, rather write it like: "The system will use `insert type of the AI algorithm to be used` to support `some functionality`", for example.