Elective Security in Web Development, KEA

Exam Project

Kristoffer Miklas

KEA

September 26, 2024

Introduction

We will have an exam project in this course, required for the exam.

You have about 25 minutes exam time. You are expected to present your project and walk through the features for 10 minutes focusing on what you did for security. Then afterwards we will ask questions, and perhaps dig into the code too.

Hand in of report and project before **December 20, 2024 at 23:59**

Provide a small back story for a company, half a page, number of employees, select a business, finance, agriculture, pet services provider, whatever you want.

You will build a small web application, consider it a template project for your company. Build a simple and secure project, with some functionality. Something the company can migrate their existing applications over to.

Project description

Create a small web site with at least the following features:

- Multilevel (privileges) login with backend authentication.
- New user registration.
- Data stored in cookie or other form (localStorage etc.)
- A list of items created by users, with option for setting visible private/public. Admin can see everything.
- Some items have a function for adding data, like adding a comment to an item or similar.
- File upload (images), a kind of profile picture might be an idea.

NOTE:
Unlike most other project I will happily accept existing code, **if** source is given, and you can explain it. The focus is not web site functionality creation, but focus is *analysis of possible vulnerabilities and what you do to mitigate risks*.

Example:
Validating email address is almost impossible, since the format allows for many features, which are often not used. You are therefore encouraged to find a good implementation of email validation, which suit your purposes in this project. You may copy this function/module into your code, including license and references to original.

You **must** be able to walk us through the module, what does it do, and what are the shortcomings of this, etc. Explain why **this** version is appropriate for your project. Maybe because it’s simpler to understand, it’s written in a clear way, your use of email as a user-id for login make it fair to have fewer features, etc.

You should not use a full featured backend framework like Django or similar. (Because it gives you much for free, and often it’s difficult to change implementation details.) If you do use a framework that handles many of your potential vulnerabilities, you must research and explain what it actually does in each case, and pen-test it.

You are allowed to use any combination of front end and backend languages. I expect that you will at least use some JavaScript, HTML, CSS etc. You may use existing helper libraries like bootstrap, and any readable/common programming language(s) you decide, like Java, Python, PHP, ...

Take steps to implement settings, security headers and/or code that prevents or minimizes the risk of at least the following attacks:

- SQL injection, and command injection.
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- XML External Entity (XXE) and serialization/injection. (See slideshow 9.)
- Client side manipulation. (So not trusting the client side, you must have server side validation.)

You are not expected to produce a mature and production quality implementation that can compete with existing frameworks. You **are** expected to be able to insert session ID, and CSRF Tokens in the right places. If you know of limitations in your product, discuss this in your report.

Recommendations

You should consider the following:

- Firewall – enable and configure the firewall on the server.
- Use of Transport Layer Security (TLS)
- Use of encryption and hashing. (Hashing passwords in the user DB table.)
- Configuration settings for your project, if using PHP php.ini, if using Nginx nginx.conf etc.

Deployment

Feel free to deploy on a virtual server somewhere, and configure the server as you like:

- Users, Apache, PHP,…
- Have a development environment.
- Maybe use a repository. (For source control.)
- Make sure you have a running copy on your machine, *so you can demo it at the exam*.

Deploy the application to a server of your choice. Example: Amazon, digitalocean, ...

Hint: Using a real name and deploying on a server will allow you to use tools like Mozilla Observatory for checking settings https://observatory.mozilla.org/

Formal stuff

- You can work in groups. Report size depends on group size!
  1 student maximum 20 pages, 2 students maximum 30 pages, 3 students maximum 40 pages, 4 students maximum 50 pages.

- About page numbers, you should not exceed the maximum, and much less may be missing something.

- It is highly recommended to be in a group - up to 4 students max!
- The exam is individual. So hand-in is individual, not as a group. But you hand-in the same report and exam project individually in Wiseflow. Wiseflow must be used.
- Before upload to Wiseflow the report and code must be ZIPed into a single archive.
- Your report should contain group info.
- Your project source code should be documented. (Commets in your code.)
- Relevant configuration files (httpd.conf, nginx.conf php.ini .htaccess …) should be included in the project if significant changes are done to these. Most relevant parts should perhaps be referenced, but not included in report. Example: ”The requirement for encryption was done using a configuration setting `add_header` in Nginx allowing us to have a HSTS header.”
- Remember references, at least include the book we used for the course.

Basically, the report should document **what** you have done, and **why** you have done it.

Work Load Comments

Please include rough estimate for time spent on this project. We have the last weeks for this project, and there is a lot of work included. I would like to set a maximum number of hours to about 100 hours per person. This is to ensure that above is NOT misunderstood, and that it does not result in a stressful experience.