Web Security 2019

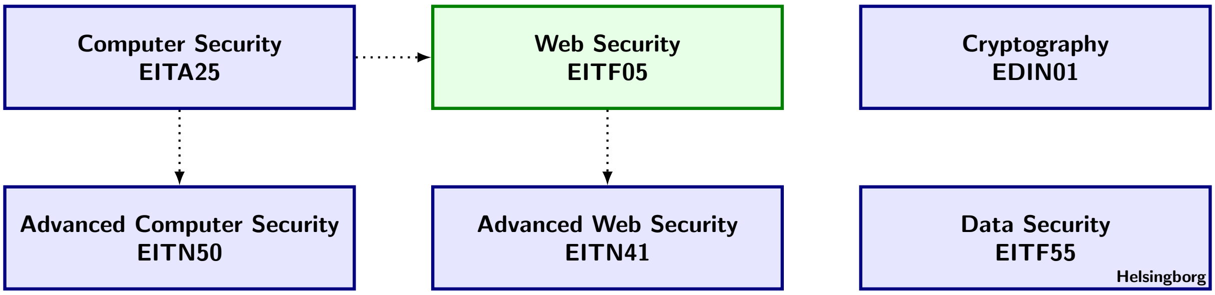
Project: Web Shop Under Attack

September 6, 2019

- This project will be done in groups of 3-4 people.

Learning goals:

- Implementing a simple and secure web shop application.
- Implementing and developing an understanding for some common web attacks.
- Improving presentation skills, both written and oral.
- Improved skills for collaborative work and role-based project configurations.


Computer Security
EITA25

Web Security
EITF05

Cryptography
EDIN01

Advanced Computer Security
EITN50

Advanced Web Security
EITN41

Data Security
EITF55
Helsingborg

1 Project description

The project has four main goals;

- to implement a simple and secure web shop application,
- implementing and developing an understanding for some common web attacks,
- to improve presentation skills, both written and oral,
- to improve skills for collaborative work and role-based project configurations.

While the project execution and outcome is the collective responsibility of the entire group, the project also includes specific role-based responsibilities for the individual group members.

The setting is as follows. A major company (the 'Company') has placed a first-time order for a secure web shop from a subcontractor. The project group represents the subcontractor.

The primary goal of the Company is to acquire a secure product with the specified functionality. Their secondary goal is to pay you as little as possible for your work.

Your primary goal is to become a long-term subcontractor for the Company. This is accomplished by providing robust technical solutions and demonstrating your excellence in the security domain.

The implementation, documentation and presentation parts of the project are described in separate sections below.

2 Role assignment

Each subcontractor employee has one of four different types of roles; chief architect, chief security analyst, chief tester and senior developer.

These roles are assigned according to the group member listing on the course home page. That is, the first person listed as a group member is chief architect, the second is chief security analyst, the third is chief tester and the fourth is senior developer.

Each group member is responsible for the project area most relevant to their role domain, but note that it will also be necessary to distribute the project work beyond these specific roles. In other words, you need to support your project group where help is needed!

3 Implementation

The implementation part of the project has three phases;

- construction,
- attack,
- functionality peer-review.

3.1 Construction phase

In the construction phase, you will build a basic but (almost) fully functional web shop. The basic requirements of the web shop are the following.

- Programming language is PHP.
- PHP sessions must be used in a reasonable way.
- The security issues related to PHP included in the course shall be considered (server and PHP configuration and protection from attacks).
- TLS will be used to secure the connection. The connection should at least be secure when sending sensitive information to the web server.
- The server is to be authenticated using a certificate (self-signed is ok).

- The user is authenticated using username and password. Define a reasonable password policy that balances complexity and security. Include explicit support for a password blacklist to exclude the most common passwords. User credentials are to be stored securely in a database of your choice (MySQL or any other). The credentials must be reasonably safe from on-line brute-force attacks and off-line TMTO/Rainbow attacks.

You are free to choose your own (visual) design of the web shop and it can sell anything you choose. Your application MUST support the following.

- New users signing up. When signing up, users choose a username, password and enter a home address.
- Existing users signing in. The username must be clearly shown on the page when a user is logged in.
- Adding items to shopping cart.
- Checkout and payment. The payment should just be simulated, but once finished the user should be presented with a receipt with all details of the purchase. That is, it should look real, although the payment functionality itself is just simulated.

Based on these requirements, your first step is to have the chief architect select and document a suitable architecture for the secure web shop. Then, the developers implement the designed architecture, giving careful consideration to all security requirements. After security features are implemented, testers verify that the required features are fully operational, and they are also responsible for maintaining a functional database for testing purposes. The security analyst is responsible for the knowledge base on possible attacks.

This concludes the constructional phase of you web shop, which should be more or less fully functional, like a real-world website, except for the payment part.

3.2 Attack phase

For the second phase of your implementation, you will go into attack mode. Consider the following list of web-based attacks:

- SQL Injection,
- Remote File Inclusion[1],
- Cross-Site Scripting (XSS),
- Cross-Site Request Forgery (CSRF),
- XSS + CSRF (circumventing CSRF protection using XSS).

The developers implement the attacks and testers verify that the attack implementations are fully functional. The security analyst has the technical responsibility for the implemented attacks and protections, so she chooses the most appropriate security measures for the developers to implement. Testers, again, verify the defense functionality. The security analyst is also responsible for disseminating knowledge on attacks and defense mechanisms among the project members.

Of the five attacks above, you will choose and implement (at least) three. You are free to come up with other suggestions as well, if you wish, but you must then discuss the sanity of your creative choice with Pegah.

Implement these attacks and use them to attack your own web shop. Make adjustments to your web shop to make it less secure as needed, so that the attacks are successful. You then also need to patch up your web shop to protect it against the attacks, so you can show that the attacks no longer work, according to the security analyst's specifications.

[1] Requires usage of two web servers.

Note that web servers and PHP are typically pre-configured to stop certain attacks. When implementing your attacks, you may need to weaken both the web page itself and the web server and PHP settings.

The effects of your attacks do not need to be severe in any way, but you must be able to clearly illustrate how the attacks work, and that you have successfully implemented them. A simple (visual) printout will suffice in most cases.

Every group member must understand how these attacks work.

3.3 Functionality review

You will perform one review of implemented functionality, and two peer-reviews of written reports. The functionality review is described here, while the report review process is described in Section 4.1.

Functionality reviews are to be performed pairwise (or possibly three) in groups. The functionality review is a face–to–face meeting where *all* group members and Pegah must be present. For signing up, time slots available for functionality reviews are displayed on Pegah's office door.

The expected output from a successful functionality review is a signed functionality review form. This form is available on the project home page (see projects), and it includes instructions on how to perform the functionality review itself. In short, it is a continuation of the role play in this instruction, and the role assignment remains the same. For preparation, it is advisable to read the instructions before you come to the functionality review.

4 Documentation

The expected documentation output is a report bundle in the form of a single pdf-file, 10–12 pages long, as specified below.

- Front page with the names of the group members. (1 page)
- Achitectural overview. (1-2 pages, chief architect coordinates)
- List of security considerations. (2-3 pages, chief security analyst coordinates)
- List of (academically acceptable and well-formatted) references. (1 page)
- Peer-reviews of your report. (2 pages)
- Improvement sheet. (1 page)
- Signed functionality review form. (1 page)
- Signed contribution statement form. (1 page)

Use a tex editor of your choice (Overleaf[2], Texmaker[3], TeXnicCenter[4],. . .) to compile the report bundle above into a single pdf-file. The report itself consists of the front page, an architectural overview of your solution, and a list of security considerations.

The architectural overview should clearly show the structure of your web shop implementation. You are required to draw an image to illustrate this structure (half page). The level of your description is the important focus here. Aim for a *very high*[5] level, do not go into insignificant details. Upon reading your architectural overview, the reader should clearly, correctly and quickly understand the structure of your web shop. Motivate your design choices where such have been made.

The list of security considerations should list all attack types and major security issues you can think of and explain if, why and/or how it is or is not applicable to your program. For each attack/issue, in a sentence or at most two, explain clearly and concisely what you have done to protect your solution. Or briefly explain why no protection is needed. You do not need to explain the attacks with more than

[2] Overleaf, `http://www.overleaf.com`, Last accessed on 2019-09-06.

[3] Texmaker, `http://www.xm1math.net/texmaker/`, Last accessed on 2019-09-06.

[4] TeXnicCenter, `http://www.texniccenter.org/`, Last accessed on 2019-09-06.

[5] Major components and information flow are good candidates for inclusion, while explicit descriptions of code, function names or UML diagrams are not.

a sentence (or at most two). Your report should clearly motivate your security related design choices. For this part, explaining how and why you have implemented password handling the way you have is a requirement. In the same way as for the architectural overview, aim for clarity, correctness and assimilation speed. That is, a reader should quickly be able to grasp the security status of your site.

For the reference list, if you are wondering what academically acceptable references look like, have a look in one of Paul's academic publications[6], but please note that not all articles listed are open access. (Why do you think that is?) Make sure that your report includes references to all books, articles, web pages and such that you have used when making your program. If you have not included a reference, then you are claiming that you are the original author. And do not use copyrighted material (images, tables,...) without explicit permission!

As a general tip on writing style, please use single spacing and a normal-sized font with serifs in your documentation. Playing around with large fonts or spacing just makes your report look weak, do not fall into that trap – you are better than that!

4.1 Double peer-review of the report

You are to produce peer-reviews of two written reports, and your report will be reviewed by two other groups. The process is as follows.

Project groups are divided into clusters of 3; groups 1-3, 4-6, and so on. If not divisible by 3, some cluster(s) may contain 1 group more or less. Project groups and cluster groups are listed on the course home page.

The cluster groups are to review all reports produced within the cluster. When you have written your report, distribute it to the other groups within your cluster group. You will receive one review of your report from each other cluster group, and you will review the report of every other cluster group.

A review is to be written densely on at most one A4 page. Verify that the report achieves the goals as specified in this instruction. Give encouraging feedback on the good parts, and identify the major points of improvement. You may check for structure, language, technical correctness, readability, and so on. Be constructive. Suggest ways of improving the report. You do not need to provide a general grade or rating.

When you have received the reviews of your report, read them and update your report accordingly. List or summarize the actions you have taken to improve your paper on at most one A4 page (improvement sheet).

4.2 Contribution statement

Individual project contributions of each group member are to be listed on the Contribution Statement Form, which is available on the course home page. All group members must sign the contribution statement.

Actively discussing and learning is also a way of contributing. Allow for the varying backgrounds of your group members. However, repeated failure to show up for meetings, or "failing" communication is not acceptable.

4.3 Submission

Last but not least, bundle up your report together with the reviews of it, the improvement sheet, the signed functionality review form and the contribution statement form (according to the list above). Hand it in electronically as *one* pdf-file, mail it to Urkund[7] at `pegah.nikbakht_bideh.lu@analys.urkund.se`.

Also, email your code and configuration files (httpd.conf and php.ini), zipped format is ok, to `pegah.nikbakht_bideh@eit.lth.se`.

Please note that those two email addresses are different.

[6] Paul's publication list, https://portal.research.lu.se Last accessed on 2019-09-06.

[7] Urkund, `http://www.urkund.com/en/about-urkund`, Last accessed on 2019-09-06.

5 Presentation

Your group is to perform a short but coordinated oral presentation in English. The main focus for this part is your presentation technique. How well can you handle giving a technical presentation within a small given time frame?

The presentations will be held in a mini-conference format. The other groups in your cluster, and Pegah, will be your audience. Note that this requires coordination among your cluster groups. Your first goal is therefore to decide on a presentation time that works for all cluster members. Match your schedules and book a time slot with Pegah. Some available time slots will be shown on the course home page.

The presentation itself is to be 7(±1) minutes long and contain/show the following.

- Describe (technically) two *new* (not in lectures) attacks from the OWASP attack listing[8].
- Explain where and how the attacks relate (or do not relate) to your implementation. You are encouraged to show actual code here.
- *Convincingly* explain what is needed for protection against these attacks. You do not need to implement this protection for your site, but you could use it to show where you need to do what.
- You are required to show code in your presentation!

All groups members must take part in the presentation, but it is not necessary that everyone speaks. Attending your own cluster session is mandatory. Attending other cluster sessions is optional.

Your main challenge here is the given time frame, so make your presentation concise and coordinated. Aim at making the presentation seem relaxed. You will need to rehearse this several times.

If you think that you can pull off a real-time demonstration of your site or code in your presentation, please do so. But make your presentation robust, so that you have presentation slides as a backup if something unexpected happens. A fully slide-based presentation is also an acceptable option. In either case, you need to convince your audience that you have a fully functional implementation, and this may be harder to do if your presentation is purely slide-based.

After your presentation, Pegah and the other groups will give feedback on your presentation. Did you achieve the presentation goals? Comment on efficiency, accessibility, flow, credibility, presentation techniques, future improvements, and so on.

Practice setting up your presentation quickly. The time schedule is as follows.

- Setting up first presentation. (3 minutes before scheduled session start)
- Presentation + feedback. (7+6 minutes)
- Setting up second presentation. (2 minutes)
- Presentation + feedback. (7+6 minutes)
- Setting up third presentation. (2 minutes)
- Presentation + feedback. (7+6 minutes)

If you need to borrow a computer for the presentation, synchronize with one of the other cluster groups, or let Pegah know well in advance.

Email the presentation file (pdf) to Pegah *the day before* your presentation, deadline at 23.59.

6 Delivery summary

1. Mail request for presentation time for your cluster group to Pegah.
2. Mail
   a) report bundle to Urkund.

[8] OWASP attack listing, `https://www.owasp.org/index.php/Category:Attack`, Last accessed on 2019-09-06.

   b) code and configuration files to Pegah.

3. Mail presentation to Pegah.

An action list is available on the course home page for your convenience.
Mail to Urkund: `pegah.nikbakht_bideh.lu@analys.urkund.se`.
Mail to Pegah: `pegah.nikbakht_bideh@eit.lth.se`

7 Hints, suggestions and comments

- Read and re-read these instructions until you understand what needs to be done. Ask if you don't.
- In order to maximize personal development, you are very much encouraged to assign project tasks according to *least* comfort. That is, for example, if you are more skilled at programming than writing, then enlarge your writing contribution in this project.
- Do not spend time making your website look nice. This is not a web design course. The security related design choices are much more important than other design issues. In other words, functionality matters very much, while looks and appearance do not.
- WampServer[9] is a package for Windows that includes the Apache web server, PHP and mySQL. It is easy to install and use. You are of course free to use other alternatives. XAMPP[10] is another alternative that is also very easy to use and available for Windows, Linux and OS X.
- After installing a web server, your first step is to write a minimal PHP page and display it in your browser.
- You are not required to use more than one computer, you can implement everything locally.
- The server certificate can be created using OpenSSL. You can use "openSSL req -new -x509 -nodes ..." to create the certificate. Note that -nodes will save your private key unencrypted, which is required if you use Windows.
- You are encouraged to discuss design alternatives with other groups, within and outside your own cluster, but each group makes their own implementation, documentation and presentation.
- Do not disclose your report to your review groups beforehand – keep it a secret until it is time for review. If you do not, you will ruin the pedagogics of the feedback process.
- You are not allowed to use a PHP framework, like cakePHP or similar. You are allowed to use JQuery for design purposes as long as it does not affect any security part of your solution. In other words, you must implement all security related parts yourself using only standard PHP tools.
- You are encouraged to divide the work among the members of your group. However, every group member needs to understand how the attacks work, and how they are implemented and mitigated.

[9] WampServer, `http://www.wampserver.com/en/`, Last accessed on 2019-09-06.

[10] XAMPP, `https://www.apachefriends.org`, Last accessed on 2019-09-06.