Mittuniversitetet
MID SWEDEN UNIVERSITY

Project: A Secure Web Application

Daniel Bosk[*]

owasp.tex 2128 2014-12-14 14:14:59Z danbos

Contents

1 Introduction **1**

2 Goals **2**

3 Reading Instructions **2**

4 Assignment **2**
4.1 Development of a Secure Web Application 2
4.2 Auditing a Web Application 3

5 Examination **3**
5.1 The Project Report 3
5.2 The Audit 4
5.3 Grading Criteria 4

1 Introduction

As the popularity of the Web grows, and hence more applications are implemented as Web applications, the importance of security in this area is ever growing. Earlier, applications could be run locally and have all data stored on the local hard-drive, thus it was only accessible to those with physical access to the system. The move of these to the Web has totally changed this. Now, everyone can access the system through a Web browser, and this makes private data more vulnerable than before.

However, Web application security is, unfortunately, not that easy. It took many years before Facebook started to use secure connections (HTTPS) for all pages after logging in to their service [Sin11]. Before this, anyone who logged in to their Facebook account could have their session hijacked by someone with access to the same network, e.g. an open wireless network in a coffee shop.

[*]This work is licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported license. To view a copy of this license, visit `http://creativecommons.org/licenses/by-sa/3.0/`.

Kamkar [Kam10] showed how flaws in a long line of Web applications, the Web interface of home-consumer routers among others, could be exploited for a sophisticated attack. On top of that, there are countless other examples of large Web applications which have had vulnerabilities, e.g.: Sony PlayStation Network [Ley11; Hun11], Ubisofts Uplay [Tho13], Yahoo Voices [Bra12; Goo12; Clu12], RockYou [Cub09] and Gawker [Gus10; Obe10].

2 Goals

The goals of this project are, that after completion you should be able to:

- state the most common attacks against Web applications.
- explain how these attacks work.
- apply methods for preventing these attacks in Web applications.
- audit application code to find vulnerabilities.

3 Reading Instructions

The main literature for the development of the project is the books by Gollmann [Gol11] and Anderson [And08], the documents from the OWASP project [OWASP13; MKC08; ASVS14] and the papers by Komanduri et al. [Kom+11; Sha+14]. Of course the same literature applies for the audit of a Web application as well.

4 Assignment

The assignment consists of two parts. The first part is the development of a secure Web application. The second part is to audit a Web application developed by someone else, to find possible vulnerabilities or to verify its security.

4.1 Development of a Secure Web Application

You are to develop a system for handling user accounts and storage of user data. This could for example be a prototype for an e-shopping Web site. The functionality you are required to have are the following:

- The user must be able to create an account.
- A visitor must be able to visit the site without first registering, but of course all functionality will not be available then.
- The users' accounts must be assumed to store sensitive personal data, e.g. address and bank account information.
- There should be a page where the user can view and edit his or her own profile.

- The users must be able to post comments which are readable for all, even for visitors who are not signed in.

As long as the above criteria are fulfilled, you are free to include any other functionality.

It is good practice to include security from the very beginning of development. As such, it is recommended that the initial time of your project is spent on setting up the environment (e.g. Web server, database server, etc.), and when you have attended a few lectures or a workshop, then you start development.

Another important aspect is writing clear and well-structured code. This makes reading the code much easier, this is beneficial for both yourself and anyone else who, sooner or later, will have to read that code again. Code which is easy to read, is often more secure – as it is easier to audit it for vulnerabilities.

4.2 Auditing a Web Application

In this part you should audit a Web application developed by someone else. First try to use their application, get to know it, that will make reading the code easier.

You are supposed to find all possible vulnerabilities in the given system. Start with the risks in the OWASP Top 10 [OWASP13]. If you cannot find any vulnerabilities, then take the approach of trying to argue for why there should not be a vulnerability in the component you are currently examining.

In addition to the application source code, you will also have the developer's report available as support. But, do not let the report guide you astray, the developer might think things are correct when they in fact are not.

5 Examination

This assignment may be done in groups of up to two students, both the development and the audit. If you have this possibility, it is recommended that you do. However, make sure to participate equally, the collaboration is for furthering discussions and avoiding security mistakes, not to divide the work in half. The examiner will choose one of you randomly to answer all questions during the presentations.

5.1 The Project Report

For the examination of the Web application you will write a report. This report is to be handed-in in the course platform, as well as handed to the auditor. Ensure the following:

- You cover all theory required to understand the report.
- The report must contain a clear description of how you have secured your developed Web application – and why that makes it secure.
- The report must be written using academic language, in either Swedish or English, have correct references, and be in PDF-format.

The report must be published for the audit before the deadline, see the booking feature in the course platform for current date and time.

For a short guide on writing reports, see the University's report guidelines in [Mit12]. The methodology chapter is not necessary for this project, and you may adjust the chapter titles a bit to better suit your needs. Remember, the emphasis is on clarity and coherence. Read IEEE's citation reference guide [Gra09] to get your citations and references correct. The Writing Lab in the Pudue University English Department provides a tutorial [Pai+13] on the matter as well.

The report must also include the project source code. However, do not include all source code as text in the report, include it in an "executable" form in separate files. You are, of course, still allowed to include snippets of your code in the report to explain certain details.

5.2 The Audit

The auditor will thoroughly test the application and read the code to find possible vulnerabilities. The auditor will write a short report which

- clearly summarizes the findings, either vulnerabilities or arguments for the security of the different components, and
- is written using academic language, in either Swedish or English, have correct references, and is in PDF-format.

You do not need to follow any particular template for this report, the emphasis is on clarity.

The auditor will then prepare a short presentation of the project in general, as well as his or her results from the audit report. A copy of the report must also be handed to the developer before this presentation is held. The presentations will be held at times specified in the course schedule. The developer must be present and may complement the auditor's presentation.

The presentation should take no longer than 10 minutes, then there is room for 5 minutes of discussion and questions from the audience.

5.3 Grading Criteria

After the audit has been done, the report and project must be handed in for grading. See the Project drop-box in the course platform for deadlines. Note that you are allowed to fix any problems discovered by the auditor before finally handing in the report to the course examiner.

The following grading criteria will be used for grading the report:

E The theory section of the report correctly explains the risks in OWASP Top 10 [OWASP13]. The Web application is protected from these risks, with few mistakes leading to vulnerabilities. The report fulfils the basic criteria for format of the report above, and it explains the security solutions used to protect the application.

D The criteria for grade E are met, but the ones for grade C are not fully met.

C The same criteria as for grade E, and in addition: The code is well-written, clear and well-documented. The report contains, in most cases, explanations for why the selected security solutions are adequate. The overall design is good, i.e. designed to be secure.

B The criteria for grade C are met, but the ones for grade A are not fully met.

A The same criteria as for grade C, and in addition: The report contains, for all cases, explanations and solid arguments for why the selected security solutions are adequate, and why they hold. There are no possible vulnerabilities, not even minor ones which are not directly covered by OWASP Top 10.

References

[ASVS14] Sahba Kazerooni, Daniel Cuthbert, Andrew van der Stock, and Krishna Raja, eds. *OWASP Application Security Verification Standard 2014*. 2014. URL: `https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf`.

[And08] Ross J. Anderson. *Security Engineering. A guide to building dependable distributed systems.* 2nd ed. Indianapolis, IN: Wiley, 2008. ISBN: 978-0-470-06852-6 (hbk.) URL: `http://www.cl.cam.ac.uk/~rja14/book.html`.

[Bra12] Anna Brading. *Yahoo Voices hacked, nearly half a million emails and passwords stolen.* July 2012. URL: `http://nakedsecurity.sophos.com/2012/07/12/yahoo-voices-hacked/`.

[Clu12] Graham Cluley. *The worst passwords you could ever choose exposed by Yahoo Voices hack.* July 2012. URL: `http://nakedsecurity.sophos.com/2012/07/13/yahoo-voices-poor-passwords/`.

[Cub09] Nik Cubrilovic. *RockYou Hack: From Bad to Worse.* Dec. 2009. URL: `http://techcrunch.com/2009/12/14/rockyou-hack-security-myspace-facebook-passwords/`.

[Gol11] Dieter Gollmann. *Computer Security.* 3rd ed. Chichester, West Sussex, U.K.: Wiley, 2011. ISBN: 9780470741153 (pbk.)

[Goo12] Dan Goodin. "Hackers expose 453,000 credentials allegedly taken from Yahoo service". In: *Ars Technica* (July 2012). URL: `http://arstechnica.com/security/2012/07/yahoo-service-hacked/`.

[Gra09] D Graffox. *IEEE Citation Reference.* Sept. 2009. URL: `http://www.ieee.org/documents/ieeecitationref.pdf`.

[Gus10] Sam Gustin. "Gawker Media Websites Hacked, Staff and User Passwords Leaked". In: *Wired* (Dec. 2010). URL: `http://www.wired.com/threatlevel/2010/12/gawker-hacked/`.

[Hun11] Troy Hunt. *A brief Sony password analysis.* June 2011. URL: `http://www.troyhunt.com/2011/06/brief-sony-password-analysis.html`.

[Kam10] Samy Kamkar. *How I Met Your Girlfriend.* Föreläsning, DefCon18. URL: del 1 – `http://www.youtube.com/watch?v=fEmO7wQKCMw`, del 2 – `http://www.youtube.com/watch?v=2ctRfWnisSk`, del 3 – `http://www.youtube.com/watch?v=vJtmZZGcR54`. 2010.

[Kom+11] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Christin Nicolas, Lorrie Faith Cranor, and Serge Egelman. "Of passwords and people: Measuring the effect of password-composition policies". In: *CHI.* 2011. URL: `http://cups.cs.cmu.edu/rshay/pubs/passwords_and_people2011.pdf`.

[Ley11] John Leyden. "Security watchers unpick PlayStation hack". In: *The Register* (May 2011). URL: `http://www.theregister.co.uk/2011/05/13/veracode_playstation_hack_analysis/`.

[MKC08] Matteo Meucci, Eoin Keary, and Daniel Cuthbert, eds. *OWASP Testing Guide.* 2008. URL: `http://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf`.

[Mit12] Mittuniversitetet. *Rapportmall för tekniska rapporter.* 2012. URL: `http://ver.miun.se/latex/miunthes/thesis/thesis.pdf`.

[OWASP13] The Open Web Application Security Project. *OWASP Top 10 - 2013. The Ten Most Critical Web Application Security Risks.* June 2013. URL: `http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf`.

[Obe10] Jon Oberheide. *Brief analysis of the Gawker password dump.* Dec. 2010. URL: `https://blog.duosecurity.com/2010/12/brief-analysis-of-the-gawker-password-dump/`.

[Pai+13] Joshua M. Paiz, Elizabeth Angeli, Jodi Wagner, Elena Lawrick, Kristen Moore, Michael Anderson, Lars Soderlund, Allen Brizee, and Russell Keck. *In-Text Citations: The Basics.* Nov. 2013. URL: `https://owl.english.purdue.edu/owl/owlprint/560/`.

[Sha+14] Richard Shay, Saranga Komanduri, Adam L Durity, Phillip Seyoung Huh, Michelle L Mazurek, Sean M Segreti, Blase Ur, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. "Can long passwords be secure and usable?" In: *Proceedings of the 32nd annual ACM conference on Human factors in computing systems.* ACM. 2014, pp. 2927–2936. URL: `http://lorrie.cranor.org/pubs/longpass-chi2014.pdf`.

[Sin11] Ryan Singel. "Facebook Enables HTTPS So You Can Share Without Being Hijacked". In: *Wired* (Jan. 2011). URL: `http://www.wired.com/threatlevel/2011/01/facebook-https/`.

[Tho13] Iain Thomson. "Ubisoft admits major hacking breach, advises password change". In: *The Register* (July 2013). URL: `http://www.theregister.co.uk/2013/07/02/ubisoft_data_breach/`.