WebGoat evaluation and lesson development

Project specification

Web application security

Deltagare: Anto Cvitic, Kristoffer Svensk
Handledare: Mads Dam

Abstract

WebGoat is a deliberately insecure web application developed by OWASP - a free and open non-profit organization with the aim to strengthen web application security by various projects where WebGoat is one of them. The purpose of WebGoat is to educate in where security breaches and vulnerabilities often occur in web applications and how to check if your own application might be at risk for an attack. It is in use by companies such as Google for educating their employees. The application works by provding the student a choice of many specific lessons on exploitation techniques of various vulnerabilities most commonly found in web applications. Each lesson has a lesson plan and hints where the lesson plan explains the background of the problem and suggests ways to exploit it, then the application gives an example of the vulnerability and the students task is to exploit it. For this essay the primary concern will be evaluating the application from a student point of view, questions asked is how well do the lessons teach specific vulnerabilities in web applications, and how well do targeted users acquire targeted skills? If we find any shortcomings in the lessons or lack of lessons we intend to provide our solutions and patches to the project.

Problem

Our intent is to help improve the WebGoat application by evaluating it as a teaching platform and suggesting improvements. Doing so we help solve the wider issues of web application insecurity. Web application development is problematic due to its wide span of technological complexity which stem from a need to keep backwards compatibility to previous standards, standards which weren't designed for the task they are applied to today. It is very difficult to teach web application security due to the need for the student to understand this myriad of technologies proficiently in order to exploit these. One solution to this is an interactive learning environment where the student not only reads about the technologies, vulnerabilities and techniques to utilize it, but also performs the mentioned techniques for exploitation. A problem solving approach is widely described and utilized in education. For this essay we intend to evaluate how good the WebGoat application reaches its goals and we intend to further improve it by suggesting improvements and writing lessons.

WebGoat contains several lessons which belong to the same type or domain of attack, for example there is around dozens of AJAX-based and Injection-type attack vectors to learn which are variations on the same underlaying flaw. Due to time constraints we will not evaluate all the lessons of any domain but rather choose a subset of these and select any other specific lesson outside any domain so that we cover as much as possible.

Project plan

We intend to acquire the security skills necessary for evaluating the lessons before setting out to try the lesson. We acquire the security skills from varying sources, such as The Open Web Application Security Project (http://www.OWASP.org/) and Web Application Security Consortium (http://webappsec.org/) will work as our primary reference. We also read the lesson plan and any background information provided by WebGoat. Following that we attempt to solve the lesson in question noting our progression. The notes we take are used to later develop new or suggest improvements to current lessons. In order to develop the lessons we need to understand how WebGoats lessons are built, by looking at examples from current lessons and reading the documentation of WebGoat we can make our own. Our pre-research shows it is possible within the time-frame to both evaluate and write new lessons. If we find any flaws in lessons we will send patches and our suggestions to the relevant maintainers.

While doing the lesson we might find it is too difficult or unsolvable for us given the information by WebGoat. In such a case we should note it and suggest ways to improve it. If a lesson lacks certain aspects for completion but is still understandable, we fill in the gaps. If a lesson is repetitive, we suggest removals.

1. Acquire understanding of a specific selection of security vulnerabilities, such as SQL-injections, HTTP Splitting.
2. Attempt to finish lessons in WebGoat
3. Note our progress, difficulties and impressions from the lesson. Can we with the given WebGoat information solve the lesson? How many hints did we need and where they useful?

4. Compare to our understanding of the problem the skills acquired by the lesson. Are we satisfied? Is it enough to learn the skill only from WebGoat? Do we think we now could develop web applications while avoiding introducing this vulnerability?
5. Look at source code of current lessons, learn how to write new lessons.
6. Write new lesson plan, implement lesson.
7. Suggest other improvements to current lessons.

Background

The WebGoat application contains lessons in how an attack can be performed and simulated on the local computer. As web applications are becoming more and more common in society with the introduction of web 2.0 and cloud-computing it is of utter importance to maintain security in this new domain. Web applications are unlike desktop applications open to a wider audience, often to the public and as such require better security than desktop applications as user often share their data through the web application. If one user compromises the system then other users also suffer.
Cross-site scripting allows attackers to introduce a client-side script into a webpage which can be used to bypass access controls on another users station thus compromising another system through the mutual use of the web application. While SQL injection is a technique that involves code injection into a database layer of an application, it is possible for attackers to gain access to large sets of data on other users such as their email and if the web application does not utilize hashed passwords, even their passwords.

A combination of these attacks was most likely used when *Bilddagboken, a Swedish community site,* was hacked in the beginning of 2008, where an attacker came over almost three hundred thousand user accounts which forced the site to immediately shut down until all user have had a chance to change their passwords.[1] As other research has shown most users use the same password on different services, thus the breach at *Bilddagboken* puts users at risk also at other services. This is an example on how important security is for web applications, especially when they contain sensitive data like personal information, e-mail address and password. Because not only can the attacker, in this case, access a users community account but can also bring access to the users e-mail and from there to other services as well.

Current state of art for web application security is under active development, from protocol based solutions like DNSSEC[2] which is currently under deployment and the root server is soon about to be signed - to a browser add-on which allows auditing and testing of security[3]. Microsoft also has a central collection of advices for their proprietary solutions.[4] However OWASP is the most influential free and open source project with the goal to strengthen we application security by educating.

Schedule

For this essay we have about four weeks time to completion. Our plan is for the first two week is to acquire the necessary information we need to begin our evaluation so we can later on choose a handful of lessons to evaluate, primary AJAX based attacks, SQL injection, XSS and try to solve them using the information we have learned and from what's provided by WebGoat. We should take detailed notes on what we experience during the lessons and on what we think was good or bad and how to improve it.

For the third week we should process this information we have gathered and use it to try to create a new lesson, suggest possible improvements to developers and contribute this to the essay.

For the last week the totality of the essay should be complete and this should give us time to go through and finalize the essay.

Week 1. Study x technique/security vulnerability and do the lessons, take detailed notes, evaluate. Write background on Essay-party and introduction.
Week 2. Study x technique/security vulnerability and do the lessons, take detailed notes, evaluate. Write about choosen lessons and security holes studied.
Week 3. Process information and create new lesson and suggest possible improvements.
Week 4. Finalize essay.

References

[1] http://www.idg.se/2.1085/1.139967
[2] http://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions
[3] https://addons.mozilla.org/sv-SE/firefox/collection/webappsec
[4] http://msdn.microsoft.com/en-us/library/ms994921.aspx