Proposal of Improving Web Application Security in Context of Latest Hacking Trends

RADEK VALA, ROMAN JASEK
Department of Informatics and Artificial Intelligence
Tomas Bata University in Zlin, Faculty of Applied Informatics
nám. T.G.Masaryka 5555
CZECH REPUBLIC
vala@fai.utb.cz, jasek@fai.utb.cz

Abstract: - Small business web application vulnerability statistics are growing in recent and new forms of hacker's attacks, such as Watering Hole Attack are appearing. Well secured enterprise web applications are threatened by smaller vulnerable webs misused by hackers in targeted attacks. This paper discuss an effective and low-cost way of creating better secured web application, applicable also in small business sector. The main issue – improperly validated user inputs, the biggest security vulnerability often misused by hackers – could be effectively fixed by open-source HTMLPurifier library which is usable in various programming languages. Moreover, the article suggests model of web application with proactive on-line embedded intrusion detection system.

Key-Words: - web application security, intrusion detection system, injection, watering hole attack, development framework, script monitoring, database profiling, QCubed, XSS

1 Introduction

Using network-based services is an integral part of modern human being. Sensitive information is provided every day to untrusted web applications and therefore not only great web portals are facing the threat of web attacks. In addition, according to the Symantec's Internet Security Thread Report [1], the largest growth area for targeted attacks in 2012 (31%) was business with fewer than 250 employees. It points to the fact, that small businesses which believed that hacker attacks are targeted especially to larger companies are now very threatened. In addition, overall security level of small business web applications is generally very low in comparison with web portals of great companies. Main problem is in insufficient security policy and lack of sophisticated hardware or software Intrusion Detection System (IDS) [2], because in this sector is usually small financial budget for IT security.

The growing number of web attacks in small business sector is connected with a new type of hackers attack, called Watering Hole Attack. [3]

2 Problem Formulation

During last decade, various IDSs and firewalls were introduced and applied to establish sufficient level of web application security. Above all, these IDSs are deployed in hardware and software infrastructure of enterprise systems and for most of small business solutions are too expensive. It follows that one of the security challenges of today is improving the security level of web application in small business sector which constantly underestimates cyber threads.

Therefore, this area is highly threatened because hackers considered it as an easier target than better secured enterprise sector. In 2012, RSA also started to talk about the increase of so-called Watering Hole Attack, a new type of targeted attack.

2.1 OWASP Top 10

According to OWASP Top 10 Vulnerabilities [4] the most common vulnerability in 2013 is still Injection [5]. All of the Top 10 Vulnerabilities are caused by web application developers, who have not followed best practices of secure development. Types of attacks from Table 1 are in general targeted directly against the web application and it does not even matter how strong firewall rulesets are or how frequent the patching mechanism is. The only effective defense is either sophisticated IDS, or web built on a secure development framework. If we are focused into the small business area, we are facing the fact, that the first option is usually not implemented due to high costs. On the other hand,

there is lot of secure development frameworks available also as an open-source.

Table 1: OWASP Top 10

Vulnerability
A1 Injection
A2 Broken Authentication and Session Management (was formerly A3)
A3 Cross-Site Scripting (XSS) (was formerly A2)
A4 Insecure Direct Object References
A5 Security Misconfiguration (was formerly A6)
A6 Sensitive Data Exposure (merged from former A7 Insecure Cryptographic Storage and former A9 Insufficient Transport Layer Protection)
A7 Missing Function Level Access Control (renamed/broadened from former A8 Failure to Restrict URL Access)
A8 Cross-Site Request Forgery (CSRF) (was formerly A5)
A9 Using Known Vulnerable Components (new but was part of former A6 – Security Misconfiguration)
A10 Unvalidated Redirects and Forwards

2.2 Watering Hole Attack

In recent year, hackers started to use a new scenario of web attack against enterprise systems. It is type of highly sophisticated targeted attack, which is misusing vulnerable web sites of various entities to defeat the stronger security of another entity.

The concept of Watering Hole Attack is similar to a predator waiting at a watering hole in a desert. The predator assumes, that the prey eventually has to come to refresh and waits for his victim. [6]

This hacker's technique uses vulnerable web sites which are considered to be visited by hacker's victims. Vulnerable web site is infected by malicious code which is able to install an exploit to victim's computer.

3 Problem Solution

The overall security of small business sector web applications should be improved, because this will lead also to higher security level of enterprise sector in the same supply chain. An effective improvement without need of high financial budget could be done using secure open-source frameworks for web application development.

PHP development framework QCubed (http://qcu.be/) was chosen for security tests of the most common vulnerability, such as injection is. QCubed is Object-Oriented PHP framework with Object Relation Mapping code generator [7]. This framework seems to be highly suitable for small business sector, thanks to low cost and rapid application development process.

By using secure development framework, the developer does not face a large number of issues and places for creating a security bug. The resulting web application should be immune especially to attack, which are based on unvalidated inputs. On the other side, some web application logic issues, such as Insecure Direct Object References, can still appear and should be carefully tested within development process.

However, above described recommendations can produce better secured web application, with any monitoring functionality, which IDS offer. In the most common small business scenario, there is a web application running on a commercial webhosting, therefore the only way to improve security is on the web application level.
The new proposed model of low cost solution is based on design of open-source web application development framework with embedded IDS. This model of web application IDS is described in Section 5.

4 Vulnerability Testing

This section describes the testing methodology, types of tested vulnerabilities and results of tests conducted on real demo web application based on QCubed framework.

4.1 Testing environment

For the testing purposes latest stable release of QCubed framework (2.2.0) was installed on local web server and demo web application were prepared. QCubed framework is a PHP webapplication development framework which allows low cost and effective webapplication development process. This framework is developed by the community and one of the main goal is the focus on security issues. Therefore the latest version of the framework is also using the third party library HTMLPurifier, a standards-compliant HTML filter library which is available in different programming languages and also in PHP [8]. The main advantage of HTMLPurifier implementation is

in highly secure user input validation. By the string validation, HTMLPurifier decomposes the string into tokens they processed as follows: The non-whitelisted elements are removed, bad practice tags are transformed, the nesting of tags is properly checked and all tags attributes are validated according to their RFCs.
Testing demo application consists of simple form with user input, which will be tested in different user input filtering modes. The testing machine runs Windows 7 OS with WAMP 2.0 software with Apache web server version 2.2.21, PHP version 5.3.8 and MySQL version 5.5.16.

4.2 User input validation testing

Tested QCubed framework is using a component based development model. For the user text inputs, there is QTextBox object dedicated. This object is used for single-line, multi-line or password input, as the instance could be changed using the property TextMode. In addition to built-in validation of the maximum string length and data type of the content, there is also HTMLPurifier implemented. In default, HTMLPurifier is not enabled, the QTextBox's property CrossScripting is set to Deny and the validation is provided using framework's core string validation. In this scenario, only simple PHP strpos function is used to search possible attack patterns such as: <applet, <embed, <style, <link, <body, <iframe, javascript:, onfocus=, onblur= etc. Other CrossScripting options are Allow (not recommended, text inputs are unsanitized) or HtmlEntities(special characters are converted to htmlentities). HTMLPurifier is possible to enable using the option CrossScripting set to HTMLPurifier value.
For the testing purpose, user input validation was set as follows: CrossScripting – Deny, HTMLPurifier and Allowfor confirmation the positive influence of attack.

4.2.1 Testing Input Data (XSS)

As a testing data, various injection examples of XSS attack from OWASP Filter Evasion Cheat Sheet were used [9].

4.2.2 Testing Results

Table 2: Results of user input XSS vulnerability

	Deny	HTML - Purifier	Allow
No Filter Evasion	Negative	Negative	**Positive**
XSS Locator 1	Negative	Negative	**Positive**
XSS Locator 2	**Positive**	Negative	**Positive**
Image XSS using the JavaScript directive	Negative	Negative	**Positive**
UTF-8 Unicode encoding	**Positive**	Negative	**Positive**
Long UTF-8 Unicode encoding without semicolons	**Positive**	Negative	**Positive**
Embedded tab	**Positive**	Negative	**Positive**
& JavaScript includes	**Positive**	Negative	**Positive**
XSS using HTML quote encapsulation	Negative	Negative	**Positive**
JavaScript link loc.	Negative	Negative	**Positive**

Results of QTextBox XSS vulnerability testing are shown in Table 2. Negative results mean that the web application is not vulnerable to XSS strings inserted to user input, whilst in case of positive, the web attack was successful.

From the testing results (Table 2) can be stated, that the simple string filtering (Deny option) fails in every case of more sophisticated attack, where special encoding is used. On the other hand, HTMLPurifier shows very satisfactory results also in case of more complex attacks.

5 Proposal model of development framework with embedded IDS

The proposed model of secure web application development framework, described in this section, is based on embedded IDS, which is working on web application level. This IDS is provided directly by framework's core files and activity of web application is stored and monitored. The stored activity data could be evaluated and one of intrusion detection methods should be applied to detect malicious behavior.

Storing and analyzing significant data from life cycle of scripts and web application should be the main goal of further research as well as

finding appropriate techniques for on-line response to any attack behavior.

5.1 Request detail storing

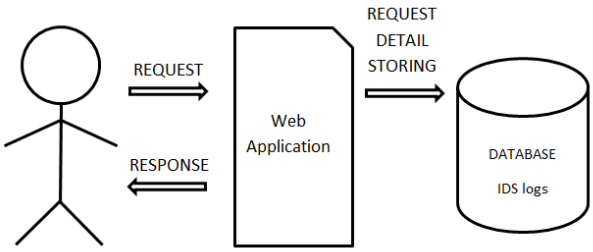

Fig. 1: Script life cycle data storing

The first phase of embedded IDS implementing deals with storing information about normal requests from client to web application. At first significant information should be identified. One request from user could contain these important parameters:

- Date and time
- Script name
- Script parameters (QCubed's Query String and Path Info)
- Number of SQL Queries done by the script
- HTTP Referer
- Client Identifier

Fig. 1 shows the principle of storing script activity data. At first client sends a request to the server to web application. Server is processed the data for return to the client via server's response. In context of QCubed script life cycle, we can use e.g. the event Form_Exit(), which is called before the end of the script, to store all of important data.

Table 3: Database table for storing script activity (simplified)

1	/qcubed_ids /category_list.php	3	15.5.2013 10:51:58	ID	http://
2	/qcubed_ids/drafts/ category_edit.php	3	15.5.2013 10:52:01	ID	http://
3	/qcubed_ids/drafts/ category_list.php	3	15.5.2013 10:52:09	ID	http://
4	/qcubed_ids/drafts/ category_edit.php	3	15.5.2013 10:52:17	ID	http:/
5	/qcubed_ids/drafts/ category_list.php	3	15.5.2013 10:52:23	ID	http://
6	/qcubed_ids/drafts/ category_edit.php	4	15.5.2013 10:52:26	ID	http://
7	/qcubed_ids/drafts/ category_list.php	3	15.5.2013 10:52:30	ID	http://
8	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:28**	ID	***NULL***
9	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:28**	ID	***NULL***
10	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:28**	ID	***NULL***
11	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:28**	ID	***NULL***
12	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:29**	ID	***NULL***
13	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:29**	ID	***NULL***
14	**/qcubed_ids/login .php**	1	**15.5.2013 10:59:29**	ID	***NULL***

Table 3 demonstrates an example of stored data which was simplified and columns Query String and Path Info were removed. As you can see, this data are higly valuable for analyse and detection of attack behavior. These types of attack can be detected using particular parameters or continuous records:

- SQL Injection (Query String and Path Info parameter, and DB query number)
- XSS (Query String and Path Info parameter)
- Automated attacks (Number of script evalutation in time)

Highlighted data in the Table 3 demonstrates also a brute force attack on the login form of web application. One of the significant factor here is a number of script execution in time. Suspicious behave can be seen also from missing the http referer record.

6 Future work

Major challenges of future research are seen in on-line detection methods. Storing important script life cycle data principle was introduced in Section 5.1 and further work will deal with the analysis of the data and proposal of detection methods. Principle of detection method should be based on an automated detection using Neural Network.

ISBN: 978-960-474-311-7

7 Conclusion

As is clearly seen from recent IT security trends, the main challenge of today is improvement of web application security level in small business sector. Recent research shows, that new form of sophisticated targeted hacker attacks are appearing. E.g. Watering Hole Attack, which is based of compromising more vulnerable small web applications to defeat stronger secured target.

One of the major problems of small business IT security is insufficient budget, which is causing operating of a vulnerable web application and missing HW and SW IDS infrastructure. Efficient solution should be based on using sufficiently secure development framework with proper input validation. In this paper framework QCubed and its options of input validation was tested. According to results from section 4.2.2, it can be stated, that the simple string filtered inputs are not immune to sophisticated injection attacks rather should be implemented validation library HTMLPurifier which provides high resistance against XSS attacks.

Section 5 describes the proposed model of secure web application development framework with embedded IDS. As a first step of the design, important data storing was implemented. Examples of the stored data are shown in Table 3.

Section 6 is focused on future work in this topic. One of the crucial parts of proactive on-line embedded IDS lies in analysis of stored data and detection of attack behavior. In future research implementation of Neural Network for attack detection should be considered.

References:

[1] Symantec Corporation. *Internet Security Threat Report 2013*. Volume 18. 2013.

[2] Ali A. Ghorbani, Wei Lu, Mahbod Tavallaee. *Network Intrusion Detection and Prevention*. Springer US, 2010.

[3] Will Gragido. *Lions at the Watering Hole – The "VOHO" Affair. Speaking of Security – The RSA Blog and Podcast* [online]. 2012 http://blogs.rsa.com/lions-at-the-watering-hole-the-voho-affair/

[4] OWASP. *OWASP Top Ten Project* [online]. 2013 https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[5] OWASP. *Top 10 2013-A1-Injection* [online]. 2013 https://www.owasp.org/index.php/Top_10_2013-A1-Injection

[6] Gavin O'Gorman, Geoff McDonald. *The Elderwood Project* [online]. 2012 http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the-elderwood-project.pdf

[7] *Welcome to Qcubed!* [online]. 2013 http://qcu.be/content/welcome

[8] HTML Purifier. *Standards-Compliant HTML Filtering* [online]. 2013 http://htmlpurifier.org/

[9] OWASP. *XSS Filter Evasion Cheat Sheet* [online]. 2013 https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet