



# SECURE PROGRAMMING

## Penetration Test Report – User blog service

### Abstract

This document presents the findings after a penetration testing on the user blog service website as part of the Secure Programming course of Fontys University of Applied Sciences.

Group D

## Contents

Introduction .....	2
First stage .....	2
Second stage. ....	6
Conclusions .....	14

## Introduction

This is a penetration testing report for the web page "User Blog Service" located initially at: <http://sepr.tigrimigri.com/sepr/>. The penetration testing was done in two stages. The first stages consisted of inputting SQL commands to test for SQL Injection, commands to test for XSS and finally finding vulnerabilities in the code itself. The second stage consisted of generating a report using OWASP Zap 2.4.3. The later stage created an issue of blocking incoming communications to the webpage therefore after communication with the client team, the website was moved by the client to <http://sepr.byethost7.com/sepr/index.php>.

## First stage

The first stage of the test did not bear any fruit.

The following SQLI commands were tried:

- **'OR '1'='1** – in order to bypass the password
- **99 and 1=1 union select version(),database() #** - in order to see the database and try to exploit that.
- **99 and 1=1 union select null,user()** – in order to see the database user.

Therefore, we couldn't bypass the login procedure or see the database structure.

After the SQL injection test we tried to do a XSS test.

Since we couldn't bypass the login, we created an account on the website. This gave us the access to the comment page.

In the comment page we tried to insert a few XSS commands in order to make the website run scripts as comments.

The following commands were tried:

- **<script>alert(document.cookie)</script>**
- **<script language="javascript">alert(document.cookie)</script>**
- **<script>alert("test")</script>**

This also didn't give any results. In figure 1 we can see the usual result of the attempts. Furthermore the " " character was blocked so we couldn't even use a string within the alert function.

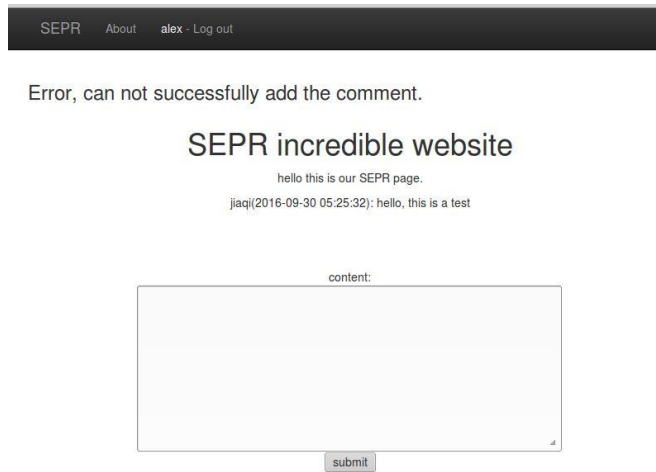
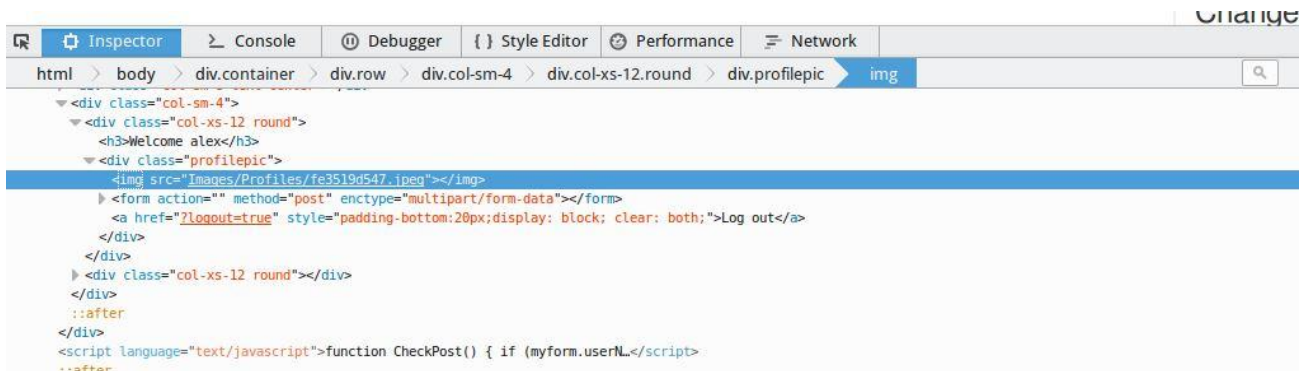
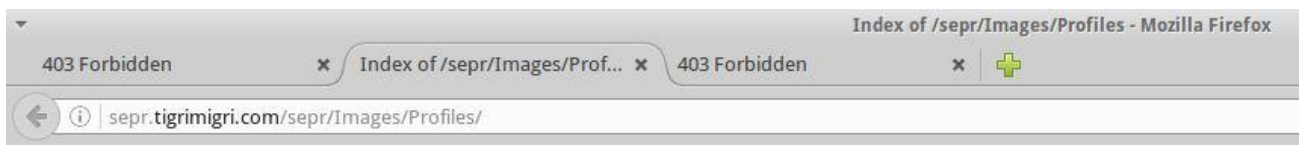


Figure 1 - Error message after XSS

After the XSS we tried to see the code of the website. In this attempt we found a few minor vulnerabilities. By uploading a picture in the upload file section and refreshing the webpage we saw in the code that the existence of a picture gave us a link to the directory where files were stored (/Images) as shown in the following picture.





## Index of /sepr/Images/Profiles

- [Parent Directory](#)
- [01a75435cc.jpg](#)
- [134a67b76f.jpg](#)
- [20c9a287e9.jpg](#)
- [22a64d8bc4.jpg](#)
- [23384c8dd6.jpg](#)
- [260f470e9f.jpg](#)
- [3689a1ad0d.jpg](#)
- [36badc1d31.jpg](#)
- [61031ee6f9.jpg](#)
- [6fcc7b019a.jpg](#)
- [8eb210ab57.jpg](#)
- [c22eb76877.jpg](#)
- [ced7f8058d.jpg](#)
- [default.jpg](#)
- [e94c2f536c.jpg](#)

Figure 3 - Actual images directory on the server

Not only did we manage to access the pictures saved in the directory we also managed to download all of them. The upload function actually generates the name of the picture and actually checks that no other file (such as .php) could not be uploaded. We however managed to upload a php file (which was a shell script b374k-shell, which would've allowed us an external access to the website), by renaming it to index.php.jpg, however the name was changed to a generated name, and given that we couldn't rename it on the server-side in order to access it.

Furthermore, we managed to find a link to a directory of php files. In the following image we can see that in the website code, there is a reference to a folder that contains a validation javascript file.

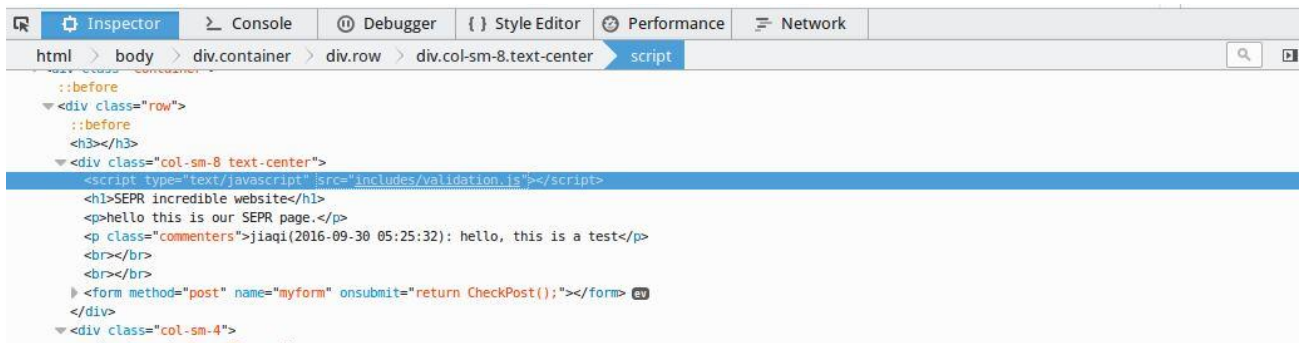
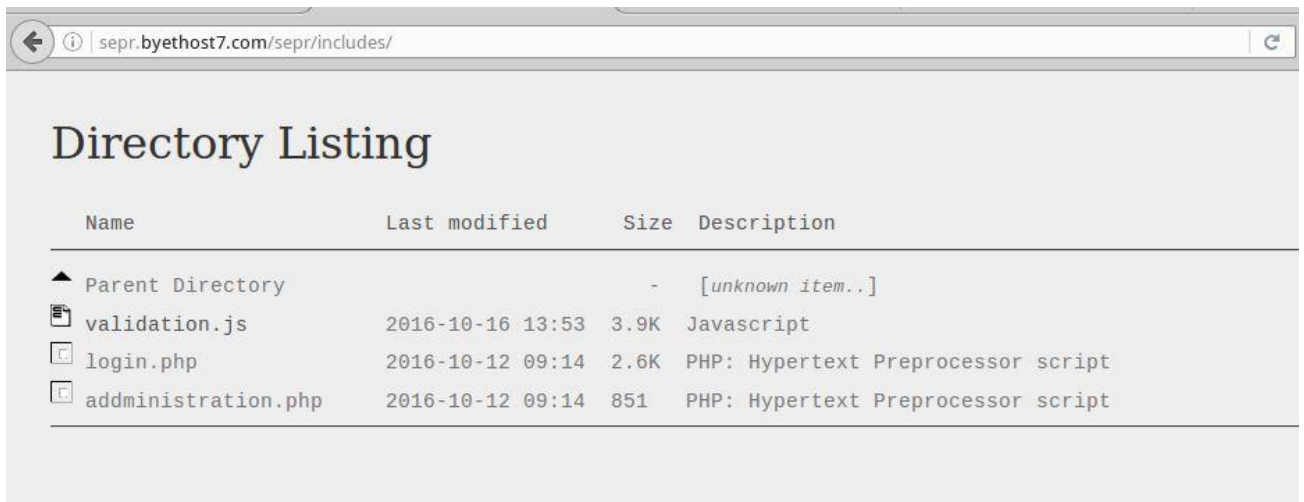


Figure 4 - Includes directory in the code

This gave us access to the following directory:



Name	Last modified	Size	Description
▲ Parent Directory		-	[unknown item..]
📄 validation.js	2016-10-16 13:53	3.9K	Javascript
📄 login.php	2016-10-12 09:14	2.6K	PHP: Hypertext Preprocessor script
📄 administration.php	2016-10-12 09:14	851	PHP: Hypertext Preprocessor script

*Figure 5 - Actual includes directory on the server (new server)*

Note: The above picture comes from the newly uploaded website, which the client team uploaded to a new host because due to our penetration testing, the initial hosting service blocked access to it.

## Second stage.

Following the failure of the above methods we decided to use a penetration testing software (OWASP Zap 2.4.3 through BackBox Linux). The software produced the following report:

### ZAP Scanning Report

#### Summary of Alerts

Risk Level	Number of Alerts
<a href="#">High</a>	1
<a href="#">Medium</a>	3
<a href="#">Low</a>	5
<a href="#">Informational</a>	0

#### Alert Detail

<b>High (Medium)</b>	<b>SQL Injection</b>
Description	SQL injection may be possible.
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	username
Attack	ZAP AND 1=1 --
URL	http://sepr.tigrimigri.com/sepr/index.php?page=home
Parameter	username
Attack	ZAP OR 1=1 --
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	submit
Attack	Change Password' OR '1'='1' --
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	submit
Attack	Change Password' AND '1'='1' --
Instances	4
Solution	<p>Do not trust client side input, even if there is client side validation in place.</p> <p>In general, type check all data on the server side.</p> <p>If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'</p> <p>If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries.</p>

	<p>If database Stored Procedures can be used, use them.</p> <p>Do <i>*not*</i> concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality!</p> <p>Do not create dynamic SQL queries using simple string concatenation.</p> <p>Escape all data received from the client.</p> <p>Apply a 'whitelist' of allowed characters, or a 'blacklist' of disallowed characters in user input.</p> <p>Apply the principle of least privilege by using the least privileged database user possible.</p> <p>In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact.</p> <p>Grant the minimum database access that is necessary for the application.</p>
Other information	<p>The page results were successfully manipulated using the boolean conditions [ZAP AND 1=1 -- ] and [ZAP AND 1=2 -- ]</p> <p>The parameter value being modified was NOT stripped from the HTML output for the purposes of the comparison</p> <p>Data was returned for the original parameter.</p> <p>The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter</p>

Reference	<p><a href="https://www.owasp.org/index.php/Top_10_2010-A1">https://www.owasp.org/index.php/Top_10_2010-A1</a></p> <p><a href="https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet">https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet</a></p>
CWE Id	89
WASC Id	19
<b>Medium (Medium)</b>	<b>X-Frame-Options Header Not Set</b>
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.

URL	<a href="http://sepr.tigrimigri.com/sepr/">http://sepr.tigrimigri.com/sepr/</a>
URL	<a href="http://sepr.tigrimigri.com/robots.txt">http://sepr.tigrimigri.com/robots.txt</a>
URL	<a href="http://sepr.tigrimigri.com/sitemap.xml">http://sepr.tigrimigri.com/sitemap.xml</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=home">http://sepr.tigrimigri.com/sepr/?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=about">http://sepr.tigrimigri.com/sepr/?page=about</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/css/style.css">http://sepr.tigrimigri.com/sepr/css/style.css</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/includes/validation.js">http://sepr.tigrimigri.com/sepr/includes/validation.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php">http://sepr.tigrimigri.com/sepr/index.php</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=about">http://sepr.tigrimigri.com/sepr/index.php?page=about</a>



Instances	10
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Other information	At "High" threshold this scanner will not alert on client or server error responses.

Reference	<a href="http://blogs.msdn.com/b/ieinternals/archive/2010/03/30/combating-clickjacking-with-x-frame-options.aspx">http://blogs.msdn.com/b/ieinternals/archive/2010/03/30/combating-clickjacking-with-x-frame-options.aspx</a>
-----------	---

<b>Medium (Medium)</b>	<b>Directory Browsing</b>
Description	It is possible to view the directory listing. Directory listing may reveal hidden scripts, include files , backup source files etc which can be accessed to read sensitive information.

URL	<a href="http://sepr.tigrimigri.com/sepr/css/">http://sepr.tigrimigri.com/sepr/css/</a>
Attack	Parent Directory
URL	<a href="http://sepr.tigrimigri.com/sepr/includes/">http://sepr.tigrimigri.com/sepr/includes/</a>
Attack	Parent Directory
Instances	2
Solution	Disable directory browsing. If this is required, make sure the listed files does not induce risks.
Reference	<a href="http://httpd.apache.org/docs/mod/core.html#options">http://httpd.apache.org/docs/mod/core.html#options</a> <a href="http://alamo.satlug.org/pipermail/satlug/2002-February/000053.html">http://alamo.satlug.org/pipermail/satlug/2002-February/000053.html</a>
CWE Id	548
WASC Id	48

<b>Medium (Low)</b>	<b>Parameter Tampering</b>
Description	Parameter manipulation caused an error page or Java stack trace to be displayed. This indicated lack of exception handling and potential areas for further exploit.

URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=">http://sepr.tigrimigri.com/sepr/index.php?page=</a>
Parameter	page
Attack	on line <b>
Instances	1
Solution	Identify the cause of the error and fix it. Do not trust client side input and enforce a tight check in the server side. Besides, catch the exception properly. Use a generic 500 error page for internal server error.
Reference	
CWE Id	472
WASC Id	20

<b>Low (Medium)</b>	<b>Cookie set without HttpOnly flag</b>
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie

will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.

URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=45af665f6a93c5594ef3e7f6fb263f36; path=
Evidence	PHPSESSID=45af665f6a93c5594ef3e7f6fb263f36; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=7edc8a0dab65e6729df17d2539a22f7e; path=
Evidence	PHPSESSID=7edc8a0dab65e6729df17d2539a22f7e; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	PHPSESSID=e8fbe56333adc3fb639dd36433caf6a5; path=
Evidence	PHPSESSID=e8fbe56333adc3fb639dd36433caf6a5; path=
URL	http://sepr.tigrimigri.com/sepr/?page=about
Parameter	PHPSESSID=8f2de5278b08f742873ccee9b424e719; path=
Evidence	PHPSESSID=8f2de5278b08f742873ccee9b424e719; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=eb93a1588e39d2206a2fc2292d6b1a9a; path=
Evidence	PHPSESSID=eb93a1588e39d2206a2fc2292d6b1a9a; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=e599de416c3f41aac4cb926690f2d84; path=
Evidence	PHPSESSID=e599de416c3f41aac4cb926690f2d84; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	PHPSESSID=d10b2a0b63e509d8e72a2925fa18dfdc; path=
Evidence	PHPSESSID=d10b2a0b63e509d8e72a2925fa18dfdc; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	PHPSESSID=793294299a9f9b6a1e14d68e62bc0ef2; path=
Evidence	PHPSESSID=793294299a9f9b6a1e14d68e62bc0ef2; path=
URL	http://sepr.tigrimigri.com/sepr/?logout=true
Parameter	PHPSESSID=d0808e218c8fa240ca5825797a2c1a2a; path=
Evidence	PHPSESSID=d0808e218c8fa240ca5825797a2c1a2a; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=5123746bb5045542d85a02ded56986c8; path=
Evidence	PHPSESSID=5123746bb5045542d85a02ded56986c8; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=98e2000dc3620235e7bffe3360487fe0; path=
Evidence	PHPSESSID=98e2000dc3620235e7bffe3360487fe0; path=
URL	http://sepr.tigrimigri.com/sepr/
Parameter	PHPSESSID=2ae3936c45f8e6840e0035fb69cb20d7; path=
Evidence	PHPSESSID=2ae3936c45f8e6840e0035fb69cb20d7; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	PHPSESSID=4639874064fa3fcb93211f2ed3ee026e; path=
Evidence	PHPSESSID=4639874064fa3fcb93211f2ed3ee026e; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	PHPSESSID=712cf71371153200cb6cf35dee73d483; path=
Evidence	PHPSESSID=712cf71371153200cb6cf35dee73d483; path=
URL	http://sepr.tigrimigri.com/sepr/?page=home

Parameter	PHPSESSID=586168632ceab965426d519f93ed00a5; path=/
Evidence	PHPSESSID=586168632ceab965426d519f93ed00a5; path=/
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	PHPSESSID=b1789d4aa021df55a36784321b91a4cd; path=/
Evidence	PHPSESSID=b1789d4aa021df55a36784321b91a4cd; path=/
URL	http://sepr.tigrimigri.com/sepr/index.php?page=home
Parameter	PHPSESSID=9dedd6d88cf474fdf795b5c0aeb8ec17; path=/
Evidence	PHPSESSID=9dedd6d88cf474fdf795b5c0aeb8ec17; path=/
URL	http://sepr.tigrimigri.com/sepr/index.php?page=about
Parameter	PHPSESSID=287ddc5938200fcb01255d40bf629fb3; path=/
Evidence	PHPSESSID=287ddc5938200fcb01255d40bf629fb3; path=/
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	PHPSESSID=1cb507004454c8d4cf817ca3e50458e7; path=/
Evidence	PHPSESSID=1cb507004454c8d4cf817ca3e50458e7; path=/
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	PHPSESSID=4828eff8ab684718c10a2a17d9e26aa7; path=/
Evidence	PHPSESSID=4828eff8ab684718c10a2a17d9e26aa7; path=/
Instances	26
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	www.owasp.org/index.php/HttpOnly
WASC Id	13
<b>Low (Medium)</b>	<b>Cross-Domain JavaScript Source File Inclusion</b>
Description	The page at the following URL includes one or more script files from a third-party domain

URL	http://sepr.tigrimigri.com/sepr/
Parameter	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
Evidence	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
URL	http://sepr.tigrimigri.com/sepr/
Parameter	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
Evidence	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
Evidence	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
URL	http://sepr.tigrimigri.com/sepr/?page=home
Parameter	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
Evidence	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
URL	http://sepr.tigrimigri.com/sepr/?page=about
Parameter	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
Evidence	https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js
URL	http://sepr.tigrimigri.com/sepr/?page=about
Parameter	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
Evidence	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js
URL	http://sepr.tigrimigri.com/sepr/index.php
Parameter	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js

Evidence	<a href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php">http://sepr.tigrimigri.com/sepr/index.php</a>
Parameter	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
Evidence	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
Parameter	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
Evidence	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
Parameter	<a href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js</a>
Evidence	<a href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=about">http://sepr.tigrimigri.com/sepr/index.php?page=about</a>
Parameter	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
Evidence	<a href="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js">https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.0/jquery.min.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=about">http://sepr.tigrimigri.com/sepr/index.php?page=about</a>
Parameter	<a href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js</a>
Evidence	<a href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js</a>
Instances	12
Solution	Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application
Reference	

<b>Low (Medium)</b>	<b>Web Browser XSS Protection Not Enabled</b>
Description	Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server

URL	<a href="http://sepr.tigrimigri.com/sepr/">http://sepr.tigrimigri.com/sepr/</a>
URL	<a href="http://sepr.tigrimigri.com/robots.txt">http://sepr.tigrimigri.com/robots.txt</a>
URL	<a href="http://sepr.tigrimigri.com/sitemap.xml">http://sepr.tigrimigri.com/sitemap.xml</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=home">http://sepr.tigrimigri.com/sepr/?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=about">http://sepr.tigrimigri.com/sepr/?page=about</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/css/style.css">http://sepr.tigrimigri.com/sepr/css/style.css</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/includes/validation.js">http://sepr.tigrimigri.com/sepr/includes/validation.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php">http://sepr.tigrimigri.com/sepr/index.php</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=about">http://sepr.tigrimigri.com/sepr/index.php?page=about</a>
Instances	10
Solution	Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.
Other information	<p>The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it:</p> <p>X-XSS-Protection: 1; mode=block</p> <p>X-XSS-Protection: 1; report=http://www.example.com/xss</p>

	<p>The following values would disable it:</p> <p>X-XSS-Protection: 0</p> <p>The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit).</p> <p>Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length).</p>
--	---

Reference	<a href="https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet">https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet</a> <a href="https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/">https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/</a>
CWE Id	933
WASC Id	14

<b>Low (Medium)</b>	<b>Password Autocomplete in browser</b>
Description	AUTOCOMPLETE attribute is not disabled in HTML FORM/INPUT element containing password type input. Passwords may be stored in browsers and retrieved.

URL	<a href="http://sepr.tigrimigri.com/sepr/">http://sepr.tigrimigri.com/sepr/</a>
Parameter	input
Evidence	<input name="password" type="password" class="form-control" id="pwd">
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=home">http://sepr.tigrimigri.com/sepr/?page=home</a>
Parameter	input
Evidence	<input name="password" type="password" class="form-control" id="pwd">
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php">http://sepr.tigrimigri.com/sepr/index.php</a>
Parameter	input
Evidence	<input name="password" type="password" class="form-control" id="pwd">
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
Parameter	input
Evidence	<input name="password" type="password" class="form-control" id="pwd">
Instances	4
Solution	Turn off AUTOCOMPLETE attribute in form or individual input elements containing password by using AUTOCOMPLETE='OFF'
Reference	<a href="http://msdn.microsoft.com/library/default.asp?url=/workshop/author/forms/autocomplete_ovr.asp">http://msdn.microsoft.com/library/default.asp?url=/workshop/author/forms/autocomplete_ovr.asp</a>
CWE Id	525

<b>Low (Medium)</b>	<b>X-Content-Type-Options Header Missing</b>
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and

	legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	<a href="http://sepr.tigrimigri.com/sepr/">http://sepr.tigrimigri.com/sepr/</a>
URL	<a href="http://sepr.tigrimigri.com/robots.txt">http://sepr.tigrimigri.com/robots.txt</a>
URL	<a href="http://sepr.tigrimigri.com/sitemap.xml">http://sepr.tigrimigri.com/sitemap.xml</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=home">http://sepr.tigrimigri.com/sepr/?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/?page=about">http://sepr.tigrimigri.com/sepr/?page=about</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/css/style.css">http://sepr.tigrimigri.com/sepr/css/style.css</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/includes/validation.js">http://sepr.tigrimigri.com/sepr/includes/validation.js</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php">http://sepr.tigrimigri.com/sepr/index.php</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=home">http://sepr.tigrimigri.com/sepr/index.php?page=home</a>
URL	<a href="http://sepr.tigrimigri.com/sepr/index.php?page=about">http://sepr.tigrimigri.com/sepr/index.php?page=about</a>
Instances	10
Solution	<p>Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.</p> <p>If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.</p>
Other information	<p>This issue still applies to error type pages (401, 403, 500, etc) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.</p> <p>At "High" threshold this scanner will not alert on client or server error responses.</p>
Reference	<a href="http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx">http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx</a> <a href="https://www.owasp.org/index.php/List_of_useful_HTTP_headers">https://www.owasp.org/index.php/List_of_useful_HTTP_headers</a>
WASC Id	15

As you can see from the above report there was a small SQLi vulnerability. However, we didn't manage to recreate it, because the website host stopped allowing the access to the page.

## Conclusions

As you can see from the above mentioned, the user blog website is quite secure. Given our intermediate knowledge of hacking techniques, we couldn't manage to access much information on the website that we shouldn't have accessed. The only disadvantage we found was the fact that we could browse directories which we shouldn't have. Given a better skill set in hacking techniques probably we could've exploit that in our advantage. As far as the SQL injection is concerned we feel that the client's website is pretty secure from intermediate hackers. As you saw in the second part, the OWASP Zap only managed to find one issue when it comes to SQL injection.

Our recommendation would be to change the access to such directories as includes and images. Furthermore, we would suggest that when uploading a picture for a user, a separate directory be generated for that user (in example Images/Profiles/<username>).