Web Vulnerability Assessment -redacted- Portal

Submitted to: -redacted-

Submitted by:

Technical/
Business POC:

Erik Santana

Date: June 14, 2022

Table of Contents

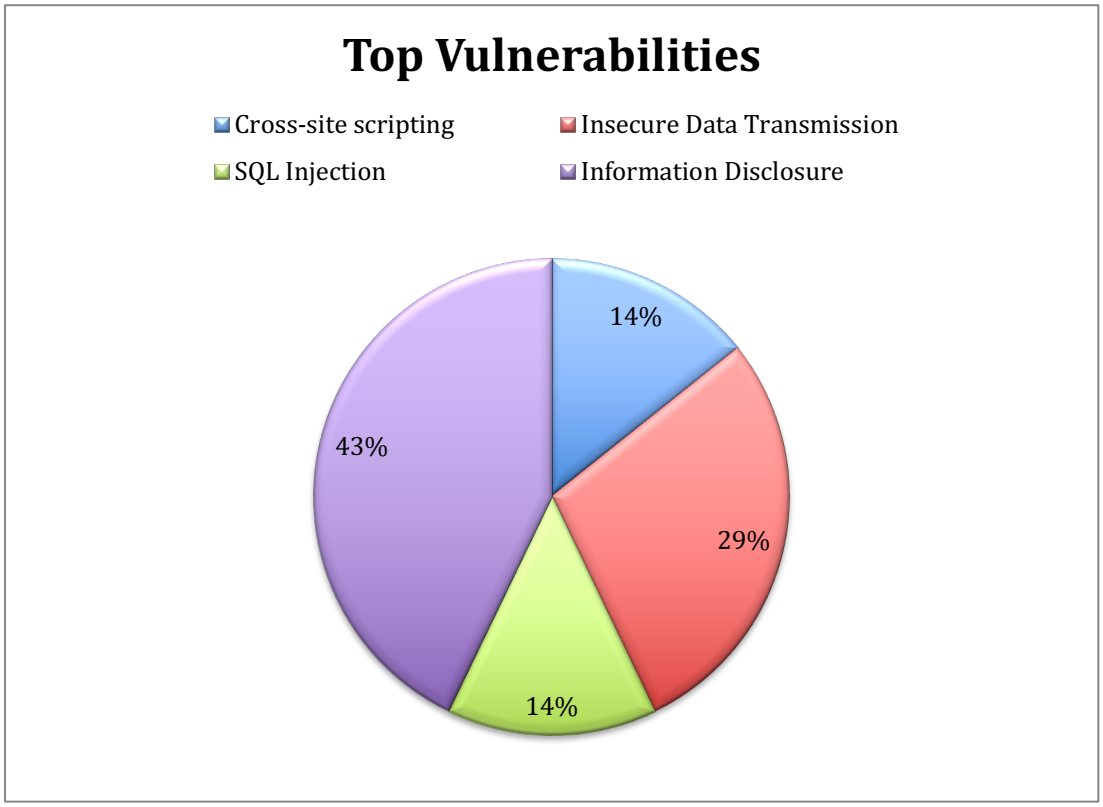
Executive Summary

Within this report are the results of the vulnerability scan for the -redacted- web portal: https://.-redacted-.com. The purpose of this report is to identify any security issues within the engagement scope of the analysis.

Engagement scope:

-redacted- was tasked to complete a vulnerability test and perform code analysis from the site with URL: https://.-redacted-.com. What follows is a summary of the encountered vulnerabilities/exposures during the penetration test:

Top Vulnerabilities



Of particular attention, the -redacted- development team should handle the following issues promptly:

- Provide input validation to prevent the persistent Cross-Site Scripting vulnerability in the ----- comment section
- Verify the SSL configuration parameters to ensure secure transmission

Vulnerability Report

Vulnerability definitions

High Risk Vulnerabilities ⍰⍰ ⍰⍰	These vulnerabilities present issues that could result in system compromise, data loss and/or information disclosure.
Medium Risk Vulnerabilities	These vulnerabilities present issues that could result in system compromise indirectly or in combination to other exploits.
Low Risk Vulnerabilities	These vulnerabilities present issues that could result in system compromise in combination to other exploits.

Vulnerability: Persistent Cross-Site Scripting

Risk: High

Details:

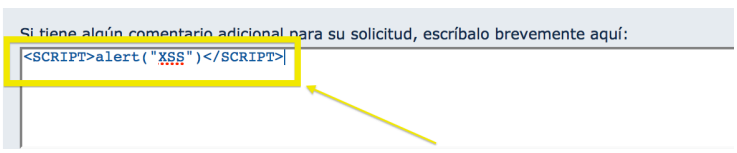
The comment section in the ------ is vulnerable to persistent cross-site scripting. It occurs when the data provided by the attacker is not validated and saved by the server. Then permanently displayed on "normal" pages returned to other users in the course of regular browsing, without proper HTML escaping. This vulnerability would allow an attacker to execute unauthorized code, which will affect each and every user who accesses the same profile.

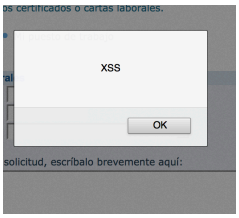
Recommendation:

The simplest form of XSS protection would be to pass all external data through a filter which will remove dangerous keywords, such as the <SCRIPT> tag, JavaScript commands, CSS styles and other dangerous HTML markup.

Proof of concept:

https://.-redacted-.com/servlet/CheckSecurity/JSP/sse_g3/sse_g3_p22_Otro.jsp?estado=31





Vulnerability: Cookie not marked as secure

Risk: High

Details:

Cookies that are not marked as secure, can be transmitted by the browser via HTTP. This could lead an attacker to intercept and hijack a victim's session. An attacker could force the victim to perform an HTTP request and steal the cookie.

Recommendation:

Mark all cookies within the application as secure.

Proof of concept:

HTTP/1.1 200 OK
Connection: Keep-Alive
Date: Wed, 27 Nov 2013 15:14:16 GMT
Transfer-Encoding: chunked
Server: Microsoft-IIS/7.5
Set-Cookie: JSESSIONID=7e3064d8b6f9797996624b406124623f6f72;path=/
Vary: Accept-Encoding
Content-Encoding:
Content-Type: text/html; charset=ISO-8859-1

Vulnerability: Insecure Transportation Security Protocol Supported (SSLv2)

Risk: High

Details:

Insecure transportation security protocol (SSLv2) is supported by your web server. SSLv2 has several flaws. For example, your secure traffic can be observed when you

have established it over SSLv2.

Recommendation:

For Microsoft IIS, you should make some changes on the system registry.
Click Start, click Run, type regedt32 or type regedit, and then click OK.
In Registry Editor, locate the following registry key:
HKey_Local_Machine\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL2\
Locate a key named "Server." If it doesn't exist, create it.
Under the "Server" key, locate a DWORD value named "Enabled." If it doesn't exist, create it and set it to "0".

Vulnerability: Blind SQL Injection

Risk: Medium

Details:

Using the following value SQL injection *"-redacted-redacted-.com AND 1=1 –"* to replace the *"HOST"* parameter in the login request, the page results were successfully manipulated and it presented a generic "non-standard" error page. This type of attack is very complex in nature and would require advanced technical knowledge to have a successful outcome thus it is rated as a medium risk vulnerability.

Recommendation:

Best practice to defend against Blind SQL Injection attacks is to make sure all servers, services and application are up to date and patched. Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. Grant the minimum database access that is necessary for the application.

Proof of Concept:

Request

GET https://-redacted-redacted-.com/sse_generico/espanol/generico_login.jsp?estado=0 HTTP/1.1
Host: -redacted-redacted-.com' AND '1'='1' --
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;)
Pragma: no-cache
Cache-control: no-cache
Content-Type: application/x-www-form-urlencoded

Content-length: 0

Vulnerability: Secure page can be cached in browser

Risk: Medium

Details:

Secure page can be cached in browser. Cache control is not set in HTTP header nor HTML header. Sensitive content can be recovered from browser storage. This would allow an attacker to recover sensitive content from browser storage.

Recommendation:

The best way is to set HTTP header with: 'Pragma: No-cache' and 'Cache-control: No-cache'.

Alternatively, this can be set in the HTML header by:

<META HTTP-EQUIV='Pragma' CONTENT='no-cache'>

<META HTTP-EQUIV='Cache-Control' CONTENT='no-cache'>

Note: Some browsers may have problem using this method.

Proof of concept:

Request

https://-redacted-redacted-
.com/sse_generico/espanol/generico_login.jsp?estado=0 HTTP/1.1
Host: -redacted-redacted-.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;)
Pragma: no-cache
Cache-control: no-cache
Content-Type: application/x-www-form-urlencoded
Content-length: 0

Response

HTTP/1.1 200 OK
Connection: Keep-Alive
Date: Thu, 05 Dec 2013 18:59:36 GMT
Content-Type: text/html; charset=ISO-8859-1
Server: Microsoft-IIS/7.5

Set-Cookie: JSESSIONID=7e306bb691b4382e0c2a7d8427637151251e;path=/

Vulnerability: Version Disclosure (IIS)

Risk: Low

Details:

From the HTTP header response the following data was extracted -- Microsoft-IIS/7.5 This information can help an attacker gain a greater understanding of the systems in use and potentially develop further attacks targeted at the specific version of IIS. An attacker can use the disclosed information to gather specific security vulnerabilities for the version identified.

Recommendation:

Use the UrlScan ultility from Microsoft to alter the "AlternateServerName=" under "options" entry.

http://www.iis.net/learn/extensions/working-with-urlscan/urlscan-3-reference

Proof of concept:

Header
Web server
HTTP response code
Server Address
connection
content-encoding
content-type
date
server
transfer-encoding
vary
Time to contact web server