Web Application
Penetration Testing Report

Product Name: Security Shepherd

Product Version: v3.1

Test Completion: 07/04/2019

Lead Penetration Tester: Trisha Gupta

Prepared For: Project Lead of Security Shepherd

Consultant Information

Name: Trisha Gupta
Email: trisha.gupta@ucdconnect.ie
UID: 18208276
Location: UCD, Belfield, Dublin 4
Manager: Mark Scanlon
Manager email: mark.scanlon@ucd.ie

Table of Contents

Executive Summary

Lead Tester: Trisha Gupta
Number of days testing: 15 - 20 days
Test Start Date: February 4, 2019
Test End Date: March 31, 2019

Project Information

Application Name: Security Shepherd Application
Version: v3.1
Release Date: Oct 11, 2018
Project Contact: Mark Scanlon

Findings:

- XSS (3 vulnerabilities)
- SQL Injection (2 vulnerabilities)
- CSRF (2 vulnerabilities)
- Broken Authentication (2 vulnerabilities)
- Broken Authorization (Failure to Restrict URL Access) (3 vulnerabilities)
- Poor data Validation (2 vulnerabilities)
- Insecure Cryptographic Storage (2 vulnerabilities)

OWASP Top 10:

- XSS (Rank 7),
- SQL Injection (Rank 1)
- Broken Authentication (Rank 2)

Total Defects: 16

Severity	#Defects
Critical	0
High	3
Medium	4

Low	0

Scope

Issues	Vulnerabilities	Time frame for testing	Time frame for report
Broken Authorization - Failure To Restrict URL Access [CWE-285]	Lesson, Challenge 1, Challenge 2	3 days	2 days
XSS [CWE - 79]	Lesson, Challenge 1, Challenge 2	3 days	2 days
Cross Site Request Forgery [CWE - 352]	Lesson, Challenge 1	2 days	2 days
Injection [CWE - 564]	Lesson, Challenge 3	2 days	2 days
Broken Authentication [CWE- 287]	Lesson, Challenge 1	1 day	3 days
Poor Data Validation [CWE-20]	Challenge 1, Challenge 2	2 days	3 days
Insecure Cryptographic Storage [CWE-327]	Lesson, Challenge 1	1 day	3 days

List of User Roles in the System

Vulnerability	Username	Password	List of URL
XSS - Cross Site Scripting 2	admin	trish	https://192.168.56.103/index.jsp
XSS - Cross Site Scripting 1	admin	trish	https://192.168.56.103/index.jsp
Cross Site Scripting Lesson	admin	trish	https://192.168.56.103/index.jsp
Cross Site Request Forgery - Lesson	admin	trish	https://192.168.56.103/index.jsp
Cross Site Request Forgery - Challenge 1	Tester	123456789	https://192.168.56.103/index.jsp
SQL Injection - Lesson	admin	trish	https://192.168.56.103/index.jsp
SQL Injection - Challenge 3	admin	trish	https://192.168.56.103/index.jsp
Broken Authentication - Lesson	admin	trish	https://192.168.56.103/index.jsp
Broken Authentication - Challenge 1	admin	trish	https://192.168.56.103/index.jsp
Failure to Restrict URL Access Lesson	admin	trish	https://192.168.56.103/index.jsp
Failure To Restrict URL Access Challenge 1	admin	trish	https://192.168.56.103/index.jsp
Failure To Restrict URL Access Challenge 1	admin	trish	https://192.168.56.103/index.jsp

Poor Data Validation - Challenge 1	admin	trish	https://192.168.56.103/index.jsp
Poor Data Validation - Challenge 2	admin	trish	https://192.168.56.103/index.jsp
Insecure Cryptographic Storage - Lesson	admin	trish	https://192.168.56.103/index.jsp
Insecure Cryptographic Storage - Challenge 1	admin	trish	https://192.168.56.103/index.jsp

Test Cases

Issues	OWASP Testing Guide v4
Broken Authorization - Failure To Restrict URL Access [CWE-285]	OWASP-AZ-001
XSS [CWE - 79]	OTG-INPVAL-001(reflected), OTG-INPVAL-002 (stored)
Cross Site Request Forgery [CWE - 352]	OTG-SESS-005
Injection [CWE - 564]	OTG-INPVAL-005
Broken Authentication [CWE- 287]	OTG-AUTHN-001
Poor Data Validation [CWE-20]	OTG-INPVAL-003
Insecure Cryptographic Storage [CWE-327]	OTG-CRYPST-001

Findings

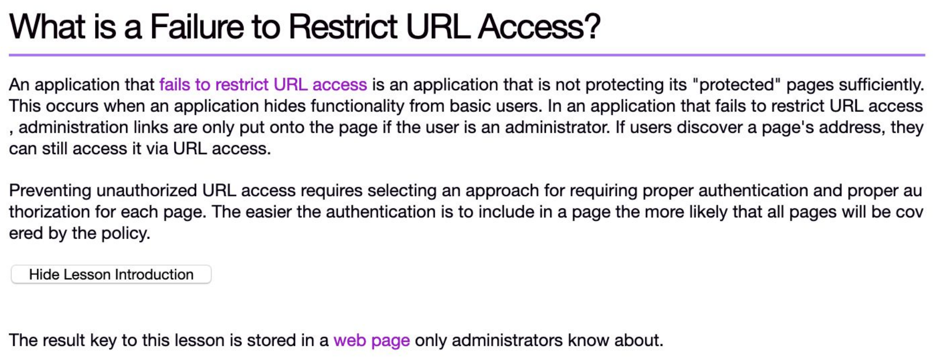
I. High - Broken Authorization [CWE-285]

The software does not perform or incorrectly performs an authorization check when an actor attempts to access a resource or perform an action.
An application that hides some functionality from its non privileged or basic users may have the vulnerability - Failure To Restrict URL Access. Only in the case that the user is an administrator, are the administration links put onto the page. Somehow, if basic users find out about the vulnerability in the administration page's address, they can be able to access the links using URL access.
In order to prevent URL access which is unauthorized, an approach for proper authorization and authentication is crucial to be selected.

a) Failure To Restrict URL Access Lesson [CWE-285]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to Lessons -> Failure To Restrict URL Access
6. Notice that the words "web page" are highlighted. Clicking on them does nothing when we hover. View the source code. Right click on the words and choose 'Inspect Element'.

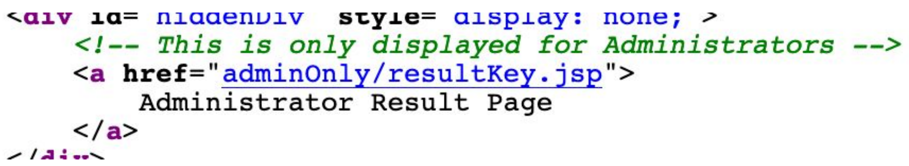

What is a Failure to Restrict URL Access?

An application that fails to restrict URL access is an application that is not protecting its "protected" pages sufficiently. This occurs when an application hides functionality from basic users. In an application that fails to restrict URL access , administration links are only put onto the page if the user is an administrator. If users discover a page's address, they can still access it via URL access.

Preventing unauthorized URL access requires selecting an approach for requiring proper authentication and proper au thorization for each page. The easier the authentication is to include in a page the more likely that all pages will be cov ered by the policy.

The result key to this lesson is stored in a web page only administrators know about.

7. Observe “This is only for administrators” and click on that. It will give you the highlighted key.



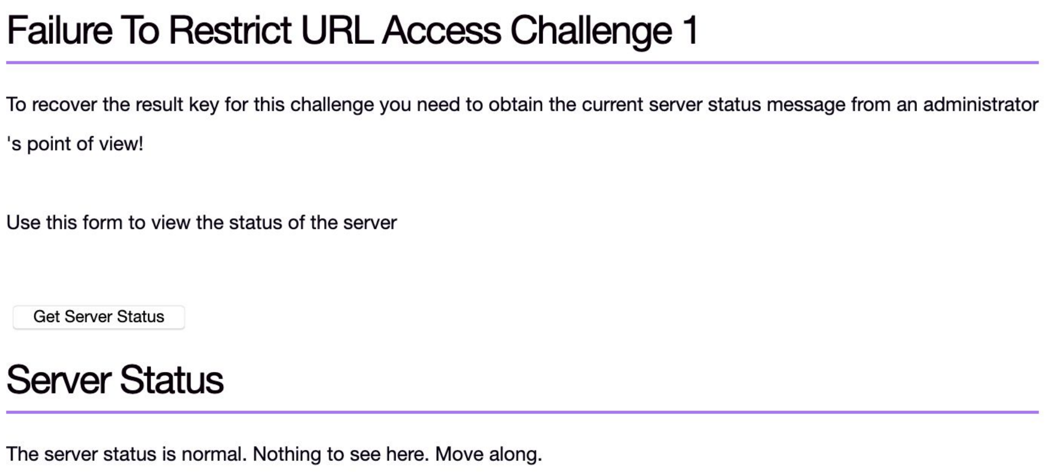
```
<div id="hiddenDiv" style="display: none;">
    <!-- This is only displayed for Administrators -->
    <a href="adminOnly/resultKey.jsp">
        Administrator Result Page
    </a>
</div>
```

8. Enter the key for successful submission of lesson.

b) Failure To Restrict URL Access Challenge 1 [CWE-285]

Steps to reproduce

1) Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2) Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3) Go to Security Shepherd https://192.168.56.103/index.jsp
4) Confirm that Burp can see and capture requests and turn off intercept in Burp
5) Go to challenges -> Failure to Restrict URL Access 1
6) Click on Get Server Status. We get something as below.



Failure To Restrict URL Access Challenge 1

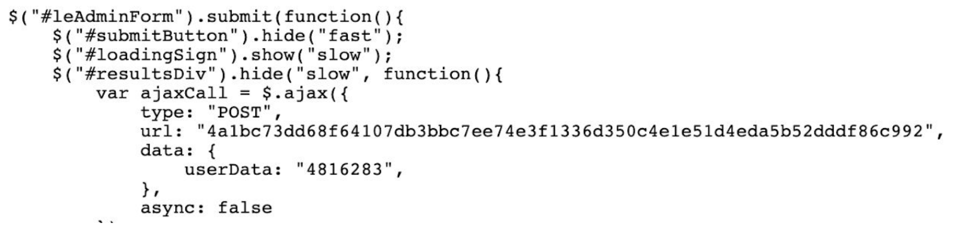
To recover the result key for this challenge you need to obtain the current server status message from an administrator 's point of view!

Use this form to view the status of the server

Server Status

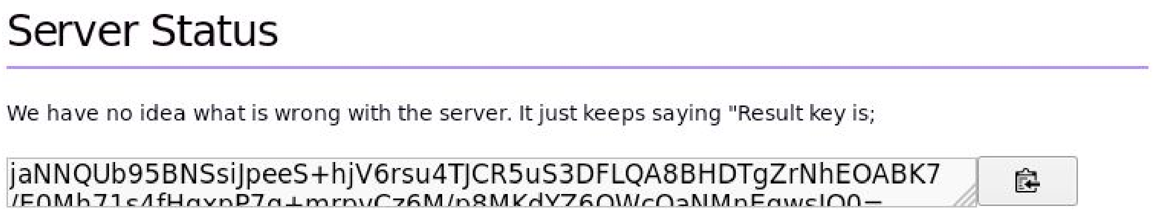
The server status is normal. Nothing to see here. Move along.

7) Check source code. There is a **Admin Form** and we also have the **URL** for the Admin Form.



```
$("#leAdminForm").submit(function(){
    $("#submitButton").hide("fast");
    $("#loadingSign").show("slow");
    $("#resultsDiv").hide("slow", function(){
        var ajaxCall = $.ajax({
            type: "POST",
            url: "4a1bc73dd68f64107db3bbc7ee74e3f1336d350c4e1e51d4eda5b52dddf86c992",
            data: {
                userData: "4816283",
            },
            async: false
```

8) Turn on intercept on BurpSuite and use it to send a post request using the URL. We get this on Security Shepherd now:



Server Status

We have no idea what is wrong with the server. It just keeps saying "Result key is;

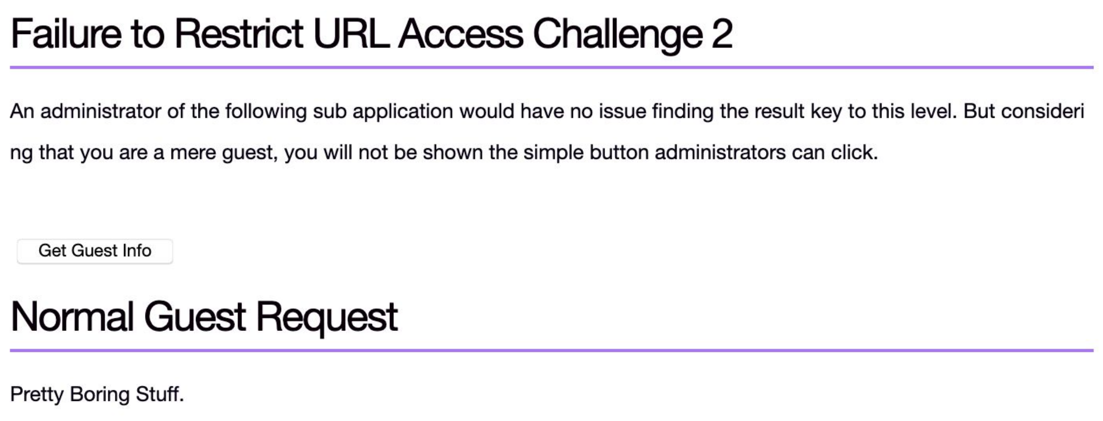
```
jaNNQUb95BNSsiJpeeS+hjV6rsu4TJCR5uS3DFLQA8BHDTgZrNhEOABK7
```

9) Submit the key and challenge solved.

c) Failure To Restrict URL Access Challenge 2[CWE-285]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to challenges -> Failure to Restrict URL Access 2
6. Click on Guest Info after flipping on the intercept in Burp:



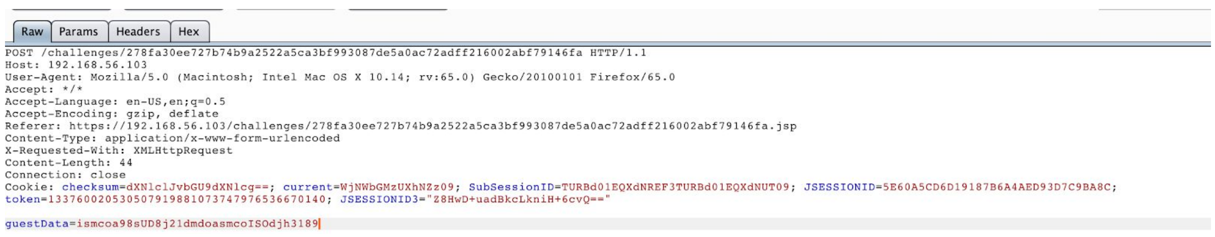
Failure to Restrict URL Access Challenge 2

An administrator of the following sub application would have no issue finding the result key to this level. But consideri
ng that you are a mere guest, you will not be shown the simple button administrators can click.

Normal Guest Request

Pretty Boring Stuff.

7. Check the HTTP request sent by intercepting on Burp when we click on it.



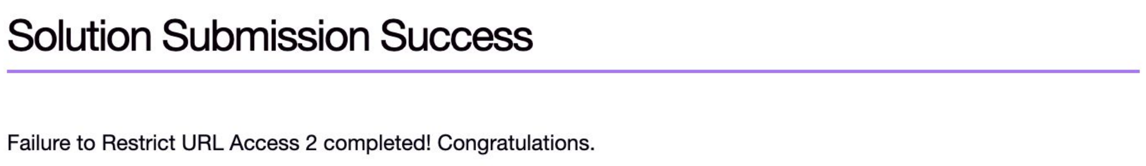
```
POST /challenges/278fa30ee727b74b9a2522a5ca3bf993087de5a0ac72adff216002abf79146fa HTTP/1.1
Host: 192.168.56.103
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://192.168.56.103/challenges/278fa30ee727b74b9a2522a5ca3bf993087de5a0ac72adff216002abf79146fa.jsp
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Content-Length: 44
Connection: close
Cookie: checksum=dXNlclJvbGU9dXNlcg==; current=WjNWbGMzUXhNZz09; SubSessionID=TURBd01EQXdNREF3TURBd01EQXdNUT09; JSESSIONID=5E60A5CD6D19187B6A4AED93D7C9BA8C;
token=133760020530507919881073747976536670140; JSESSIONID3="Z8HwD+uadBkcLkniH+6cvQ=="

guestData=ismcoa98sUD8j21dmdoasmcoISOdjh3189
```

8. Check the source code of the page to find "**278fa30ee727b74b9a2522a5ca3bf993087de5a0ac72adff216002abf79146fahghghmin|adminData|youAreAnAdminOfAwesomenessWoopWoop**" in the eval() function.

9. Use these information to replace guestData with adminData with the above name, and the URL we found to make a POST request to server to replace the guestData part above.

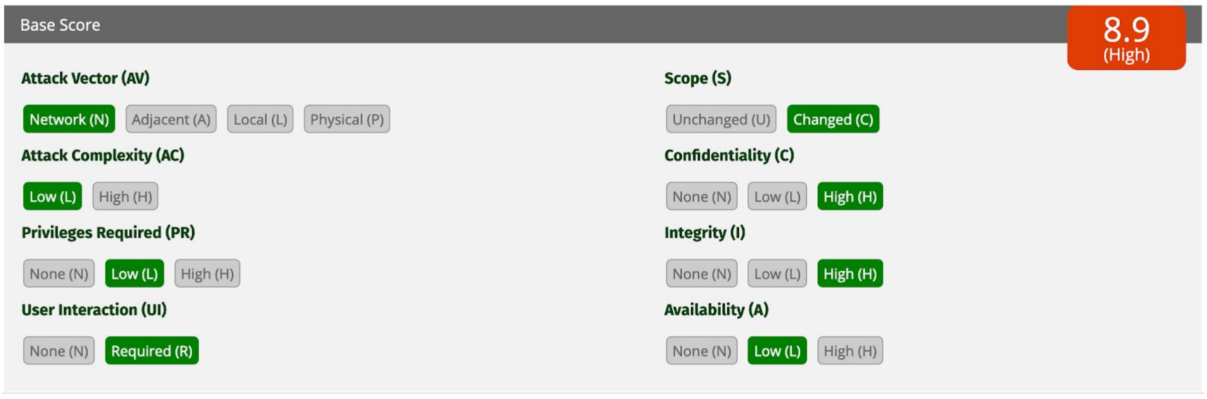
9. Challenge solved!



Solution Submission Success

Failure to Restrict URL Access 2 completed! Congratulations.

CVSS Score:



Base Score 8.9 (High)

Metric	Options
Attack Vector (AV)	[✓ Network (N)] Adjacent (A) Local (L) Physical (P)
Attack Complexity (AC)	[✓ Low (L)] High (H)
Privileges Required (PR)	None (N) [✓ Low (L)] High (H)
User Interaction (UI)	None (N) [✓ Required (R)]
Scope (S)	Unchanged (U) [✓ Changed (C)]
Confidentiality (C)	None (N) Low (L) [✓ High (H)]
Integrity (I)	None (N) Low (L) [✓ High (H)]
Availability (A)	None (N) [✓ Low (L)] High (H)

Recommended Mitigation -

In order to prevent broken authorization by URL access needs us to ensure proper authorization and authentication for our pages:

- In order to decrease effort needed to maintain the policies for authorization and authentication, they must be made role based.
- In order to decrease hard coded policy aspects, we must make configurable policies.
- Default access should be denied and only explicit access must be granted to specific users for web pages by enforcement mechanism.
- In order to allow access, we must ensure that the conditions are proper given the page is part of workflow.

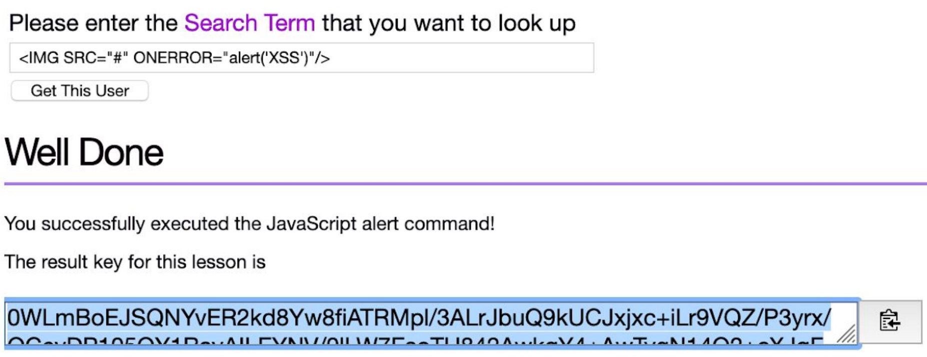
II. Medium - XSS [CWE -79]

This vulnerability takes place in the occasion that untrusted data in a web browser is used by a certain application with not enough validation. The browser executes the script when it interprets the page if this untrusted data holds a client side script. Attackers tend to use this XSS attack - one of the most widespread vulnerability in web applications - to hack user sessions by running scripts on the user's browser. It has potential to redirect useir to risky websites.

a) Cross Site Scripting Lesson [CWE - 79]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to Lessons -> Cross Site Scripting
6. In the search box, try putting a random name, say user. It says no information is found.
7. Try putting the XSS attack factor (eg: **<IMG SRC="#" ONERROR="alert('XSS')"/>)**


Please enter the Search Term that you want to look up

<IMG SRC="#" ONERROR="alert('XSS')"/>

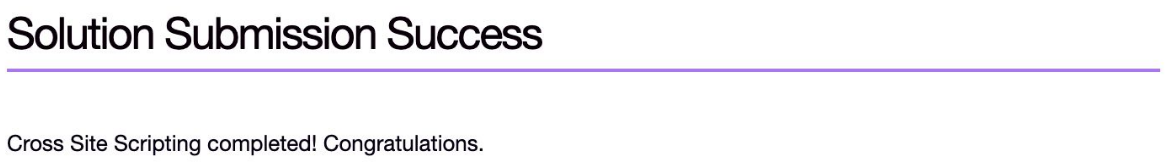
Well Done

You successfully executed the JavaScript alert command!

The result key for this lesson is

0WLmBoEJSQNYvER2kd8Yw8fiATRMpl/3ALrJbuQ9kUCJxjxc+iLr9VQZ/P3yrx/

8. Copy the key and enter above to complete this lesson.

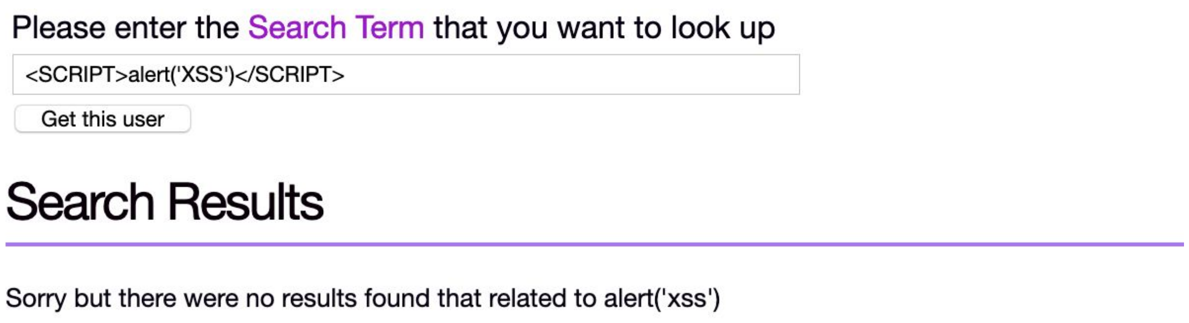

Solution Submission Success

Cross Site Scripting completed! Congratulations.

b) Cross Site Scripting Challenge 1 [CWE - 79]

Steps to reproduce

1. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
2. Go to Security Shepherd https://192.168.56.103/index.jsp
3. Go to Challenges -> XSS Challenge 1
4. Try the first payload script in the search box


Please enter the Search Term that you want to look up

<SCRIPT>alert('XSS')</SCRIPT>

Search Results

Sorry but there were no results found that related to alert('xss')

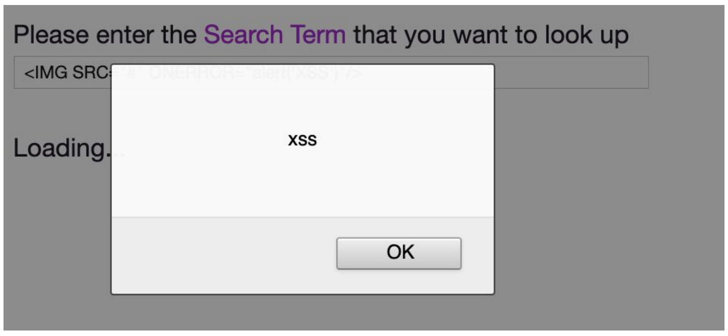
5. Observe how the title says “Find a XSS vulnerability in the following form”

6. Inspect the element “form” to find out what form is being talked about.

7. It says javascript is needed.

8. Try the second script given in lesson - <IMG SRC="#" ONERROR="alert('XSS')"/>

9. Since “alert” is being used here, the javascript attribute, the challenge gets solved.
10. It shows us this box:

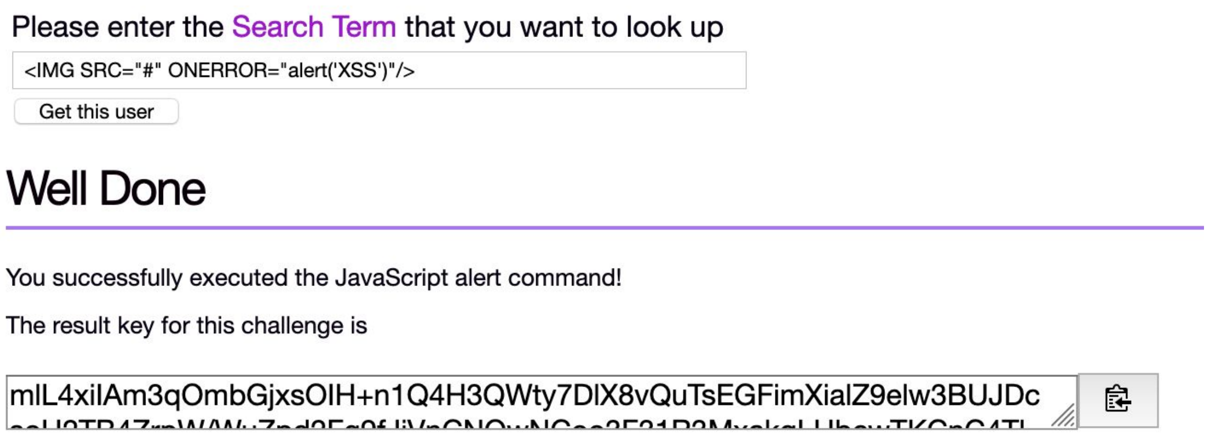

Please enter the Search Term that you want to look up

<IMG SRC="#" ONERROR="alert('XSS')"/>

Loading...

xss

11. Clicking on “OK” gives us the key which solves our challenge:



Please enter the Search Term that you want to look up

```
<IMG SRC="#" ONERROR="alert('XSS')"/>
```

Well Done

You successfully executed the JavaScript alert command!

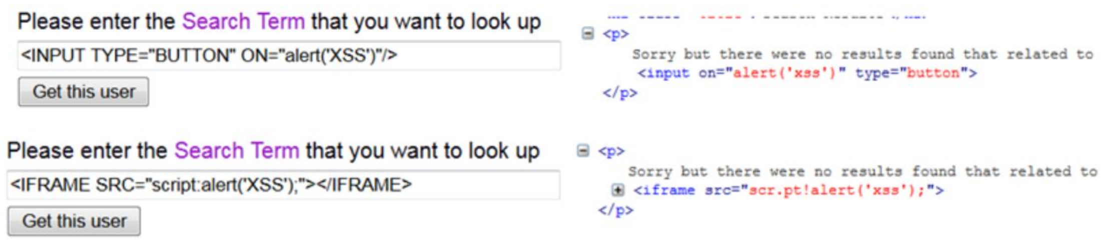
The result key for this challenge is

```
mlL4xilAm3qOmbGjxsOIH+n1Q4H3QWty7DlX8vQuTsEGFimXialZ9elw3BUJDc
```

c) Cross Site Scripting Challenge 2 [CWE - 79]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to Challenges -> XSS - Cross Site Scripting Challenge 2
6. Try the three attack vectors given in the XSS - Cross Site Scripting lesson in the box.
7. Notice their HTTP Response in their frame source code with help from BurpSuite by turning intercept on.
8. As we notice, onclick is o.ick; onerror is o.err; javascript: is by javascr.pt!
9. Check if "on" or "script" itself are also replaced



Please enter the Search Term that you want to look up

```
<INPUT TYPE="BUTTON" ON="alert('XSS')"/>
```

```
<p>
    Sorry but there were no results found that related to
    <input on="alert('xss')" type="button">
</p>
```

Please enter the Search Term that you want to look up

```
<IFRAME SRC="script:alert('XSS');"></IFRAME>
```

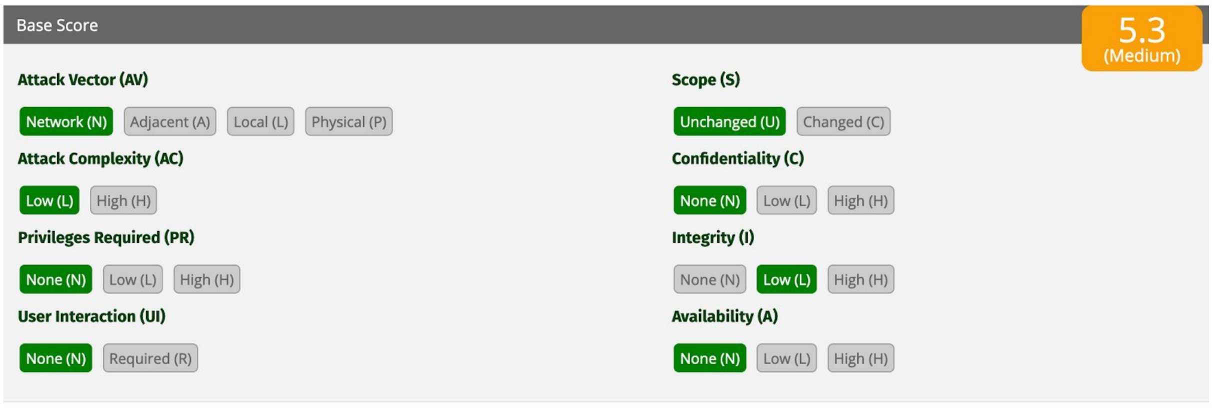
```
<p>
    Sorry but there were no results found that related to
    <iframe src="scr.pt!alert('xss');">
</p>
```

10. Go to w3school's website to check by trial and error which on is not blocked. Notice onSelect is not blocked.

11. Use this as search input - <IMG SRC="#" ONSELECT="alert('XSS')"/>.

12. Solution Key submission is successful!

CVSS SCORE:


Base Score

5.3 (Medium)

Attack Vector (AV): [✓ Network (N)] Adjacent (A) Local (L) Physical (P)

Attack Complexity (AC): [✓ Low (L)] High (H)

Privileges Required (PR): [✓ None (N)] Low (L) High (H)

User Interaction (UI): [✓ None (N)] Required (R)

Scope (S): [✓ Unchanged (U)] Changed (C)

Confidentiality (C): [✓ None (N)] Low (L) High (H)

Integrity (I): None (N) [✓ Low (L)] High (H)

Availability (A): [✓ None (N)] Low (L) High (H)

Recommended Mitigation -

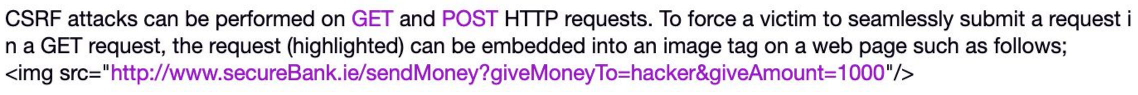
- Ensure to not insert untrusted data on web pages
- Flipping off the HTTP TRACE support on all web servers to prevent attacker from stealing cookies using JavaScript even when cookies are disabled.
- Prior to the insertion of untrusted data into HTML Content, use HTML Escape, attribute escape, javascript escape, css escape, url escape.
- Sanitization of HTML Markup with an appropriate Library

III. Medium - Cross Site Request Forgery [CWE -352]

This type of vulnerability sends a forged HTTP request from the victim’s browser using session cookie thereby deceiving the victim to interact with the application they are using. CSRF attacks occur in the case that the application is unable to make sure that it is the user using it and not some middleman. If used at an administrator, CSRF attack may have severe consequences to data protection and forgery.

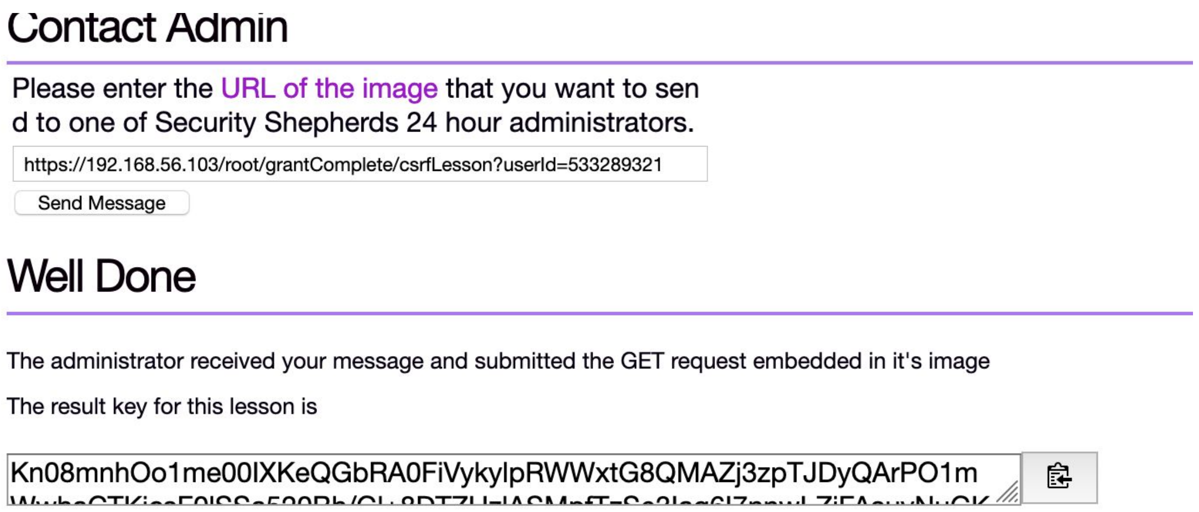
a) Cross Site Request Forgery Lesson [CWE - 352]

Steps To Reproduce:

1. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
2. Go to Security Shepherd https://192.168.56.103/index.jsp
3. Go to Lessons: Cross Site Request Forgery.
4. Notice the example provided:
5. 

CSRF attacks can be performed on GET and POST HTTP requests. To force a victim to seamlessly submit a request i
n a GET request, the request (highlighted) can be embedded into an image tag on a web page such as follows;
<img src="http://www.secureBank.ie/sendMoney?giveMoneyTo=hacker&giveAmount=1000"/>

6. Use this example to create your own URL starting with: https://192.168.56.103 followed by the GET request: /root/grantComplete/csrfLesson?userId=exampleID
7. Finally replace the “example id” with id given, i.e.: 533289321.
8. Enter the complete url i.e. https://192.168.56.103/root/grantComplete/csrfLesson?userId=533289321



Contact Admin

Please enter the URL of the image that you want to sen
d to one of Security Shepherds 24 hour administrators.

https://192.168.56.103/root/grantComplete/csrfLesson?userId=533289321

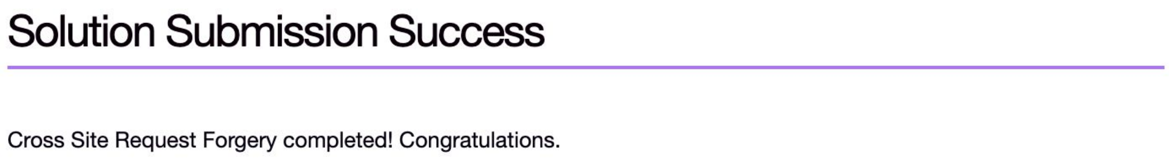
Well Done

The administrator received your message and submitted the GET request embedded in it's image

The result key for this lesson is

Kn08mnhOo1me00IXKeQGbRA0FiVykylpRWWxtG8QMAZj3zpTJDyQArPO1m
[illegible]

9. Use the key to complete this lesson:

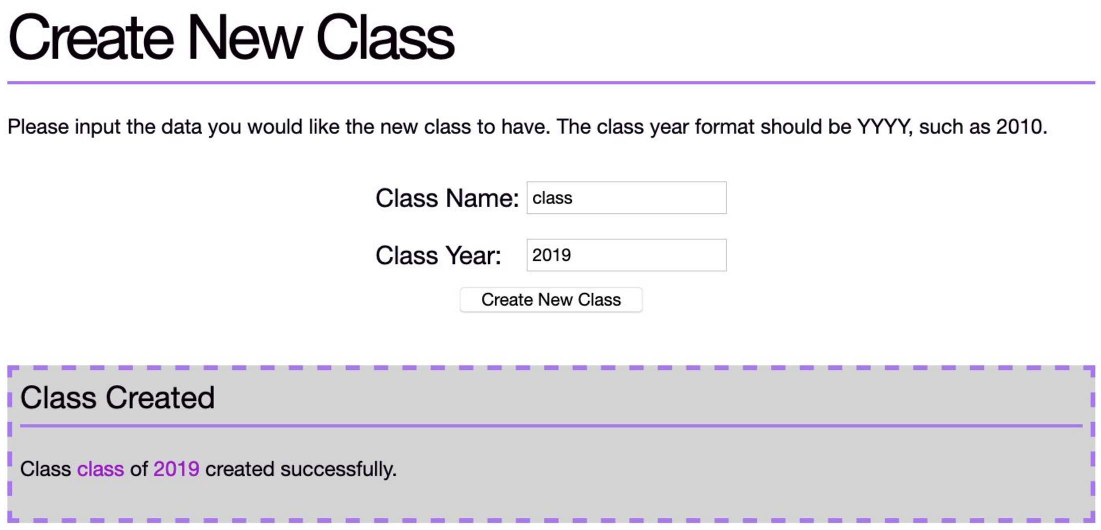

Solution Submission Success

Cross Site Request Forgery completed! Congratulations.

b) Cross Site Request Forgery Challenge 1 [CWE - 352]

Steps To Reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Move to Challenges - CSRF Challenge 1
6. We need to create a class to solve this challenge, go to “Admin” Tab followed by “User Management” followed by “Create Class”.


Create New Class

Please input the data you would like the new class to have. The class year format should be YYYY, such as 2010.

Class Name: class

Class Year: 2019

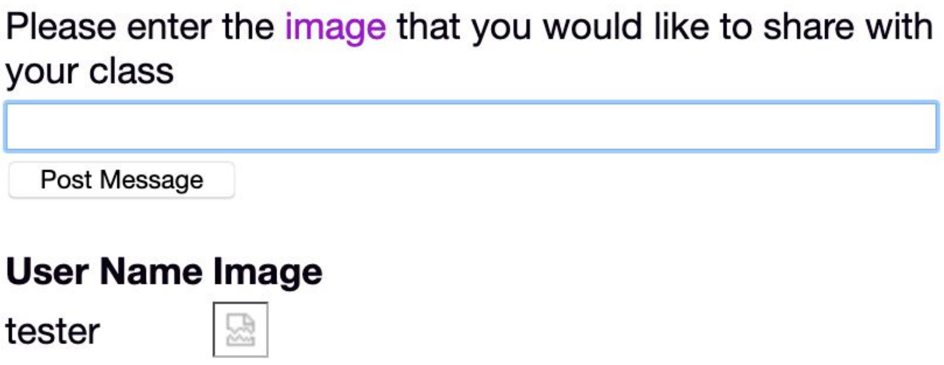
Class Created

Class class of 2019 created successfully.

7. Now, add minimum 2 players. Enter any details in the add player tab - username: tester, password:123456789. Second player username: testers, password:123456789.

8. Logout, and now login with the credentials of one of the added players (tester).

9. Notice that this is now accessible:

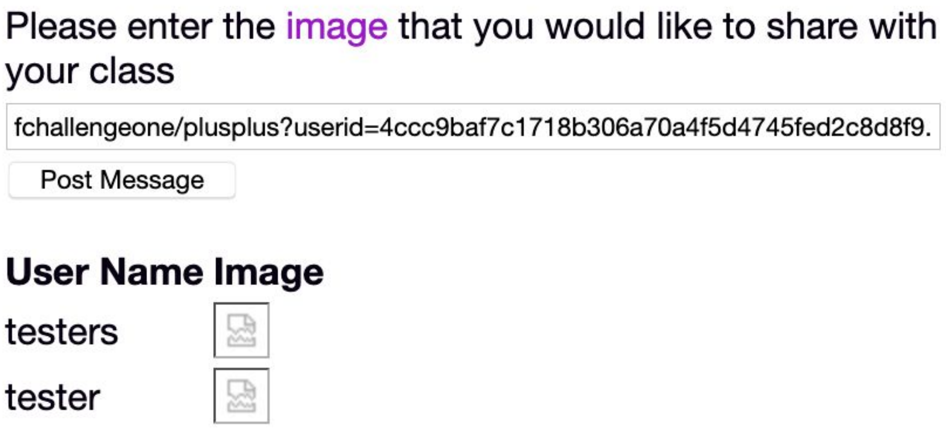

Please enter the image that you would like to share with your class

Post Message

User Name Image

tester

10. Generate the URL as https://192.168.56.103/user/csrfchallengeone/plusplus?userid=637e8d2e65542fe82fe6da3b0356bc0865b0b791 where the “exampleID” in “GET” was replaced by the given id.
11. Now, logout and login as the second user (testers) and repeat the step 14.
12. You will notice this:


Please enter the image that you would like to share with your class

fchallengeone/plusplus?userid=4ccc9baf7c1718b306a70a4f5d4745fed2c8d8f9.

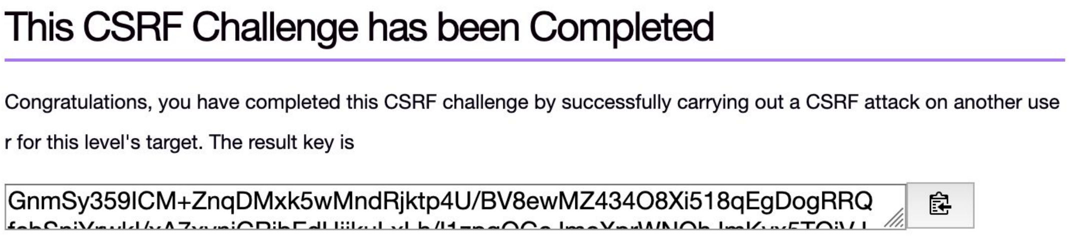
Post Message

User Name Image

testers

tester

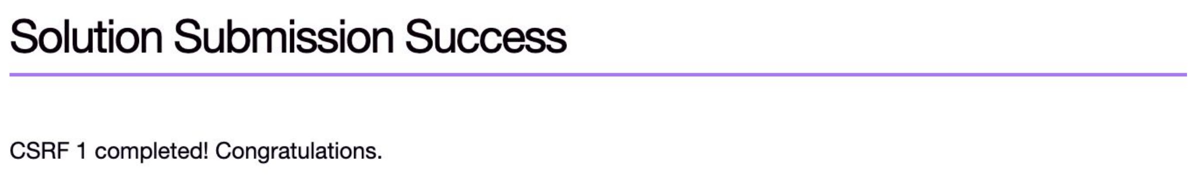
13. Logout again and when we login as the first user (tester) again, and click on this challenge:


This CSRF Challenge has been Completed

Congratulations, you have completed this CSRF challenge by successfully carrying out a CSRF attack on another use
r for this level's target. The result key is

GnmSy359ICM+ZnqDMxk5wMndRjktp4U/BV8ewMZ434O8Xi518qEgDogRRQ

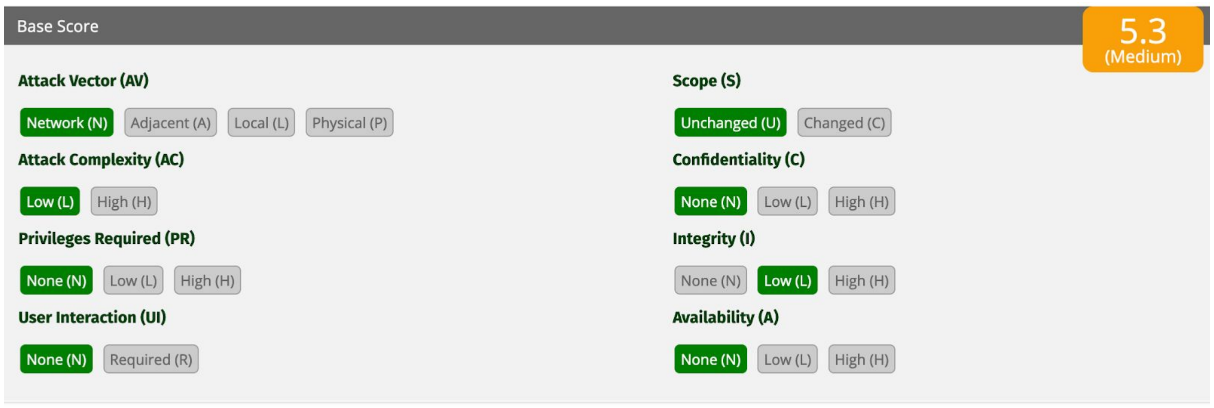
14. Copy the key above to complete the challenge:



Solution Submission Success

CSRF 1 completed! Congratulations.

CVSS Score:



Base Score — 5.3 (Medium)

Attack Vector (AV): [✓ Network (N)] Adjacent (A) Local (L) Physical (P)

Attack Complexity (AC): [✓ Low (L)] High (H)

Privileges Required (PR): [✓ None (N)] Low (L) High (H)

User Interaction (UI): [✓ None (N)] Required (R)

Scope (S): [✓ Unchanged (U)] Changed (C)

Confidentiality (C): [✓ None (N)] Low (L) High (H)

Integrity (I): None (N) [✓ Low (L)] High (H)

Availability (A): [✓ None (N)] Low (L) High (H)

Recommended Mitigations -

- To strengthen the architecture or a design, using a trusted framework or library that prevent this vulnerability to occur e.g. OWASP CSRFGuard
- Make sure that application does not have any Cross site Scripting issues as it can bypass to cause CSRF.
- In the event that a dangerous operation is performed by a user, a confirmation request be sent to make sure of the user’s intentions.
- Making use of the "double-submitted cookie" method by Felten and Zeller. This method states that each time a user browses a website, it must produce a pseudorandom value which is saved as cookie on his device. Then each time the site receives a post request, we must ensure it is coming from the same cookie value. This method requires JavaScript.
- Barring the use of GET requests that trigger a state change.
- HTTP Referer header should be originating from expected pages.

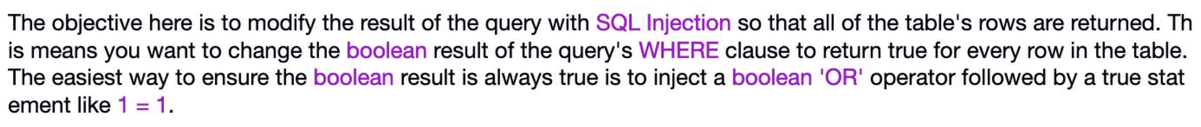
IV. Medium - SQL Injection [CWE -564]

This vulnerability occurs in the case that when an interpreter receives malware or hostile data in the form of a query or command. This may deceive him to execute unwanted commands or even access data which is unauthorized. These attacks are quite severe and can be exploited to compromise the confidentiality of a system by getting hold of its information. It can thus affect system's integrity by modifying the data. They can be initiated when someone who is interacting with the system uses data passed to the application. The data can be interpreted from the code itself as the parameters get incremented to the string which is passed to the SQL server.

a) Injection Lesson [CWE - 564]

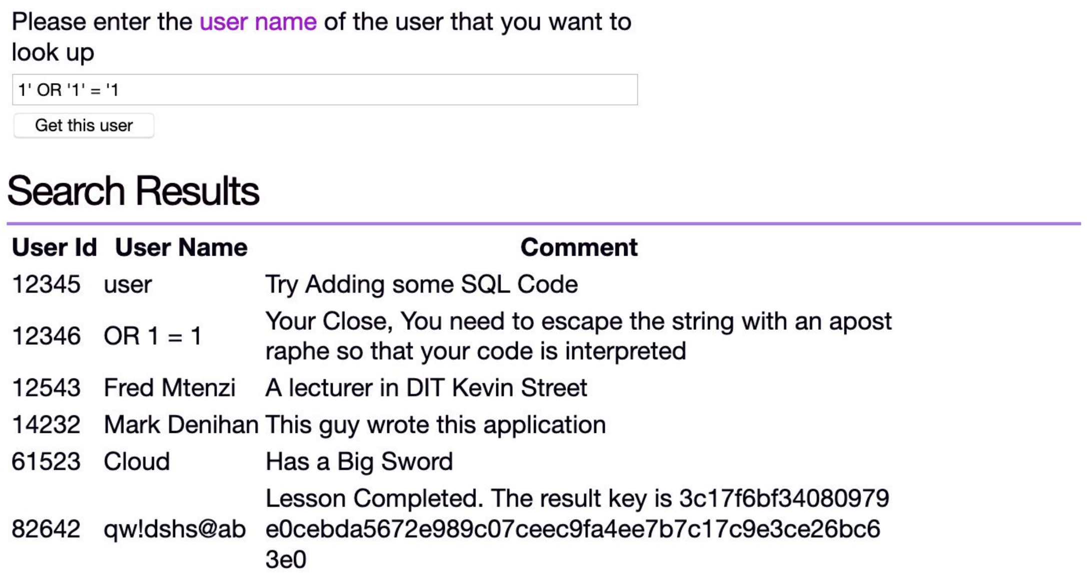
Steps to reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to lessons -> SQL Injection
6. Notice - "Our purpose is to change the [boolean]() result of the query's [WHERE]() clause to return true for every row in the table"



The objective here is to modify the result of the query with SQL Injection so that all of the table's rows are returned. Th
is means you want to change the boolean result of the query's WHERE clause to return true for every row in the table.
The easiest way to ensure the boolean result is always true is to inject a boolean 'OR' operator followed by a true stat
ement like 1 = 1.

7. Enter 1' OR '1' = '1 in the search box.


Please enter the user name of the user that you want to look up

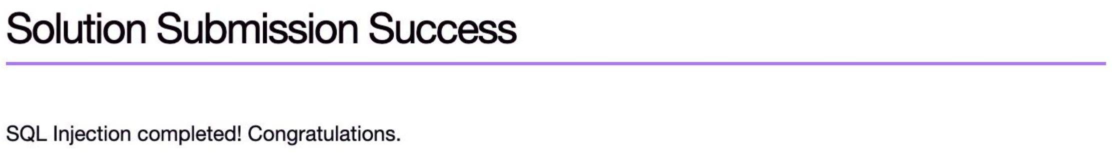
```
1' OR '1' = '1
```

Get this user

Search Results

User Id	User Name	Comment
12345	user	Try Adding some SQL Code
12346	OR 1 = 1	Your Close, You need to escape the string with an apost raphe so that your code is interpreted
12543	Fred Mtenzi	A lecturer in DIT Kevin Street
14232	Mark Denihan	This guy wrote this application
61523	Cloud	Has a Big Sword
82642	qw!dshs@ab	Lesson Completed. The result key is 3c17f6bf34080979 e0cebda5672e989c07ceec9fa4ee7b7c17c9e3ce26bc6 3e0

8. Enter the solution key to finish this lesson.

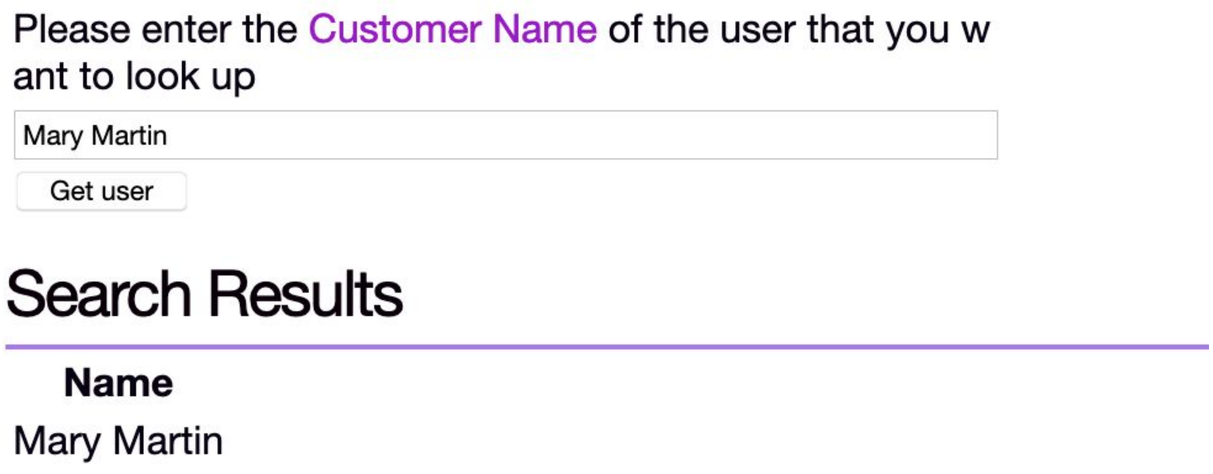

Solution Submission Success

SQL Injection completed! Congratulations.

b) Injection Challenge 3 [CWE - 564]

Steps to reproduce:

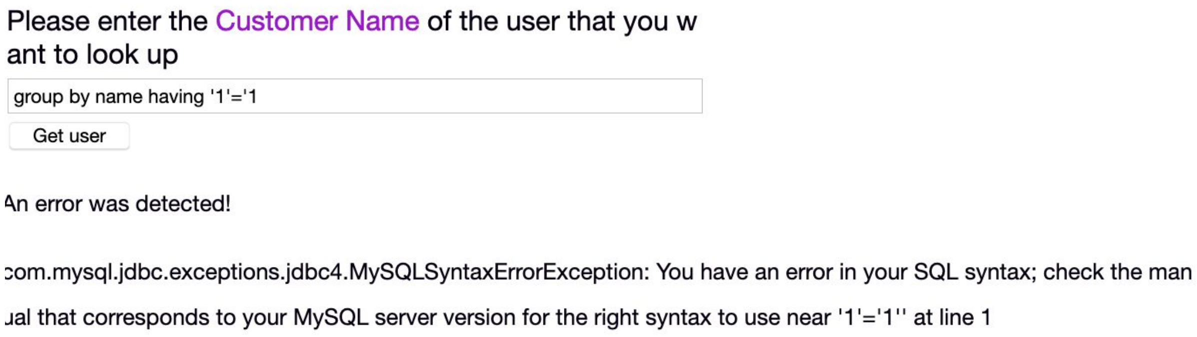
1. Go to Security Shepherd https://192.168.56.103/index.jspand login
2. Go to Challenges -> Injection -> SQL Injection Challenge Three
3. Our objective is to get the credit card number of customer name. Try inputting the name - Mary Martin in Search Box:


Please enter the Customer Name of the user that you want to look up
Mary Martin
Get user

Search Results

Name
Mary Martin

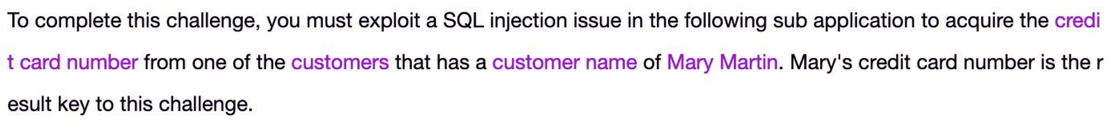
4. Use the “GROUP BY” query - group by name having '1'='1


Please enter the Customer Name of the user that you want to look up
group by name having '1'='1
Get user

An error was detected!

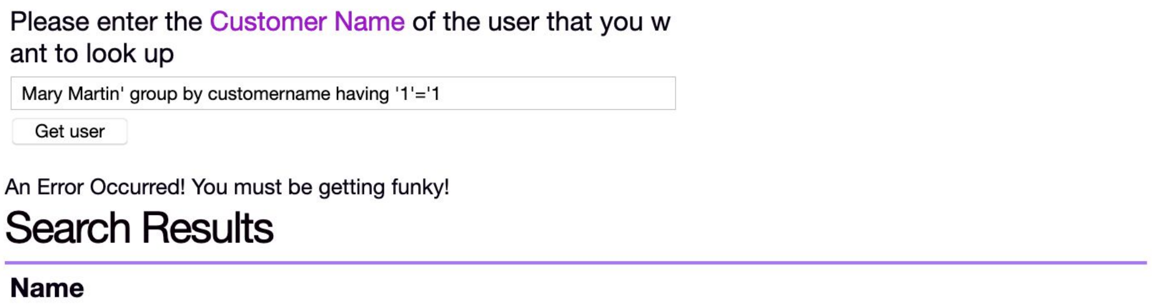
```
com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '1'='1'' at line 1
```

5. Observe the given challenge and notice the highlighted words, we will need to use them:



To complete this challenge, you must exploit a SQL injection issue in the following sub application to acquire the credit card number from one of the customers that has a customer name of Mary Martin. Mary's credit card number is the result key to this challenge.

6. To make the use of boolean values, try Mary Martin' group by customername having '1'='1


Please enter the Customer Name of the user that you w ant to look up

Mary Martin' group by customername having '1'='1

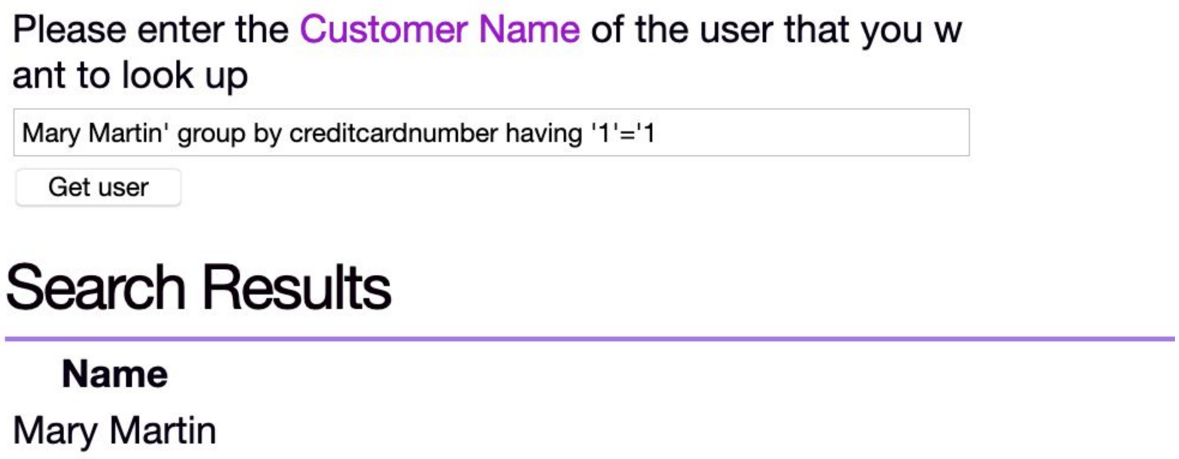
Get user

An Error Occurred! You must be getting funky!

Search Results

Name

7. Since we need credit card number, try - Mary Martin' group by creditcardnumber having '1'='1


Please enter the Customer Name of the user that you w ant to look up

Mary Martin' group by creditcardnumber having '1'='1

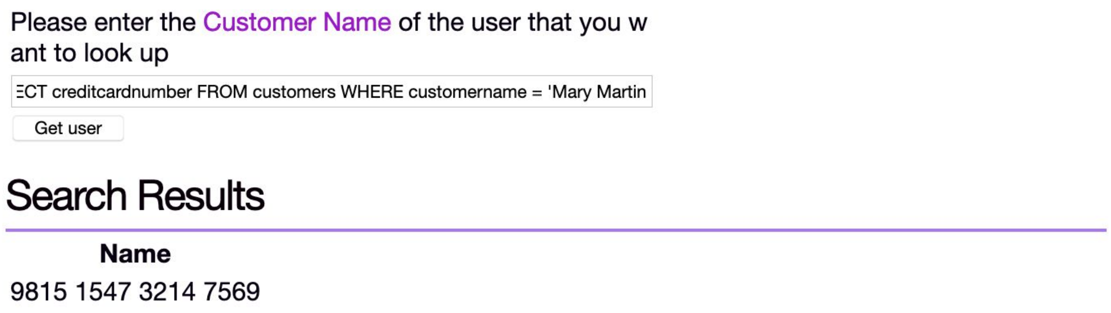
Get user

Search Results

Name

Mary Martin

8. It does give the correct name, hence we realize the need to UNION these two queries into - Mary Martin' UNION SELECT creditcardnumber FROM customers WHERE customername = 'Mary Martin


Please enter the Customer Name of the user that you w ant to look up

ECT creditcardnumber FROM customers WHERE customername = 'Mary Martin

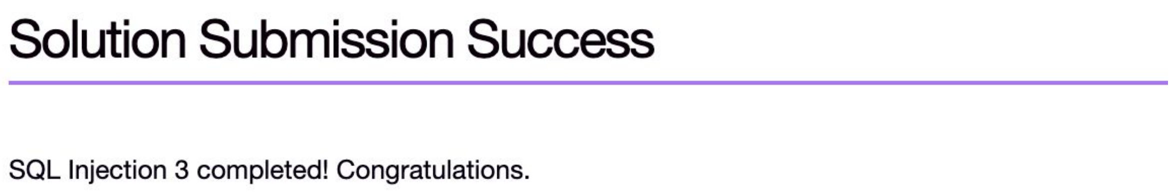
Get user

Search Results

Name

9815 1547 3214 7569

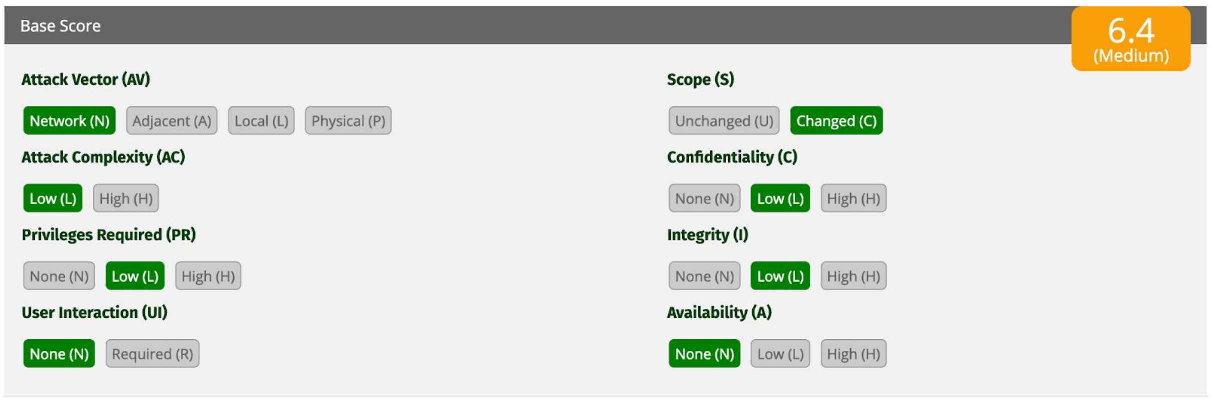
9. Copy the number into submission key to solve the challenge.



Solution Submission Success

SQL Injection 3 completed! Congratulations.

CVSS Score:



Base Score — 6.4 (Medium)

Attack Vector (AV)	[✓ Network (N)] Adjacent (A) Local (L) Physical (P)
Attack Complexity (AC)	[✓ Low (L)] High (H)
Privileges Required (PR)	None (N) [✓ Low (L)] High (H)
User Interaction (UI)	[✓ None (N)] Required (R)
Scope (S)	Unchanged (U) [✓ Changed (C)]
Confidentiality (C)	None (N) [✓ Low (L)] High (H)
Integrity (I)	None (N) [✓ Low (L)] High (H)
Availability (A)	[✓ None (N)] Low (L) High (H)

Recommended Mitigation -

- Parameterised Queries or prepared statements should be used.
- Using stored procedures by being safe, i.e., not including any dynamic SQL generation. Use this in conjunction with input validation or proper escaping.
- Escaping user supplied input must be used as last resort
- When creating user accounts to a SQL database, one must obey the Principle of Least Privilege. This means that minimum privileges be given to users to access their account.
- Injection attacks may be caused due to prepared statements in which variables are not bound together. Hence, using SQL strings which bind variables may help prevent this attack.
- In every SQL command, whitelist style check should be used to disallow meta-characters instead of escaping them.

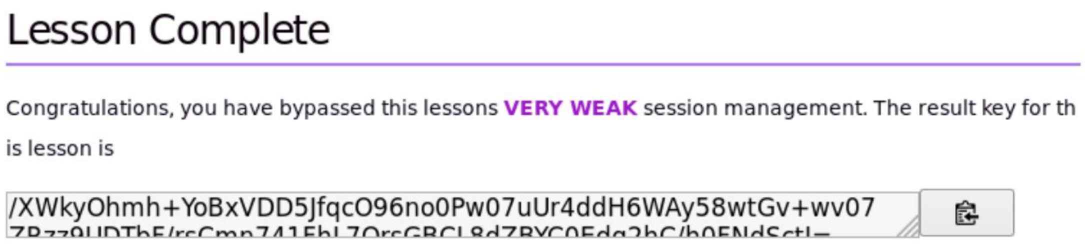
V. High - Broken Authentication [CWE- 287]

Other vulnerabilities pose certain security risks which may lead to attacks against the authority of an application. For example, using CSS vulnerability, session tokens may be stolen to lead to session management vulnerability. These flaws are common in functionalities like password management, secret question etc. This vulnerability has the potential to modify the credentials of the victim.

a) Broken Session Management Lesson [CWE- 287]

Steps to Reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.1.200 and login.
4. Go to Lessons -> Broken Session Management
5. On Burp Suite turn intercept on. Click on “Complete this lesson” Button.
6. Check on BurpSuite to see the interception raw, where lessonComplete=lessonNotComplete. Change this lessonNotComplete to lessonComplete
7. Click on forward and check Security shepherd:



Lesson Complete

Congratulations, you have bypassed this lessons **VERY WEAK** session management. The result key for th is lesson is

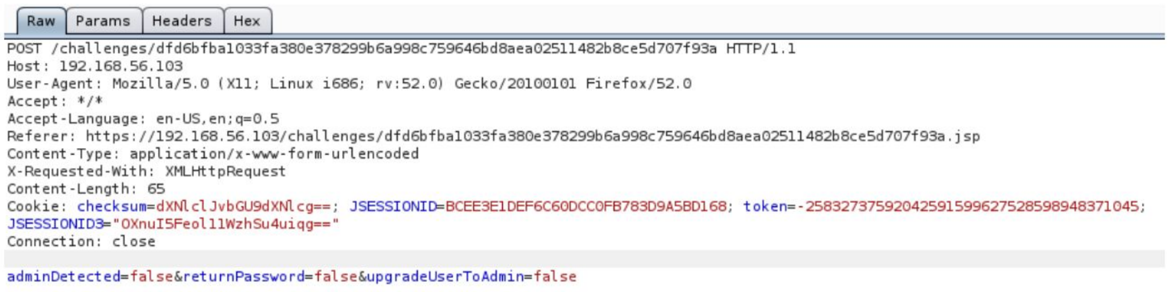
/XWkyOhmh+YoBxVDD5JfqcO96no0Pw07uUr4ddH6WAy58wtGv+wv07
ZRzz9UDThF/rsCmn741EhL7OrsGBCL8dZBYC0Edq2hC/h0ENdSctl=

b) Broken Session Management Challenge 1 [CWE- 287]

Steps to Reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp

4. Go to Challenges -> Session Management Challenge 1
5. Intercept as HTTP post request on burpsuite when we click on “Administrator only button”
6. We get this:



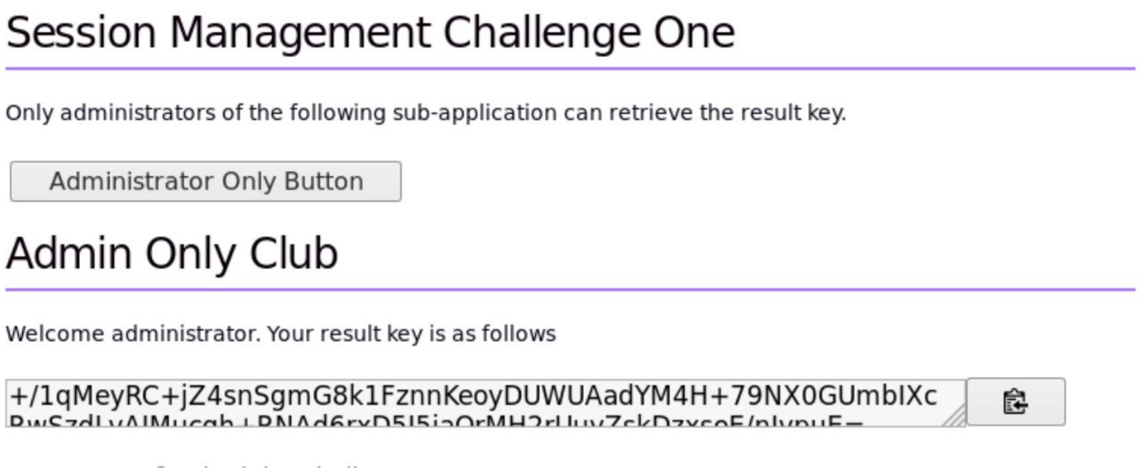
Raw Params Headers Hex

```
POST /challenges/dfd6bfba1033fa380e378299b6a998c759646bd8aea02511482b8ce5d707f93a HTTP/1.1
Host: 192.168.56.103
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.56.103/challenges/dfd6bfba1033fa380e378299b6a998c759646bd8aea02511482b8ce5d707f93a.jsp
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Content-Length: 65
Cookie: checksum=dXNlclJvbGU9dXNlcg==; JSESSIONID=BCEE3E1DEF6C60DCC0FB783D9A5BD168; token=-25832737592042591599627528598948371045;
JSESSIONID3="OXnuI5Feol1lWzhSu4uiqg=="
Connection: close

adminDetected=false&returnPassword=false&upgradeUserToAdmin=false
```

7. Notice checksum and the 3 boolean parameters: adminDetected, returnPassword, upgradeUserToAdmin.

8. Go to decoder on BurpSuite. Notice checkSum to have “==” means it may be base 64 format. Hence, decode checksum value as base 64.
9. We observe it to give “userRole = user”
10. Now, amend “user” to “administrator” and further encode as base 64.
11. Now replace this value in the checkSum and turn the boolean attributes to true.



Session Management Challenge One

Only administrators of the following sub-application can retrieve the result key.

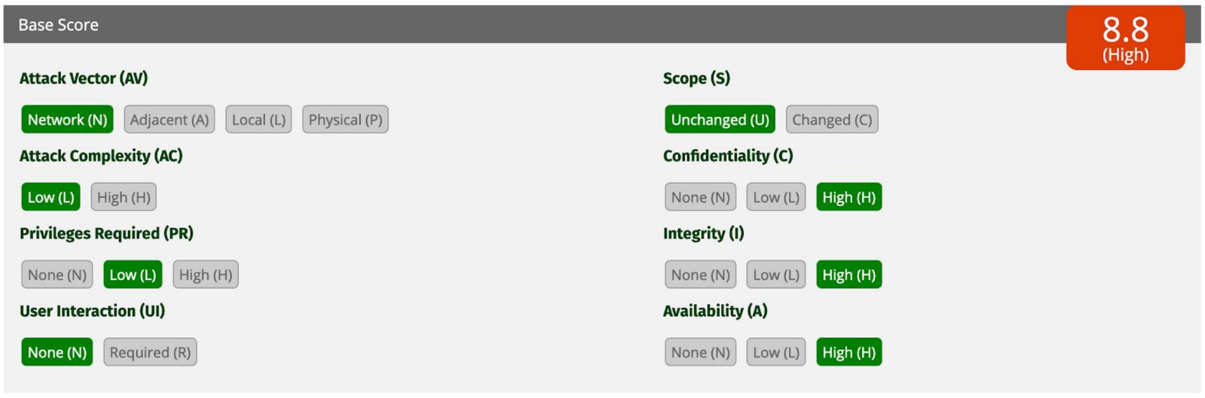
Admin Only Club

Welcome administrator. Your result key is as follows

+/1qMeyRC+jZ4snSgmG8k1FznnKeoyDUWUAadYM4H+79NX0GUmbIXc

12. Enter the key to solve this challenge

CVSS Score:



Base Score — 8.8 (High)

Attack Vector (AV)
[✓ Network (N)] Adjacent (A) Local (L) Physical (P)

Attack Complexity (AC)
[✓ Low (L)] High (H)

Privileges Required (PR)
None (N) [✓ Low (L)] High (H)

User Interaction (UI)
[✓ None (N)] Required (R)

Scope (S)
[✓ Unchanged (U)] Changed (C)

Confidentiality (C)
None (N) Low (L) [✓ High (H)]

Integrity (I)
None (N) Low (L) [✓ High (H)]

Availability (A)
None (N) Low (L) [✓ High (H)]

Recommended Mitigation -

- Making use of viable authentication and session management controls that match with the requirements defined in OWASP's Application Security Verification Standard areas and provide developers with a simple interface.
- Use of Transport Layer Security (TLS), which prevents active eavesdropping and passive disclosure in network traffic.
- Cookies must be sent using secure attribute, HTTPOnly attribute, domain and path attribute or expire and max-age attribute.
- Avoiding XSS flaws may help in prevention of stealing ids and hence prevent this vulnerability.

VI. Medium - Poor Data Validation [CWE-20]

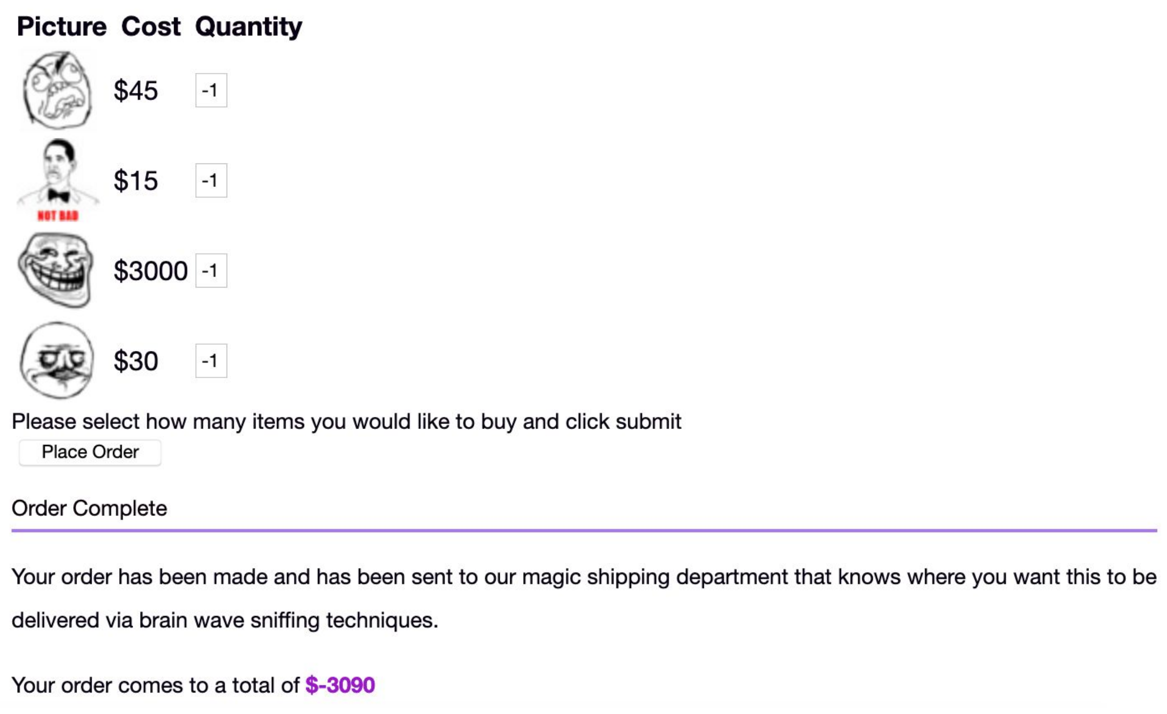
The failure to sufficiently validate submitted data by client or environment leads to Poor Data Validation. This weakness in itself is of low severity but may lead us to other major vulnerabilities in applications like file system attacks and buffer overflows, thereby increasing its attack impact. Client submitted data must not be trusted as it may be tampered with. They can cause business logic attacks or server errors.

Submitted data should be ensured to be strongly typed with appropriate syntax, length, use of permitted characters and within boundaries of range. The process of data validation must performed both on server side as well as on client side.

a) Poor Data Validation Challenge 1 [CWE-20]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp and login
4. Confirm that Burp can see and capture requests and turn off intercept in Burp
5. Go to Challenges -> Poor data validation -> Challenge 1
6. Enter all 1s in the quantity. Then try for all -1. You get the following which means data validation for negative number is absent:



Picture	Cost	Quantity
	$45	-1
	$15	-1
	$3000	-1
	$30	-1

Please select how many items you would like to buy and click submit

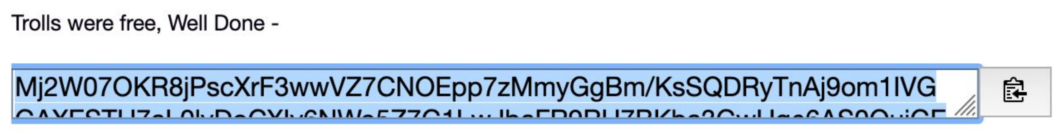
Place Order

Order Complete

Your order has been made and has been sent to our magic shipping department that knows where you want this to be delivered via brain wave sniffing techniques.

Your order comes to a total of $-3090

7. Now try -100 of $30 troll and 1 of $3000 troll. Total comes to be 0:



Trolls were free, Well Done -

Mj2W07OKR8jPscXrF3wwVZ7CNOEpp7zMmyGgBm/KsSQDRyTnAj9om1IVG

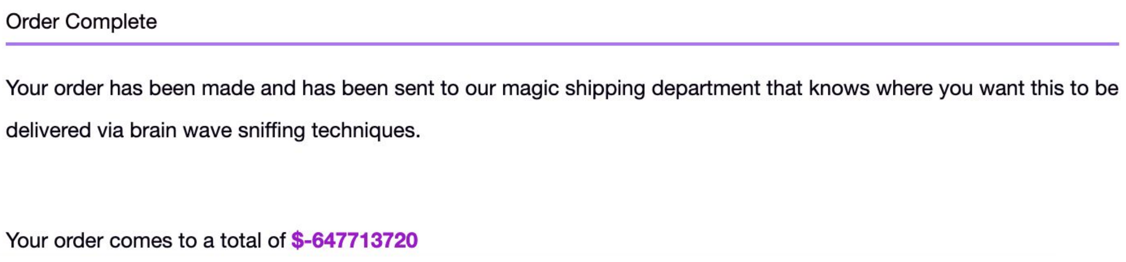
8. Challenge solved!

b) Poor Data Validation Challenge 2 [CWE-20]

Steps to reproduce

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp
4. Move on to Challenge 2.
5. Entering -1 in quantity still gives a total of $0 which means data validation for negative is present in this challenge.

6. Negative data validation is present, so we can now test boundary issues. Enter a number say 99999999.



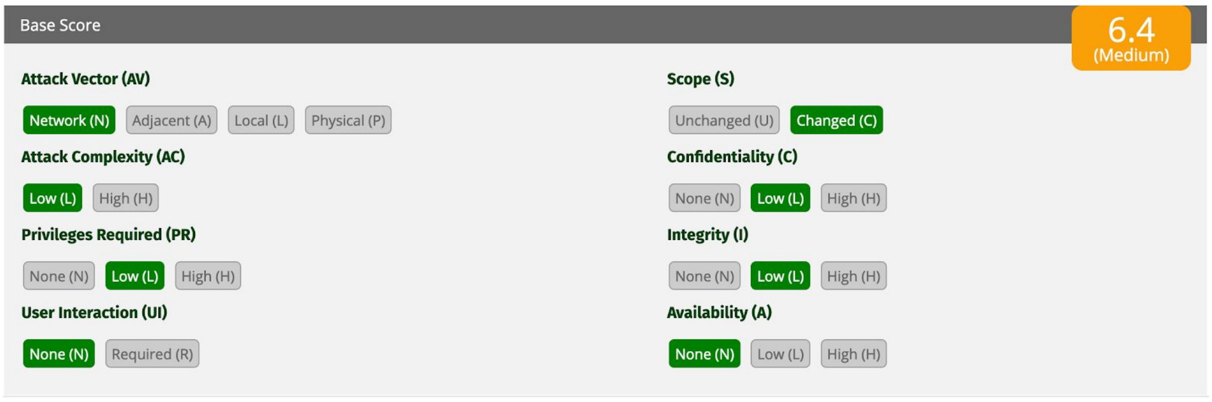
Order Complete

Your order has been made and has been sent to our magic shipping department that knows where you want this to be delivered via brain wave sniffing techniques.

Your order comes to a total of $-647713720

7. And we also get a key which when submitted gives success. Challenge completed!

CVSS Score:



Base Score — 6.4 (Medium)

Attack Vector (AV): Network (N) | Adjacent (A) | Local (L) | Physical (P) — selected: Network (N)

Attack Complexity (AC): Low (L) | High (H) — selected: Low (L)

Privileges Required (PR): None (N) | Low (L) | High (H) — selected: Low (L)

User Interaction (UI): None (N) | Required (R) — selected: None (N)

Scope (S): Unchanged (U) | Changed (C) — selected: Changed (C)

Confidentiality (C): None (N) | Low (L) | High (H) — selected: Low (L)

Integrity (I): None (N) | Low (L) | High (H) — selected: Low (L)

Availability (A): None (N) | Low (L) | High (H) — selected: None (N)

Recommended Mitigation -

- Use an input validation framework such as Struts
- use a whitelist of acceptable inputs that strictly conform to specifications.
- Validate free-form Unicode text
- Ensure that any input validation performed on the client is also performed on the server.
- Prevent XSS and Content Security Policy

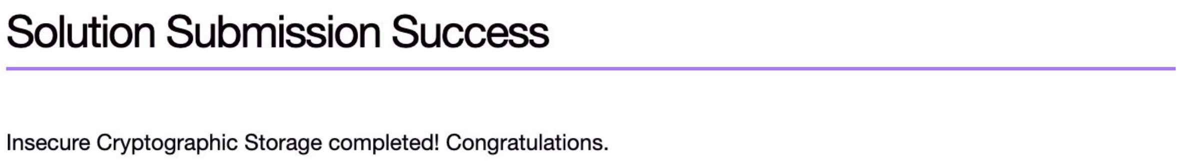
VII) High- Insecure Cryptographic Storage [CWE-327]

The most common issue in cryptographic storage is simply **not encrypting data** that deserves encryption. When encryption is employed, **unsafe key generation and storage**, **not rotating keys** and **weak algorithm usage** is common. Use of **weak or unsalted hashes** to protect passwords is also common. These mistakes can compromise all of the data that should have been encrypted. Typically this information includes sensitive data such as health records, credentials, personal data, credit cards, etc.

a) Insecure Cryptographic Storage Lesson [CWE-327]

Steps to Reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp and login
4. Move to Insecure Cryptographic Storage Lesson
5. Take the ciphered sentence and enter it in Burp Suite decoder.
6. Decode it with base 64. Copy the result key in submission box on Security Shepherd.


Solution Submission Success

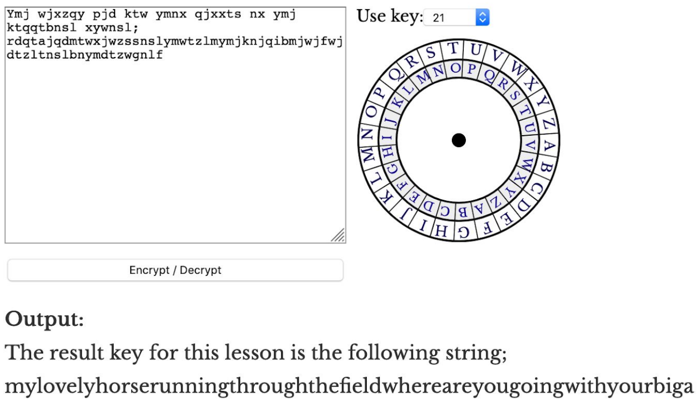
Insecure Cryptographic Storage completed! Congratulations.

b) Insecure Cryptographic Storage Challenge 1 [CWE-327]

Steps to Reproduce:

1. Download and run Burp Suite https://portswigger.net/burp/download.html (making sure you have Oracle Java Installed)
2. Utilising Firefox set the system proxy to route traffic through Burp - "Open Menu" button in the right hand corner -> Advanced -> Network (tab) -> Connection "Settings Button" -> Manual proxy configuration. The default for Burp is 127.0.0.1 with a port of 8080
3. Go to Security Shepherd https://192.168.56.103/index.jsp and login
4. Go to Insecure Cryptographic Storage -> Challenge 1

5. Use the Caeser cipher and enter it in an online decipher website ( ). It uses brute force by shifting it by as many as possible.
7. Try from key 13-21. Finally 21 gives us:



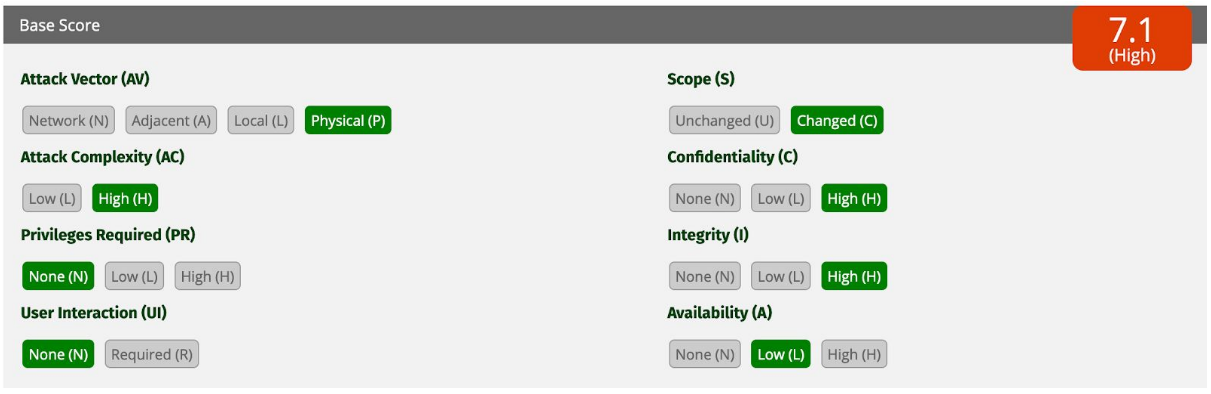
```
Ymj wjxzqy pjd ktw ymnx qjxxts nx ymj
ktqqtbnsl xywnsl;
rdqtajqdmtwxjwzssnslymwtzlmymjknjqibmjwjfwj
dtzltnslbnymdtzwgnlf
```

Use key: 21

Output:
The result key for this lesson is the following string;
mylovelyhorserunningthroughthefieldwhereareyougoingwithyourbiga

7. Use the string in submission key to complete the challenge.

CVSS Score:



Base Score 7.1 (High)

Attack Vector (AV): Network (N) | Adjacent (A) | Local (L) | [✓ Physical (P)]

Attack Complexity (AC): Low (L) | [✓ High (H)]

Privileges Required (PR): [✓ None (N)] | Low (L) | High (H)

User Interaction (UI): [✓ None (N)] | Required (R)

Scope (S): Unchanged (U) | [✓ Changed (C)]

Confidentiality (C): None (N) | Low (L) | [✓ High (H)]

Integrity (I): None (N) | Low (L) | [✓ High (H)]

Availability (A): None (N) | [✓ Low (L)] | High (H)

Recommended Mitigations -

- Ensure encryption of all data at rest to prevent these threats.
- Make sure that encryption of offsite backups along with separate management and back-up of keys.
- Ensure key management is in place and strong keys are used.
- Ensure hashing of password and correct salt usage

Recommendations and Conclusions

As revealed from this penetration test, about 16 vulnerabilities were detected in the Security Shepherd application. These vulnerabilities ranged from high to medium. This clearly shows that they have severe impacts on the safety of the website. Appropriate resources should thus be allocation so that these vulnerabilities may be solved. Recommended mitigations have also been given in this report owing to each issue, along with their CVSS score and steps to reproduce. It was also observed how some vulnerabilities lead to other vulnerabilities, hence avoiding these at an earlier stage is highly recommended.

1. The use of strong credentials such as username and passwords is a must.
2. Remove hard coded URLs so it gets more hard to hack into sensitive information.
3. Patch Management Program: As per NIST SP 800-4010, following the guidelines a patch management program must be implemented regularly so as to decrease the attacks
4. OWASP’s Application Security Verification Standard must be referred to.
5. Use of TLS and SSL should be implemented along with session cookies.
6. Regular Vulnerability checks should be implemented.

Risk Rating

The overall risk identified to Security Shepherd as a result of the penetration test is **Medium to High.** It will not be too difficult for hackers to successfully plan attacks on Security Shepherd in the above found ways or more.