Name: Jun Wang 000791814

Lab 1: Mapping the Web Application

Web Application Security

Mark:

Bonus:

Table of Contents

[Lab Outcomes 2](#_Toc18952032)

[Background Reading 2](#_Toc18952033)

[Introduction 2](#_Toc18952034)

[Required Hardware/Software 2](#_Toc18952035)

[1.0 Installing DVWA 2](#_Toc18952036)

[2.0 Installing WebGoat 3](#_Toc18952037)

[3.0 HTTP Transactions 4](#_Toc18952038)

[3.0 Spidering 4](#_Toc18952039)

[4.0 Discovering Hidden Content 5](#_Toc18952040)

[5.0 Mapping the Application 5](#_Toc18952041)

[6.0 Sign-Off – Lab 1: Mapping the Web Application 6](#_Toc18952042)

Web Application Security

Lab 1: Mapping the Web Application

# Lab Outcomes

* Enumerate the web application.
* Analyze the attack surface.

Background Reading

Read the textbook sections listed in the Course Schedule.

Introduction

Before you attack a web application, familiarize yourself with what it can do and how it works. You need to know what you’re up against!

Required Hardware/Software

* VM Ubuntu 18.04 – 2 CPU, 4GB Ram, 20 GB hard disk
  + Docker
    - WebGoat v7.1
    - DVWA (Dawn Vulnerable Web App)
* VM Kali
  + Burp or other Web proxy (scanner)
  + DirBuster

# Installing DVWA

DVWA is a deliberately insecure web application

See <https://www.youtube.com/watch?v=5BG6iq_AUvM> and https://github.com/ethicalhack3r/DVWA

To get DVWA running, needed to remove Apache2

sudo apt autoremove apache2

Goto DVWA docker setup

https://github.com/ethicalhack3r/DVWA

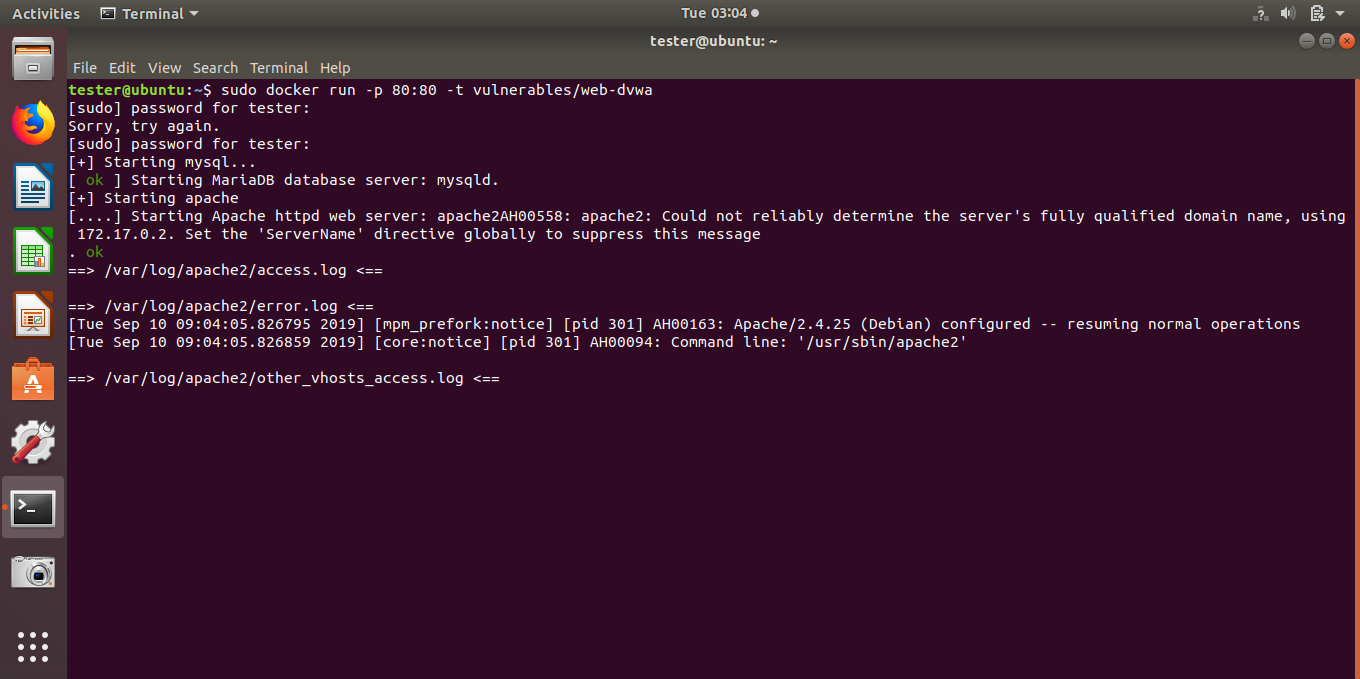
https://hub.docker.com/r/vulnerables/web-dvwa/

Install DVWA

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

Run DVWA

sudo docker run -p 80:80 -t vulnerables/web-dvwa



DVWA

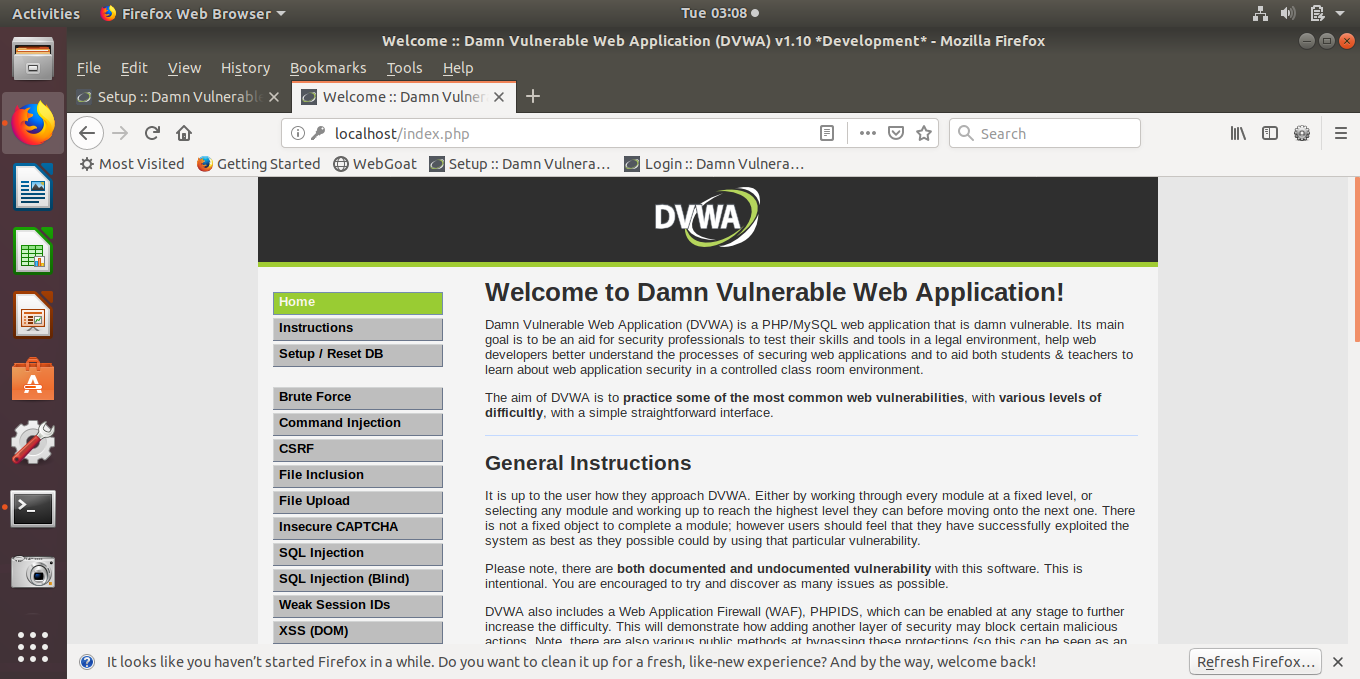
http://localhost/login.php

Login with default credentials

To login you can use the following credentials:

•Username: admin

•Password: password



run then reset database

http://localhost/setup.php

# Installing WebGoat

WebGoat is a deliberately insecure web application maintained by the Open Web Application Security Project (OWASP) to test vulnerabilities.

1. Using your Ubuntu VM,
   1. Setup Ubuntu for required packages
      1. apt-transport-https ca-certificates curl software-properties-common
2. Install Docker

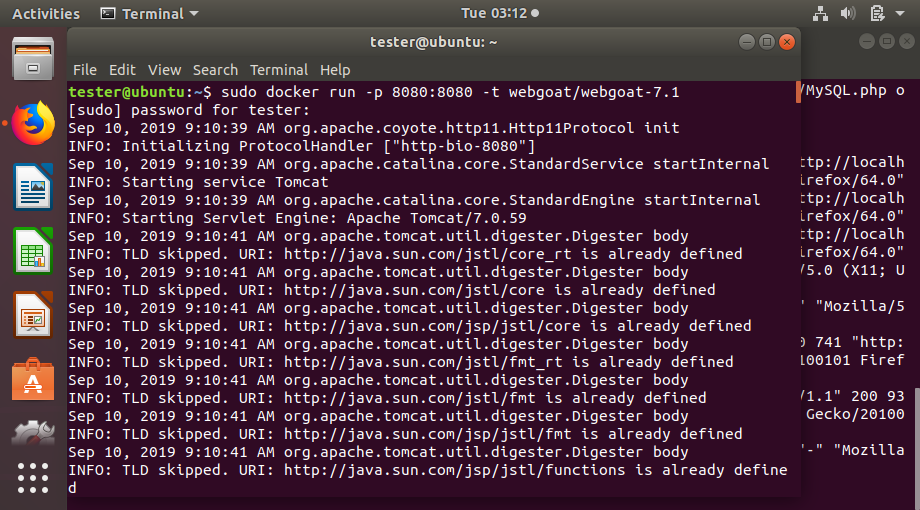
https://www.how2shout.com/how-to/steps-to-install-docker-ce-on-ubuntu-18-04-19-04.html

1. Then install the Docker version of WebGoat Install Webgoat

docker pull webgoat/webgoat-7.1

1. Run Webgoat

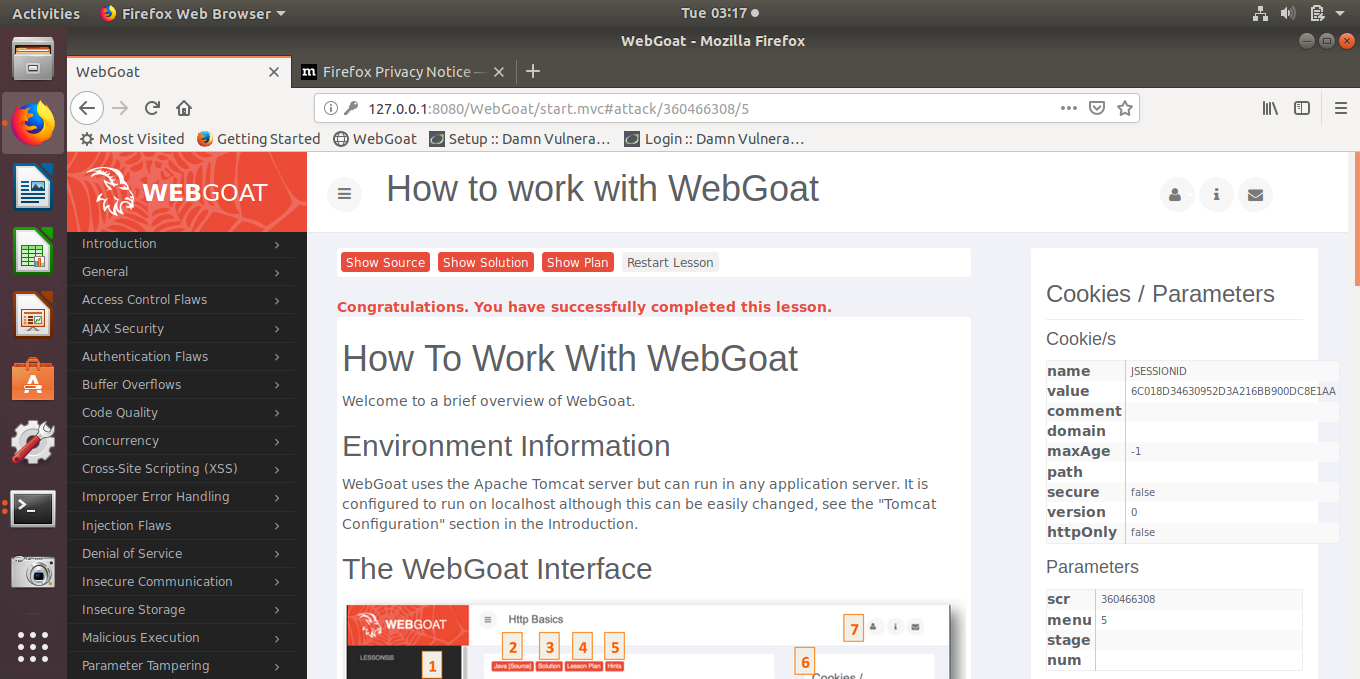
sudo docker run -p 8080:8080 -t webgoat/webgoat-7.1



1. Open WebGoat in your browser.

127.0.0.1:8080

**Tip:** Bookmark this page. You will use it for this entire course.



1. Log in as a guest.
2. Read the **How to work with WebGoat** section under *Introduction*.

# Explore the Applications

Using browser, proxy or other tools, answer the questions below:

firefox->rightclick->inspect elements->network->click on any request-> headers->raw header

1. Identify the Web Server version of DVWA: **Apache/2.4.25(Debian)**

|  |
| --- |
|  |

1. Identify the Web Application Server version of DVWA: **PHP/MySQL**

**PHP/7.0.30**

**Mysqlnd 5.0.12**

|  |
| --- |
|  |

1. Identify the Web Server version of WebGoat:  **Apache-Coyote/1.1**

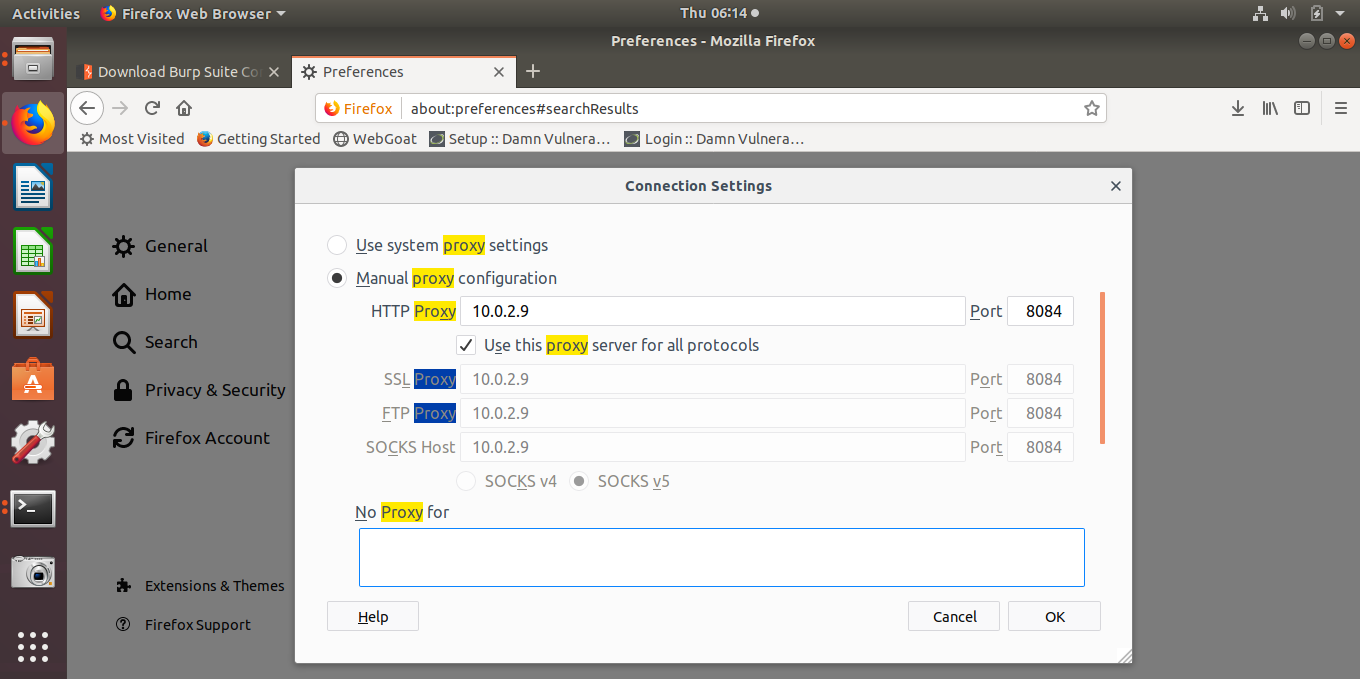
|  |
| --- |
| Insert evidence here. |

1. Identify the Web Application Server version of WebGoat: **Tomcate 7.0.59**

|  |
| --- |
| Insert evidence here. |

# HTTP Transactions

1. In WebGoat, complete the **Http Basics** lesson under *General*.
2. Follow the [instructions to set up a proxy](https://github.com/WebGoat/WebGoat/wiki/Using proxy tools) (https://github.com/WebGoat/WebGoat/wiki/  
   Using%20proxy%20tools) for intercepting HTTP requests to the WebGoat server using Burp Suite.
3. Change the proxy listener to port **8084**.
4. In the browser, select the **Use this proxy server for all protocols** option and remove all the entries under *No Proxy for*.



1. Demonstrate to your instructor how to intercept the request and change the name that is submitted from the form.

|  |
| --- |
| **Step1 set vm and kali networks to same network**  **Step2 run webgoat on vm , login on firefox. Set proxy on vm firefox.**  **Step3 on kali, turn off burp interception. Set target -> add -> vm ‘s ip address ; port :8080**  **options->proxy listeners : interface port 8084**    **Step4 turn on burp interception, fill the form and submit request on vm .**  **Step5 check and make sure the request package is intercepted by burp.**  **Step 6 Forward the request to vm. The outcome is that the name I input is reversed.** |
|  |

# 3.0 Spidering

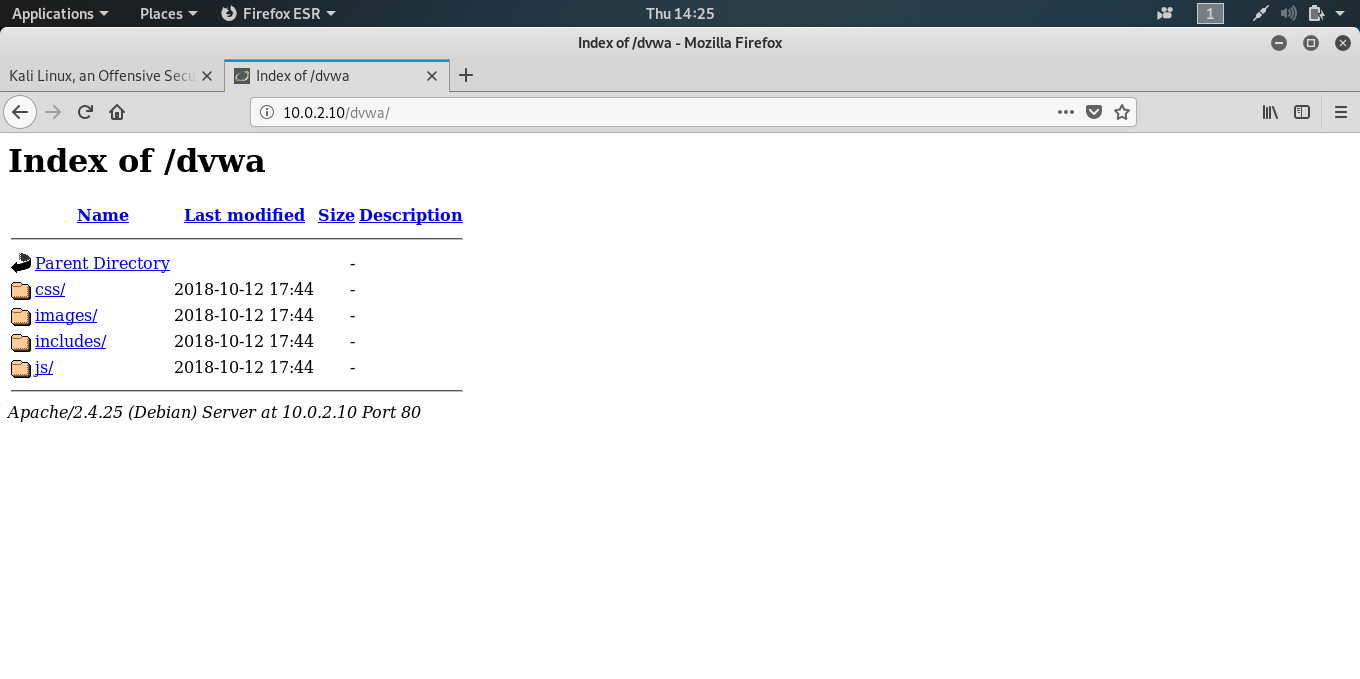
1. Start the DVWA VM.

(http://www.computersecuritystudent.com/SECURITY\_TOOLS/DVWA/DVWAv107/lesson1/)

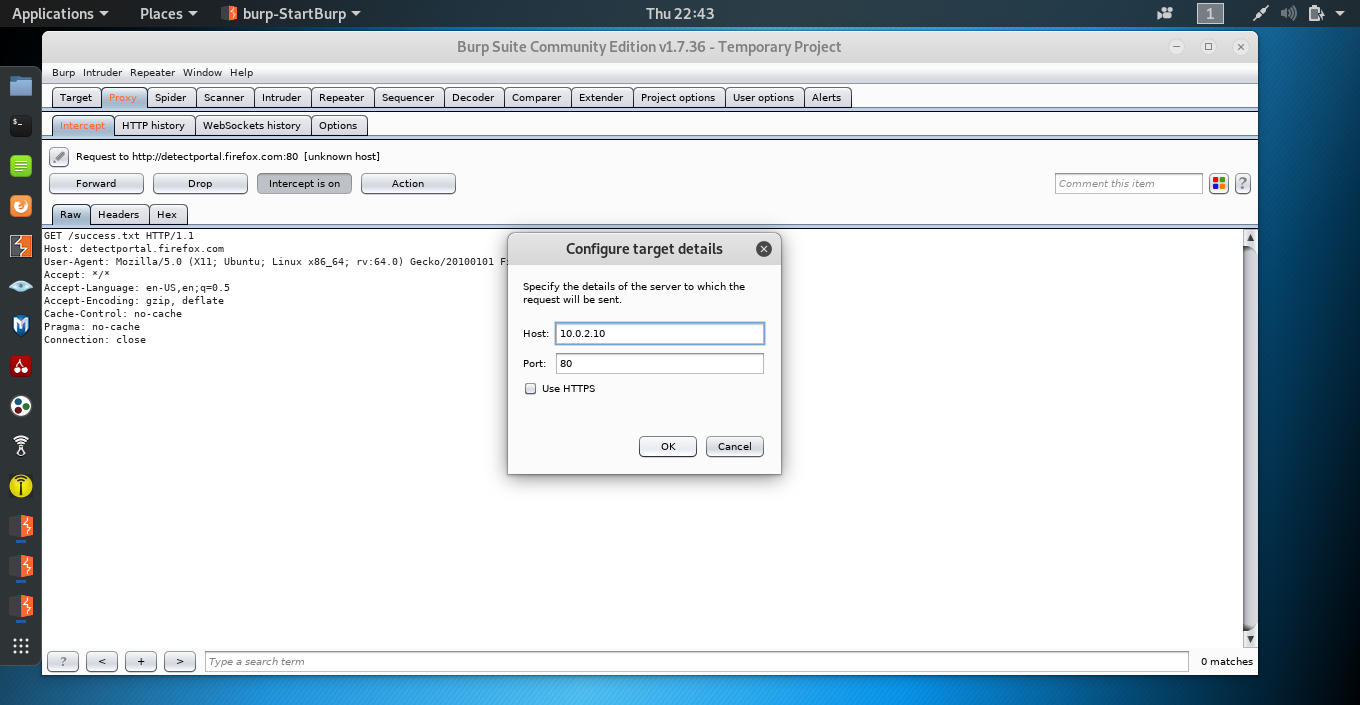
1. Launch the XAMPP Control Panel that is pinned to the task bar.

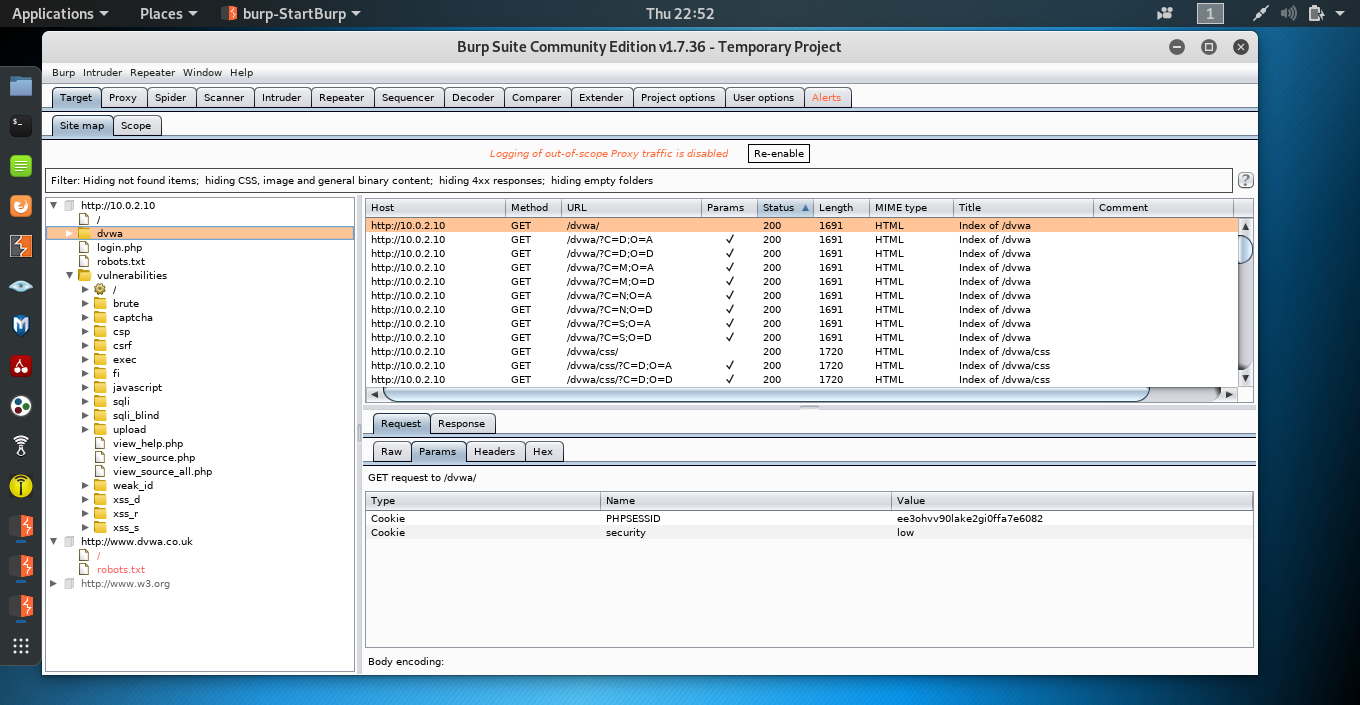
Apache and MySQL servers start automatically.

1. Find the IP address of the VM (Ubuntu) and make sure you can connect to it in your Kali VM at **http://<ip\_adress>/dvwa**.



1. Use Burp or other proxy to Spider to perform an automatic enumeration of DVWA.

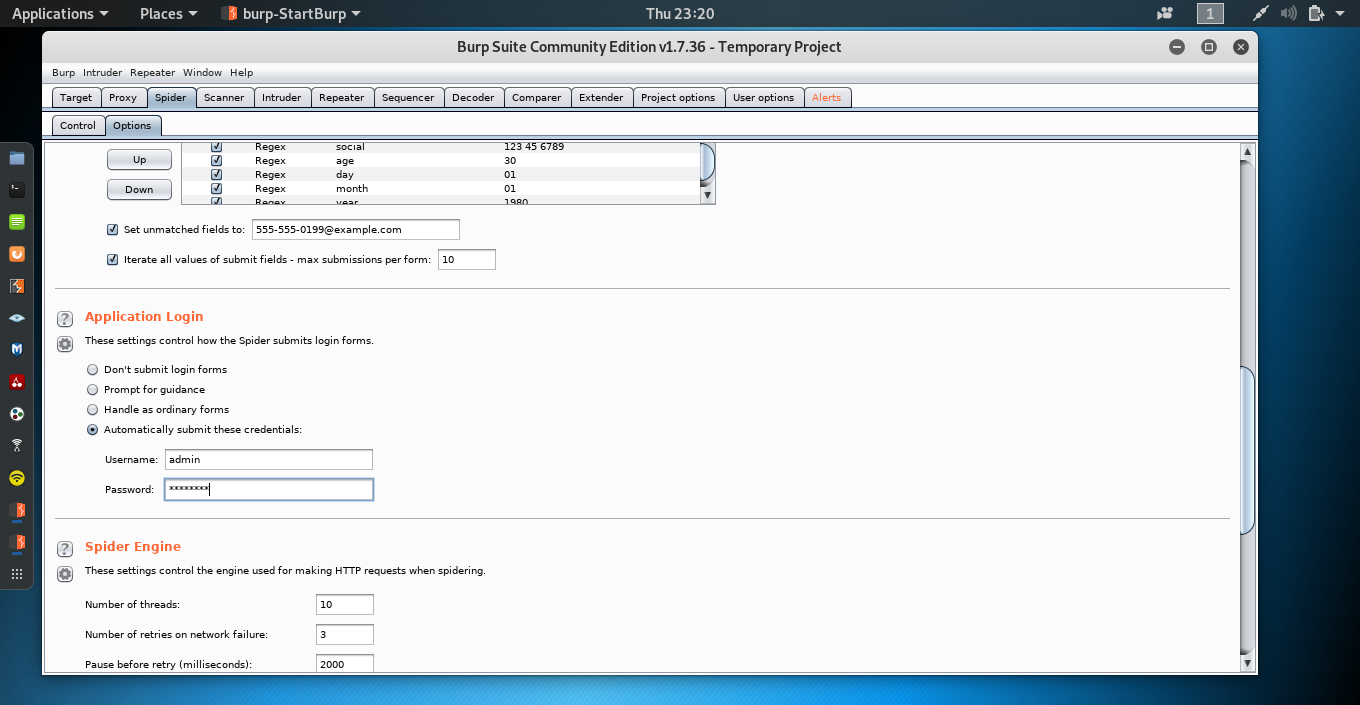


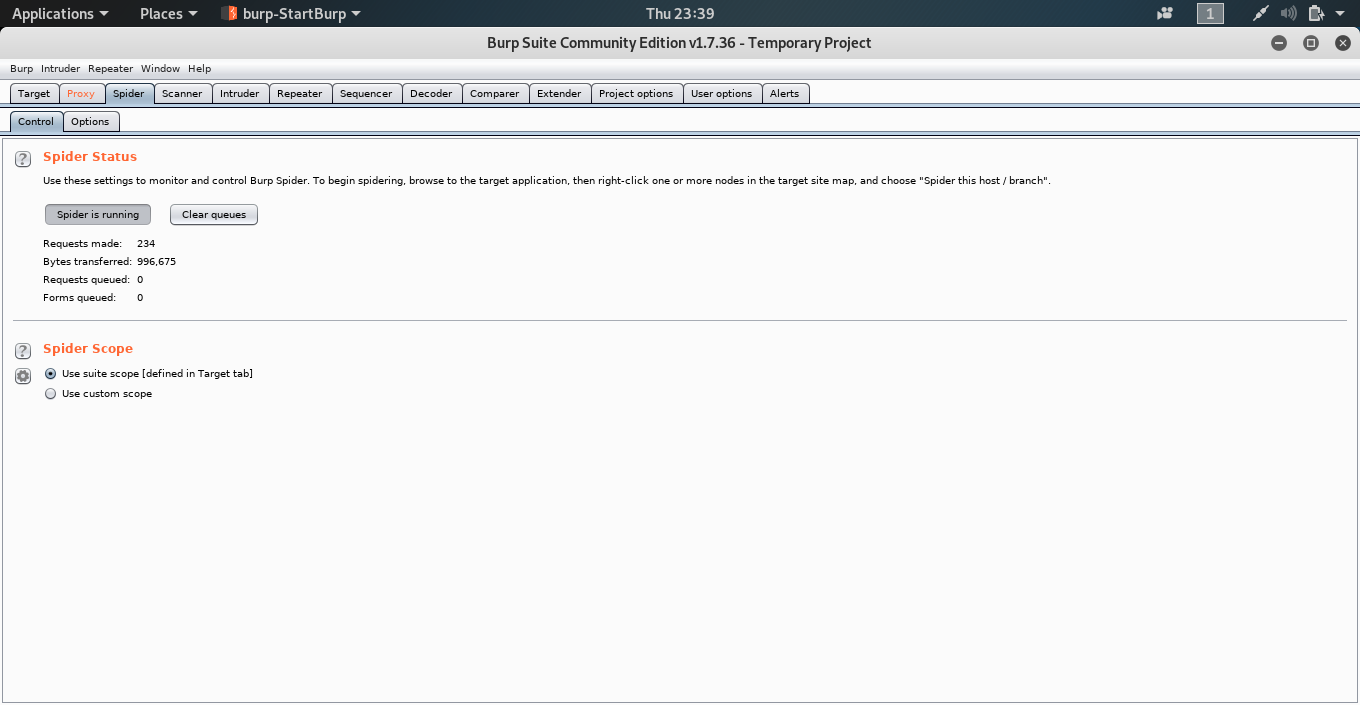
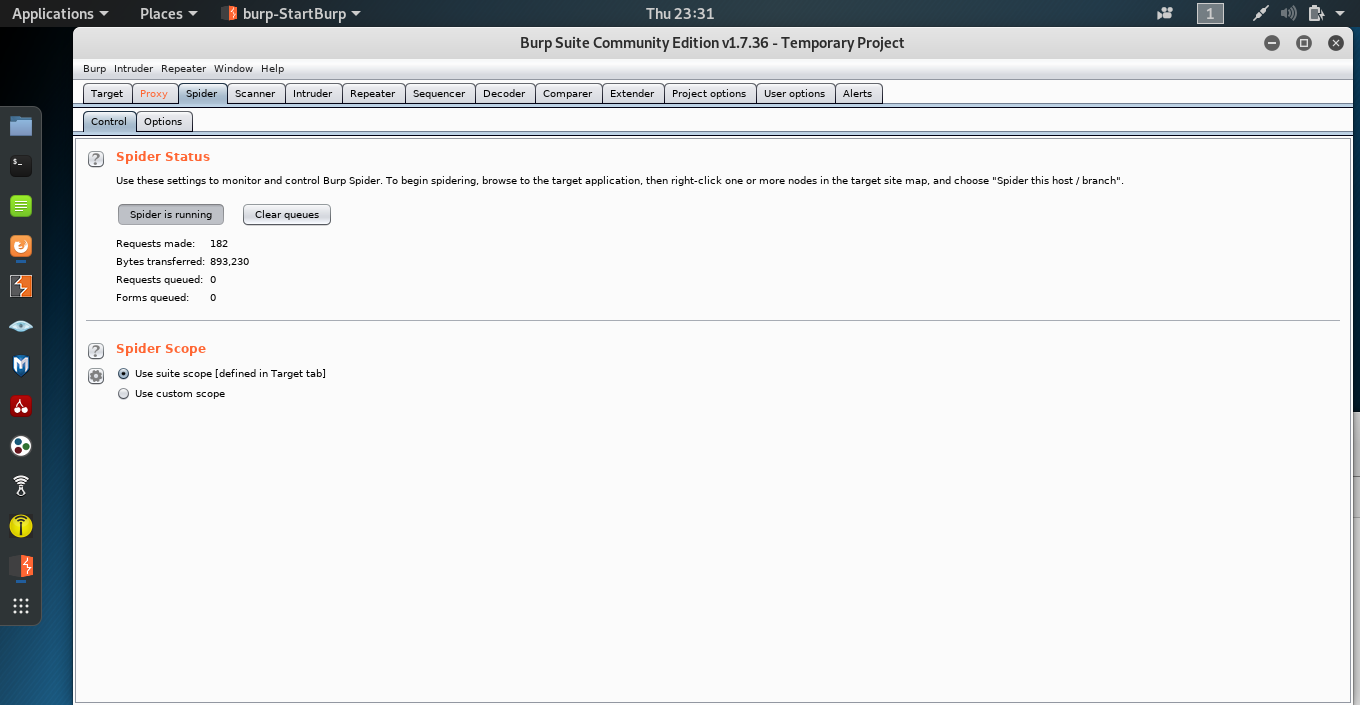
1. Demonstrate and explain the results to your instructor.
   1. Were you prompted for login credentials and guidance during the spidering process?

**Yes.**

|  |
| --- |
|  |

* 1. What does this mean and how does it affect the results?

Note: Set credentials on burp: spider->options->Application Login->automatically submit these credentials.

\_\_**Normally, we do not have the credentials, which means we have to ignore the form and this results in less information we can grab from the target. On the left screenshot, 893230 bytes were obtained without credentials, while with credentials, the right screenshot shows 100,000more bytes were spidered.**

# 4.0 Discovering Hidden Content

You can use DirBuster to perform a brute-force probe for common directories in a web application.

1. Start DirBuster.

**Note:** This can take a minute or two, so be patient.

1. Target the DVWA and choose a wordlist from **/usr/share/dirbuster/wordlists/**. (Optionally, choose one of the smaller word lists to speed up the scan.)
2. Compare the results to the results from your spidering session, and identify any additional data that was discovered by DirBuster.

|  |
| --- |
|  |

1. Demonstrate and explain to your instructor how this can help enumerate the web application.

**Dirbuster discovered 82 directories and 246 files by using a small word list. It revealed the following directories icon, docs, external, config and vulnerabilities which are hidden and spider failed to reveal.**

**Dirbuster also provides response codes, which indicate the unreachable files and the reasons.**

# 5.0 Mapping the Application

Mapping a web application takes practice. There are many automated tools that can help you work through the process, but they aren’t magic and they can’t find everything. It’s up to you as the security professional to fill in the gaps.

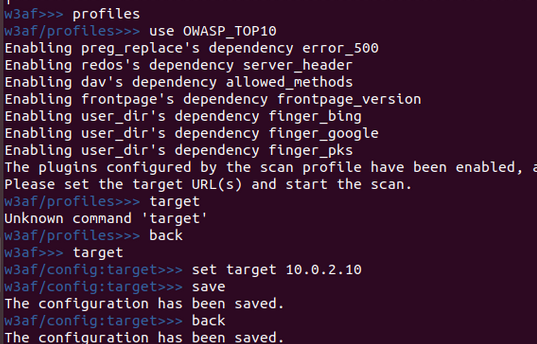
1. Turn the security setting of DVWA to **low**.
2. Review how to map a web application from your class notes and the textbook.
3. Go to p. 106 of the textbook and perform the Hack Steps on DVWA.
4. Use the Web Application Attack and Audit Framework (w3af) to map DVWA, and then compare the results to your findings from the previous step.

**Note:** Use the OWASP\_TOP10 profile.

1. Perform the same type of scan using Burp Suite and compare the results to w3af.

**Burp is an integrated platform for performing security testing of web applications. w3af is a Web Application Attack and Audit Framework. Comparing with Burp suite, w3af focus on finding and exploiting all web application vulnerabilities. W3af provides detailed information about vulnerabilities it found on the web application including, description summary and fix guidance. It also provides an easy access to launch an exploit.**

1. Demonstrate and explain the results to your instructor.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



|  |
| --- |
|  |

# 6.0 Sign-Off – Lab 1: Mapping the Web Application

Detach this page and submit it to your instructor.

Name: Jn Wang

Student ID: 000791814

|  |  |
| --- | --- |
| **Section** | **Instructor Initials** |
| 2.0 HTTP Transactions |  |
| 3.0 Spidering |  |
| 4.0 Discovering Hidden Content |  |
| 5.0 Mapping the Application |  |