# Start

This document describes how to install and configure Moodle on Ubuntu 14.04.

# About Moodle

Moodle is a free software e-learning platform, also known as a Learning Management System.

As of June 2013 it had a user base of 83,008 registered and verified sites, serving 70,696,570 users in 7.5+ million courses with 1.2+ million teachers. Equipped with a horde of smart features such as virtual quizzes, assignment submission, wiki, grading, IM services, and online discussion boards, Moodle enables a seamless online learning experience for end-users. Also, by virtue of being a modular software, it is capable of enhanced functionality through customized plugins.

### Step 1

This guide is based on Ubuntu 12.04 server, so you should set up a basic Ubuntu 12.04 server installation before you continue with this tutorial. The system should have a static IP address. I use *192.168.0.100* as my IP address in this tutorial and *server1.example.com* as the hostname.

### Step 2 Install Apache/MySQL/PHP

I will be using root credentials(which give me compelte control over the server), if you are doing installation with any other user just prefix sudo before all commands:

First Login into the server , using the username and password.

Next type in **Sudo Su.** It will ask you for the password again

Next we need to install Apache, MySQL and PHP before Moodle installation.

**apt-get -y install apache2 mysql-client mysql-server php5**

You will be asked to provide a password for the MySQL root user this password is valid for the user *root@localhost* as well as *root@server1.example.com*, so we don't have to specify a MySQL root password manually later on:

*New password for the MySQL "root" user:*    
*Repeat password for the MySQL "root" user:* 

Moodle require some more packages we will install them as follows:

**apt-get -y install graphviz aspell php5-pspell php5-curl php5-gd php5-intl php5-mysql php5-xmlrpc php5-ldap git-core**

I will be using *git* utility for downloading Moodle so I have installed *git-core*. Which make it easer for all.

For ubuntu 12.05 it has php.5.3 installed and it needs PHP 5.4 for moodle to run.

So we need to install pyton packages which allow manage the repositories that you install software from.

sudo apt-get install python-software-properties

After this we need to download the following personal packages. This one is very popluar.

sudo add-apt-repository ppa:ondrej/php5

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install php5

Were all good to go :D

### Step 3 : Download Moodle

Now we will download Moodle as follows:

1. **cd /opt**
2. **git clone git://git.moodle.org/moodle.git**
3. **cd moodle**
4. **git branch -a**
5. **git branch --track MOODLE\_27\_STABLE origin/MOODLE\_27\_STABLE**
6. **git checkout MOODLE\_27\_STABLE**

Next we need to copy the Moodle content to the */var/www/html* as:

1. **cp -R /opt/moodle /var/www/html/**
2. **mkdir /var/moodledata**
3. **chown -R www-data /var/moodledata**
4. **chown -R www-data /var/www/html/moodle**
5. **chmod -R 777 /var/moodledata**
6. **chmod -R 0755 /var/www/html/moodle**

We need to change the default storage engine to *innodb*:

1. **nano /etc/mysql/my.cnf**

Give values at the end of the file as:

* #Basic settings for moodle
* default-storage-engine = innodb

### Step 4 Database initialization

We need a database for Moodle, I will create  the database for the Moodle as follows:

* mysql -u root -p

Here we are adding *database=moodledb user=moodleuser* and *password=moodlepassword*:

* CREATE DATABASE moodledb DEFAULT CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;
* GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,CREATE TEMPORARY TABLES,DROP,INDEX,ALTER ON moodledb.\* TO moodleuser@localhost IDENTIFIED BY 'moodlepassword';

Further moving ahead:

* FLUSH PRIVILEGES;
* Exit

Restart services

* service apache2 restart
* service mysql restart

### Step 5 :Changing the default.conf

In my Ubuntu 12.04 LTS, the document root was set to /var/www/html. It was configured in the following file:

/etc/apache2/sites-available/000-default.conf

So just do a

sudo nano /etc/apache2/sites-available/000-default.conf

and change the following line to what you want:

DocumentRoot /var/www/html

I changed it to /var/www/html/moodle/

Also do a

sudo nano /etc/apache2/apache2.conf

and find this

<Directory /var/www/html/>

Options Indexes FollowSymLinks

AllowOverride None

Require all granted

</Directory>

and change /var/www/html to your preferred directory

and save it.

After you saved your changes, just restart the apache2 webserver and you'll be done :)

sudo service apache2 restart

Were all done

### Step 6 : Web installation of Moodle

Now we will proceed with the Moodle web-installation. Open a browser of your choice and open the link *http:192.168.0.100/moodle/install.php*

Select your language and press *Next*:

Change the value of *Data directory* to */var/moodledata*and press *Next*:

Press *Next*:

Give the values at the time of creation of the database, in my case it was as follows:

Database host    :   *localhost*  
Database name    :    *moodledb*  
Database user    :    *moodleuser*  
Database password    :    *moodlepassword*  
table prefix    :   *mdl\_*  (or any valaue of your choice)  
Database port    :    *3306*  
Unix socket    :    It will remain blank.

After giving the values press *Next*:

Press *Continue*:

Press *Continue*:

Now create an admin user for Moodle, you can use any value as per your choice. In my case I am using:  
Username    :    *admin*  
New password    :    *Howtoforge@1*  
First name    :    *Srijan*  
Surname    :    *Kishore*  
EMail    :    *admin@example.com*  
  
All other field I will be using its default value, ofcourse you can change it later.

Press*Update profile*:

Give any name as per your choice, I am using a test name as:

Full site name    :   *Test-moodle*  
Short name for site    :    *test*  
Self registration    :    *Disable*

After giving the values  press *Save changes*:

Now go to *Site Administration*-->*Server*-->*System Paths* and define the paths as follows:

Path to du    :    */usr/bin/du*  
Path to aspell    :    */usr/bin/aspell*  
Path to dot    :    */usr/bin/dot*

Press *Save Changes*.

Now we are done with the installation part of Moodle, We can access the Moodle page at  *http://192.168.0100/moodle/login/index.php* as follows:

Your credentials will be *Username*=*admin* and *Password*=*Howtoforge@1*.  
Next page will be the default welcome page.

Now we can add courses and use Moodle as per our requirement. Congratulations! You now we have a fully functional Moodle instance on our Ubuntu 12.04

Now we will proceed with the Moodle web-installation. Open a browser of your choice and open the link *http:192.168.0.100/moodle/install.php*

### How to hide Apache Version and OS Identity from Errors

In above picture, you can see that **Apache** is showing its **version** with the **OS installed** in your server. This can be a major security threat to your web server as well as your Linux box too. To prevent **Apache** to not to display these information to the world, we need to make some changes in Apache main configuration file.

Open configuration file with vim editor and search for “**ServerSignature**“, its by default On. We need to **Off** these server signature and the second line “**ServerTokens Prod**” tells Apache to return only Apache as product in the server response header on the every page request, It suppress the OS, major and minor version info.

# vim /etc/httpd/conf/httpd.conf (RHEL/CentOS/Fedora)

# vim /etc/apache/apache2.conf (Debian/Ubuntu)

ServerSignature Off

ServerTokens Prod

# service httpd restart (RHEL/CentOS/Fedora)

# service apache2 restart (Debian/Ubuntu)

### 2. Disable Directory Listing

We can **turn off** directory listing by using **Options directive** in configuration file for a specific directory. For that we need to make an entry in **httpd.conf** or **apache2.conf** file.

<Directory /var/www/html>

Options -Indexes

</Directory>

### Use mod\_security and mod\_evasive Modules to Secure Apache

These two modules “**mod\_security**” and “**mod\_evasive**” are very popular modules of Apache in terms of security.

#### Mod\_security

Where **mod\_security** works as a **firewall** for our web applications and allows us to **monitor traffic** on a real time basis. It also helps us to protect our websites or web server from **brute force attacks**. You can simply install **mod\_security** on your server with the help of your default package installers.

##### **Install mod\_security on Ubuntu/Debian**

$ sudo apt-get install libapache2-mod-security

$ sudo a2enmod mod-security

$ sudo /etc/init.d/apache2 force-reload

##### **Install mod\_security on RHEL/CentOS/Fedora/**

# yum install mod\_security

# /etc/init.d/httpd restart

##### **Mod\_evasive**

**mod\_evasive** works very efficiently, it takes one request to process and processes it very well. It prevents **DDOS attacks** from doing as much damage. This feature of **mod\_evasive**enables it to handle the **HTTP brute force** and **Dos** or **DDos** attack. This module detects attacks with three methods.

* If so many requests come to a same page in a few times per second.
* If any child process trying to make more than **50** concurrent requests.
* If any **IP** still trying to make new requests when its temporarily **blacklisted**.

**mod\_evasive** can be installed directly from the source. Here, we have an Installation and setup guide of these modules which will help you to set up these Apache modules in your Linux box.

* [**Protect Apache using Mod\_Security and Mod\_evasive**](http://www.tecmint.com/protect-apache-using-mod_security-and-mod_evasive-on-rhel-centos-fedora/)

### 13. Securing Apache with SSL Certificates

Last, but not the least **SSL certificates**, you can secure your all the communication in an encrypted manner over the Internet with SSL certificate. Suppose you have a website in which people login by proving their Login credentials or you have an E- Commerce website where people provides their **bank details** or **Debit**/**Credit** card details to purchase products, by default your web server send these details in plain – text format but when you use **SSL certificates** to your websites, **Apache** sends all this information in encrypted text.

You can **purchase SSl certificates** from So many different SSL providers like**namecheap.com**. If you are running a very small web business and do not willing to**purchase an SSL certificate** you can still assign a **Self signed certificate** to your website.**Apache** uses the **mod\_ssl** module to support **SSL certificate**.

# openssl genrsa -des3 -out example.com.key 1024

# openssl req -new -key example.com.key -out exmaple.csr

# openssl x509 -req -days 365 -in example.com.com.csr -signkey example.com.com.key -out example.com.com.crt

Once your certificate has been created and signed. Now you need to add this in Apache configuration. Open main configuration file with vim editor and add the following lines and restart the service.

<VirtualHost 172.16.25.125:443>

SSLEngine on

SSLCertificateFile /etc/pki/tls/certs/example.com.crt

SSLCertificateKeyFile /etc/pki/tls/certs/example.com.key

SSLCertificateChainFile /etc/pki/tls/certs/sf\_bundle.crt

ServerAdmin ravi.saive@example.com

ServerName example.com

DocumentRoot /var/www/html/example/

ErrorLog /var/log/httpd/example.com-error\_log

CustomLog /var/log/httpd/example.com-access\_log common

</VirtualHost>

Open up your browser, type **https://example.com**, and you will be able to see the new **self-signed certificate**.

These are few security tips that you can use to **secure your Apache web server**installation. For more useful security tips and ideas, see the official online documentation of