Operating Systems – Linux 1-2. Laboratory – Measurement Guide and Report Template

## Legend

This document displays commands to be entered or choices to be selected in blue bold text. When these commands or choices are only referenced **black bold text** is used.

## Introduction

You are a member of a group of young engineers who just created a prototype product that is promising enough to build a startup company on. Each members of the group have their own tasks and responsibilities: there are members working on the development, there is a member for fund raising, another for graphics and design and so on. Your task is to provide the IT background for the project. Unfortunately you are on a very low budget: as there are no investments yet, all costs are paid by the team members. As for the reputation of the young startup it is very important to only use legal software and not to conflict any licensing conditions.

By now the prototype is ready to be introduced to the investors and to the general public. For that aim the team decides to set up a trendy one page static HTML website. The designer member of the team will be responsible for the graphics and design, and your task is to set up the server computer for the homepage. When working on this task you have to focus on two key requirements: On the one hand the costs must be kept low, therefore setting up a standalone server is not an option. On the other hand the server might experience high load from a large number of visitors after the project has been announced, thus the solution should be easy to scale up when needed.

Based on these requirements you decide to rent a virtual private server (VPS), which starts pricing from 5 USD/month in 2016. The smallest VPS servers have 512 MB RAM, a single processor core and 10-20 GB storage. Most VPS service providers offer time based billing and dynamic server resizing, thus for high load times the underlying hardware can temporarily be upgraded, to fulfill the second requirement.

## 1. Find a suitable VPS service

Search the Internet for VPS services in the EU that fit your requirements for the VPS!

**Enter the name of the chosen VPS provider and their offered package!**

**Insert a screenshot of the details of the VPS package you selected!**

## 2. Set up a virtual machine

After creating the VPS you decided to set up a virtual machine on your own computer to experiment with new software and configuration locally first, without affecting the live site. Now you have to create a virtual machine with the same parameters of your VPS package!

Open the VMWare Player application and choose the Create a New Virtual Machine option!

When you set up the virtual machine using the wizard:

* Do not set installation media („I will install the operating system later”)!
* Choose Linux operating system and Ubuntu!
* Create the disk file of the virtual machine to the local drive of your computer!

**Take a screenshot of your new virtual machine!**

## 3. Install the operating system

Because of the limited resources of your VPS and the low budget you cannot install a commercial OS on your virtual machine like Windows. There is a wide variety of Unix based operating systems you can choose from: most of them are open source and free to use, and have very low hardware requirements. Your choice is the Ubuntu 14.04 which is the latest LTS (Long Term Support) version of the Ubuntu operating system.

To start the Ubuntu installer:

* Download the ISO installer disc image file from the website (**Ubuntu 14.04 LTS mini.iso**)!
* Start your virtual machine!
* In VMWare Player open the Player menu and the Removable devices submenu, and choose CD/DVD and Settings!
* Set the „Use ISO image file” parameter to the downloaded ISO image file!

After saving your settings reboot the virtual machine!

**Take a screenshot of the main menu of the installer disc!**

After booting from the install disc ISO, select Install!

**Select a language**

Choose English – English!

**Select your location**

This question is related to the location of the installed system. As English language was chosen previously English speaking locations are offered by default. Choose other from the original list and select the Europe and Hungary options!

**Configure locales**

Select United States – en\_US.UTF-8.

**Configure the keyboard**

Answer No to the keyboard detection question and select English (US) on the next two screens. This setting will only affect the system console: for SSH connection the local settings of the host running the SSH client will apply.

**Configure the network**

You have to provide a hostname for your computer. As this server does not have a FQDN, you could select any name you like. For the laboratory, use your oslab as your host name!

**Choose mirror of the Ubuntu archive**

The closer the mirrors are the faster downloads will be thus select Hungary and hu.archive.ubuntu.com from the options. When asked for proxy, leave the field empty (use TAB to leave the field) and choose Continue!

**Set up users and passwords**

You have to set up your first user. The full name must be student, the login name must be student and the password must be ospass! The system informs you that the password is too simple, but for the laboratory it will be comfortable to use a simple password, thus select Yes to use the weak password.

You are asked if you want to encrypt your home directory: answer No!

**Configure the clock**

Verify that the system offered the right time zone for the location of the server. If the setting is correct answer Yes, otherwise set the correct time zone!

**Partition the disks**

For partitioning the disk choose the Guided – use entire disk option: it will automatically set up all necessary partitions and the decisions made by this pro gram are just fine for the current requirements.

The system finds the only hard drive of the virtual machine under the name of SCSI33 (sda), so confirm this selection to move on. The installer asks for one more confirmation before writing the changes to disk, so answer Yes!

**Configuring tasksel**

The installer suggests to keep automatic updates off and to take care of package updates manually. In our scenario security must be in focus, thus you have to enable update by selecting Install security updates automatically.

**Software selection**

The installer is able to set up the new server for typical tasks by installing the related packages. Now no packages should be selected for installation (not even Basic Ubuntu Server), as we will install all necessary packages by hand later. To leave the selection list without selecting any item press Tab and move on by selecting Continue!

**Install the GRUB boot loader**

The installer can not found any other operating system on the disc, thus it suggests installing the GRUB boot loader into the Maser Boot Record of the disks. As there are no other systems to boot you can safely answer Yes!

**Finish the installation**

The installer asks if the system clock is set to UTC. Answer Yes as in most cases this is the situation. Now you are at the end of the installation. Select Continue to reboot the virtual machine, and to boot the newly installed Ubuntu operating system.

After rebooting the virtual machine log in with the student name and ospass password.

**Insert a screenshot of the console after your first login!**

Your first task is to check for available software updates. To do so execute the sudo apt-get update command to synchronize the local package database with the central servers. After that run sudo apt-get upgrade to install every available updates. Most likely there will be no new updates as the installer already installs the latest packages, but it is always good to make sure.

**Create a screenshot of the output of apt-get upgrade!**

## 4. Installing an SSH server

Accessing the physical console of a server computer is uncomfortable and in many cases it might be impossible as many servers do not have keyboard or display attached. To allow remote access you have to install an SSH (Secure Shell) server!

To install the necessary packages execute sudo apt-get install openssh-server!

After the installation you have to be aware of the IP address of the virtual machine to initiate an SSH connection. To determine the IP address enter the ip a command and look for the **eth0** interface and the **inet** parameter!

**Make a note of the IP address of the virtual machine here!**

Start an SSH client application (e.g. **PuTTY**) on your host computer and enter the IP address from above to the host field. When prompted for login use the student name and ospass password just as if you were using the console directly.

**Insert a screenshot of the SSH connection after logging in!**

From now on we suggest you to work through this SSH connection, as you can easily copy command from this guide into the **PuTTY** terminal window: to paste a command previously copied to the clipboard simply click right with your mouse in the **PuTTY** window! If necessary, you can open multiple SSH connection to the same server.

## 5. Installing a webserver

The server was created to host a website, thus you need to install a web server application

To install the most widely used web server application for Linux named **Apache2** execute the sudo apt-get install apache2 command! After finishing the installer open a web browser on your host machine and enter the IP address of the virtual machine into the address field. The default installation page of **Apche2** should be displayed by your browser!

**Take a screenshot of the default page in your web browser!**

As the default page says the content you see is served from the **/var/www/html** folder. To list the contents of this directory use the ls -la /var/www/html command!

From the output of **ls** you can see the name of the file being displayed in the web browser, and it is also visible that all files in the **html** directory are owned by the root user and the root group. From now on the content of this directory will be maintained by the student user, so you have to make this user the owner of the directory to grant the necessary permissions. To do so execute the sudo chown -R student:student /var/www/html command!

The **chown** command as most Unix commands does not display any output when the operation was successful, only error messages are shown when needed. To ensure that the permissions have been successfully updated execute the ls -la /var/www/htmlcommand again!

**Create a screenshot of the contents of the html directory displayed by the ls command!**

## 6. Upload the website

As the designer has not yet finished the final website only a basic template is provided to you to replace the default apache website. You can download this template from the website in an archive named **static-html-website.zip**.

Use a web browser on your host computer to download the static HTML website ZIP archive!

Most SSH servers are set up by default to also provide SFTP access for the users. Use the **WinSCP STFP client on your host computer to connect to the virtual machine: use the IP address of the virtual machine as host for the connection and your login credentials used in SSH.**

**When the SFTP connection is set up, navigate to the** **/var/www/html** folder and upload the **static-html-website.zip** file to the server!

To extract the uploaded ZIP archive navigate on your SSH shell to the html folder by executing the cd /var/www/html command. To extract the archive enter the unzip -o static-html-website.zip command!

The unzip operation fails. Why? Examine the error message, and solve the problem!

**What was the problem and how did you solve it?**

Extract the archive after solving the problem with the original unzip command. You can recall previous commands in the shell by using the up/down arrow keys on your keyboard. After unzipping reload the page in your browser to see the changes!

**Take a screenshot of the static website in the browser served by your server!**

To clean up, delete the ZIP archive from the server by executing the rm /var/www/html/static-html-website.zip command!

## 7. Setting up a webshop

The product of your startup is successful and the production has just been started. To provide the product directly to the customers the company decides that you have to create a web shop. From the wide variety of open source web shop systems you choose the OpenCart system, that has the following system requirements:

* Web Server (Apache 2 suggested)
* PHP 5.3+
* Database (MySQLi suggested)

Your task is to install the necessary system components, download the OpenCart archive from the website and install it into your server.

#### 7.1. Installing PHP

The **Apache2** server installed is only able to server static files by default. To enable PHP based dynamic content you have to install and enable the PHP apache module.

To install the module execute the sudo apt-get install libapache2-mod-php5 command! Before confirming the installation take a look at the dependency packages displayed by the package manager on the **The following extra packages will be installed** line: here you can see that some other dependency packages will be installed as well, such as the **php5-common** main PHP package, the **php5-cli** console interface, etc. Confirm the installation by pressing Enter!

The package installer also enables the module. To check if PHP is understood and processed by our Apache server, enter the following echo "<?php phpinfo(); ?>" > /var/www/html/test.php command:

* **echo** writhes its arguments to its standard output
* the **>** redirect the standard output of its preceding command to file
* thus the PHP code in quotation marks will be written into the **/var/www/html/test.php** file.

Check if the file has been created successfully by looking at the output of the less /var/www/html/test.php command. To quit from the **less** viewer press the q key. To check if PHP is working on the server enter the virtual machine’s IP address in the address bar of a web browser on your host computer followed by /test.php! If all previous steps were successful then you should see the output of the **phpinfo()** function informing about installed PHP version and enabled modules.

**Insert a screenshot of the test.php in the browser!**

Check if the PHP version in the page header and the Apache version (search for **Apache Version** on the page) fulfills the requirements of OpenCart!

**Is the installed PHP and Apache version suitable for OpenCart?**

#### 7.2. Installing the MySQL database server

The last missing requirement of OpenCart is a MySQL database server. To install the server and the PHP interface as well execute the sudo apt-get install mysql-server php5-mysql command! The installer asks for a password for the MySQL root user: set it to dbpass!

When the installer finishes execute the mysql -u root -p command to open a connection to the server from the terminal. When prompted for password enter the previously set dbpass password! If the login was accepted the MySQL console is presented to you where standard SQL commands can be executed.

For OpenCart you need to create an empty database and to register a MySQL user to access this database. To do these tasks execute the following commands line by line in the MySQL console:

CREATE DATABASE opencart;

CREATE USER 'opencartuser'@'localhost' IDENTIFIED BY 'openpass';

GRANT ALL PRIVILEGES ON opencart.\* TO 'opencartuser'@'localhost';

To check if the database was created correctly enter the show databases; command into the MySQL console.

**Take a screenshot of the output of the show databases; command!**

Type exit to quit from the MySQL console.

#### 7.3. Installing OpenCart

To install OpenCart first you have to download the source code from the server into the html directory on your virtual machine.

1. Navigate to the html directory by entering: cd /var/www/html
2. Download the source code directly from the server to the virtual machine by executing: wget -O opencart.zip http://home.mit.bme.hu/~eredics/os/opencart.zip
3. Extract the archive: unzip -o opencart.zip
4. Remove the archive: rm opencart.zip

After downloading the source code open a web browser and enter the IP address of the virtual machine followed by the /opencart ending. The browser loads the OpenCart installer page. To move on with the installation accept the license conditions by clicking to Continue!

The second step of the installation checks all necessary PHP libraries. Some missing requirements are found among the libraries, so first these have to be installed and enabled. Execute:

sudo apt-get install php5-curl php5-gd php5-mcrypt to install the missing packages.

sudo php5enmod mcrypt to enable the necessary module.

sudo service apache2 restart to restart the web server.

Reload the page in the browser and if all required library is present select Continue!

In the third step enter the parameters of the previously created database and database user:

* Username: opencartuser
* Password: openpass
* Database: opencart

The OpenCart **admin** user should have a password of admin, and any e-mail address is fine for the setup. To finish the installation click Continue! Your newly created web shop should be up and running in the browser.

**Create a screenshot of the newly installed webshop!**

To finish the installation delete the installer directory from the virtual machine by entering the rm -r /var/www/html/opencart/install command.

## 8. Setting up a Windows file share

The designer of the team starts working on theming the new web shop. Most of the configuration can be done through the web interface of OpenCart but some theming requires file level access to the server. As the designer works on a Windows PC the most confortable solution would be to provide a Windows file share that can be mounted as network drive on that PC.

Windows file shares can be set up on Linux with the Samba application. To install Samba execute the sudo apt-get install samba command! After successful installation you have to create a Linux user for the designer: as you do not want him/her to log in through SSH the password and the shell access has to be disabled. Enter the sudo adduser --no-create-home --disabled-password --disabled-login sambauser command to create a user called **sambauser**! Any optional parameter asked by this command can be left blank.

The previously created user does not have a home directory or a password. To allow Samba access for him/her, you have to set a Samba password: enter sudo smbpasswd -a sambauser and set the password to sambapass!

The user configuration is done, now you have to create a share to be accessed by **sambauser**. To create a share you will use the **nano** editor to open the Samba configuration file by entering the sudo nano /etc/samba/smb.conf command! Scroll down to the bottom of the file by using the Page Down key and insert the following lines (if you use PuTTY you can paste the contents of the clipboard any time by right clicking into the PuTTY window):

[opencart]  
 path = /var/www/html/opencart  
 browseable = yes  
 read only = no  
 guest ok = no  
 valid users = sambauser

To exit **nano** press Ctrl+X and save your modification with the Y and Enter keys. To make Samba reload its configuration enter the sudo service smbd restart command!

If your host computer is running Windows 10, there are a few additional steps to do (for earlier versions of Windows you can skip to the next paragraph):

* Open Start menu -> Run -> Enter “cmd” without quotes and click Run
* Enter the following command into the Windows command shell: net use \\{IP address of your server} /u:sambauser, and press Enter
* Enter the sambapass password if prompted
* Close the Windows command shell.

To test your new share from your host machine select the Start menu and Run. Enter \\ followed by the IP address of the virtual machine. Windows should display the newly created opencart share on the virtual machine. When you try to open it you must enter the previously set sambauser and sambapass to identify yourself and to see the contents of the selected share.

**Create a screenshot of the shared directory opened from the host machine!**

## 9. Setting up a firewall

The previously created Windows file share works fine, the only problem is that it accepts connections from all over the Internet. As the designer only wants to connect to the server from the local network it would be more secure to restrict access to the Sambas server from the local network. To do so you have to install a firewall.

The most used firewall in the Linux environment is called **Iptables** and is present in Ubuntu as well. To list the active firewall rules enter sudo iptables -L in the terminal! The output of the command shows no active rules at the moment.

To list the open ports on the virtual machine enter the sudo netstat -lpn4 command!

**Create a screenshot of the list of the open ports!**

In the **Local Address** column three different addresses are present:

* 0.0.0.0: ports open on this address are available from all networks
* 127.0.0.1 (localhost): ports open here are available only from the local machine
* the external IP: ports open here are open on this interface of the virtual machine only

You can see ports 22, 139 and 445 open towards the Internet. The **netstat** output also shows that port 22 belongs to the SSH server, thus it should be left open. Ports 139 and 445 are Samba related and access should be restricted to the local network.

Configuring directly **iptables** is a complex task, but to make it simple Ubuntu has a frontend called **ufw** (Uncomplicated Firewall). First execute the following **ufw** commands:

sudo ufw allow ssh

sudo ufw allow http

sudo ufw allow from 152.66.0.0/16 to any app samba

sudo ufw enable

The first two commands define allow rules for SSH and HTTP for all clients. The third rule allows Samba access only from the local 152.66.0.0 subnet. The last command enables the firewall.

If you enter sudo iptables -L the **iptables** representation of the previous rules generated by **ufw** can be examined. The much more readable **ufw** rules can be printed by the sudo ufw status command.

**Create a screenshot of the output of the ufw status command!**