Assignment 5

Home Server/NAS – Final assignment

OSYS 1000

Mike LeBlanc

Contents

[Introductions: 3](#_Toc132319804)

[Installing Ubuntu/initial setup: 3](#_Toc132319805)

[SSH: 3](#_Toc132319806)

[Firewall: 5](#_Toc132319807)

[File Transfer Protocol: 5](#_Toc132319808)

[I LOVE L.A.M.P! 7](#_Toc132319809)

[Apache: 7](#_Toc132319810)

[MySQL: 8](#_Toc132319811)

[PHP: 8](#_Toc132319812)

[Plex Server: 9](#_Toc132319813)

[Conclusions 11](#_Toc132319814)

# Introductions:

At last. The final project of this course, the ‘pièce de résistance’ if you will. In this document, I will walk you through the steps necessary to integrate your very own home server to host your own website within your walls, create a place to store files and pictures, protect the server with its own Firewall, access the server from another computer through SSH, and create a media server using PLEX.

# Installing Ubuntu/initial setup:

First, we need to get ourselves a nice Operating System to run our server. For this, we’re going to get the latest version of Ubuntu. Here’s a handy link!

https://ubuntu.com/download/server

Hit the download button, and we’ll get started in VMware to house this.

For VMware installation, we will set our configuration as follows:

* CPU: at least 2 cores
* RAM: at least 4GB of ram
* Storage: At least 50GB of storage (more if you want to house more films in plex)
* Network: Bridged (this will be so we can access our server via an IP address

You’ll want to name your server appropriately. For myself, my username will be ‘mleblanc’ and the server’s name will be ‘lx01ml’ for linux server, 01st rack (virtual), ml my initials.

Once this setup concludes, you should see the server sitting idle, waiting for you to sign in as yourself.



Figure : What you should see upon first login/boot.

You’ll want to login using the credentials you set earlier. Once we’re in, let’s go ahead and update and upgrade our Ubuntu OS, using two commands.

‘sudo apt-get update -y’  
‘sudo apt-get upgrade -y’

The ‘-y’ just preemptively agrees to update and upgrade the system. Once this is done, restart, and we can get into the meat and potatoes of the server.

# SSH:

Now, SSH is simple and easy, and comes pre-installed on most modern flavors of Linux. To check to see if ours has this wonderful feature, we will type:

‘sudo systemctl status ssh’

And once our password is accepted, we get this lovely message:

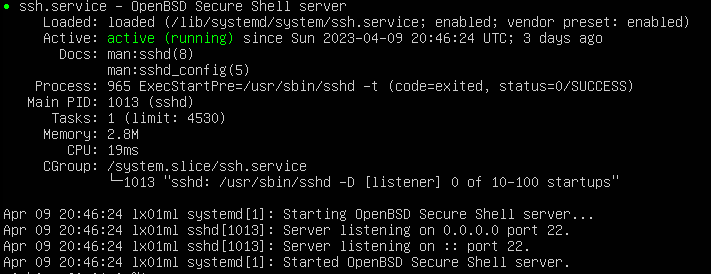


Figure : SSH is operational!

However, if this doesn’t exist, we would type:

‘sudo apt install openssh-client’

And now we can access our server from another terminal by typing:

‘ssh [mleblanc@192.168.2.79](mailto:mleblanc@192.168.2.79)’ in my case OR

‘ssh username@IPaddress’

Now, if you don’t know your IP address, we’ll install net tools with:

‘sudo apt install net-tools’

Now we can run if-config and it will show us our IP address we’ve been assigned.

Also, don’t forget to create a rule for the firewall to allow the connection of SSH to the server.

‘sudo ufw allow ssh’

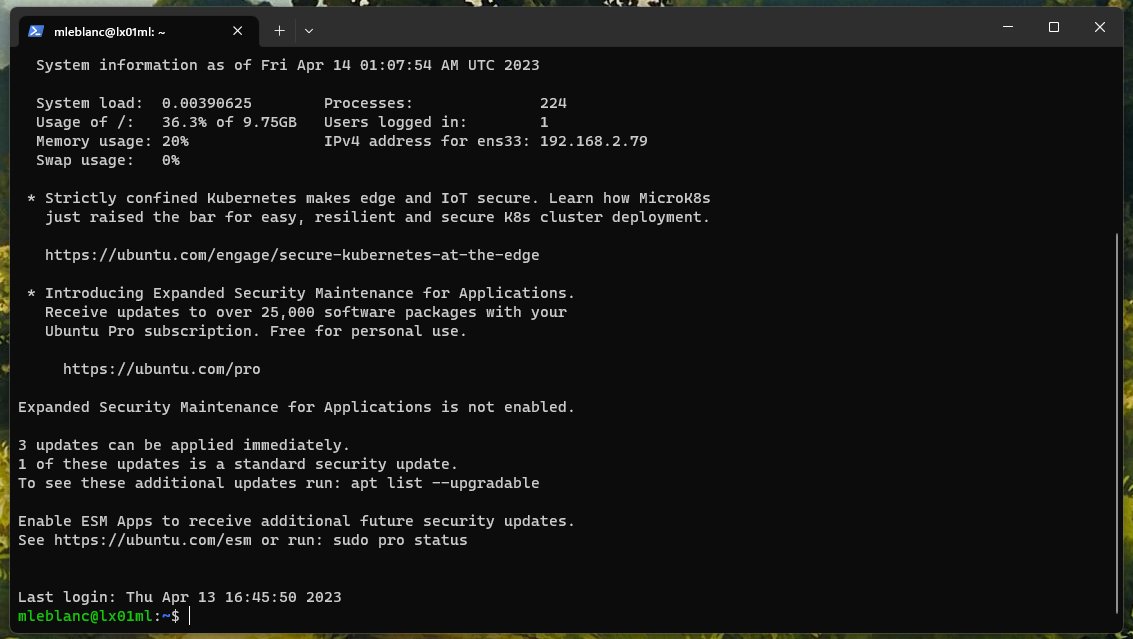


Figure : Connection to Ubuntu Server through Windows PowerShell

# Firewall:

The uncomplicated firewall is something else that comes already prepared, but not enabled. First, we’ll enable the firewall, then allow outgoing connections, and deny incoming connections. This will allow us to open only the ports we need.

‘sudo ufw enable’

‘sudo ufw default allow outgoing’  
and  
‘sudo ufw default deny incoming’

Now, as we install our other applications, we will allow ports through the firewall. We already have SSH set up to be allowed through.

# File Transfer Protocol:

Having a place to throw all the junk pictures and videos that you may need later is a more useful tool than you might imagine. Using this tool is also much quicker, easier, and way less of a headache than using a thumb drive.

To start, we’ll go ahead and run this command:

‘sudo apt install vsftp’

Now comes some configuration. We’ll want to keep users in their own personal folder, and allow users to upload files, in our case, that’s any laptop or other pc in the house. That file is /etc/vsftp.conf. Using Vi, we’ll uncomment **chroot\_local\_user=**YES, also we’ll want to add **write\_enable=YES** to the file. Before we quit, we’ll ensure a little more security. In the same file, we’ll add **ssl\_enable=YES**. This ensures that only people with a viable key will be able to access FTP.

We’ll now restart vsftp to ensure changes take effect with:

‘sudo systemctl restart vsftp.service’

For us to be able to access the FTP server, we are going to allow ports through the firewall.

‘sudo ufw allow 20/tcp’

‘sudo ufw allow 21/tcp’

Sudo ufw allow 5000:6000/tcp’

Port 20 and 21 allow us to connect to the FTP server, and 5000 through 6000 are for files to pass through to the client.

Now, on the client, (I suggest Filezilla) we can send files to and from our server.

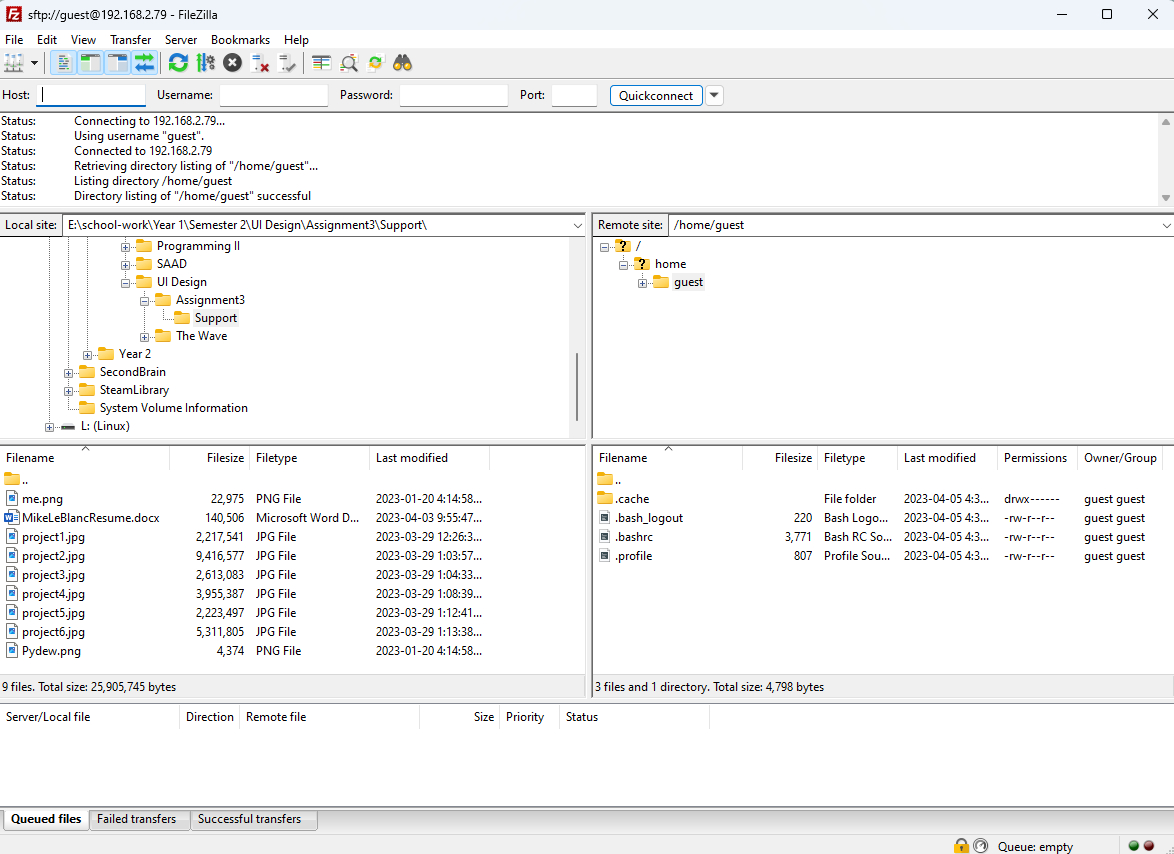


Figure : Personal connection to server with FileZilla

# I LOVE L.A.M.P!

Great movie! Anchorman was a classic for Will Farrell. Anyway! By using this stack, we can easily have a web developer’s dream setup! This powerful and flexible stack is what started the likes of Facebook, Instagram, and Twitter. We already have the L, which is Linux, now we need the A.M.P:

- Apache

- MySQL

- PHP

Now, we can install all these separately, but there’s an easy way!

‘sudo apt install lamp-server’

Now, we have a bit of setup, so we will start with A and end with P.

### Apache:

To start off, we’re going to want to install Apach2 with the command:

‘sudo apt install apache2’

Don’t forget to allow http and https through the firewall!

‘sudo ufw allow http|https’

Now, we could type in our IP address that the server possesses, and we’d see the Apache default page like so:

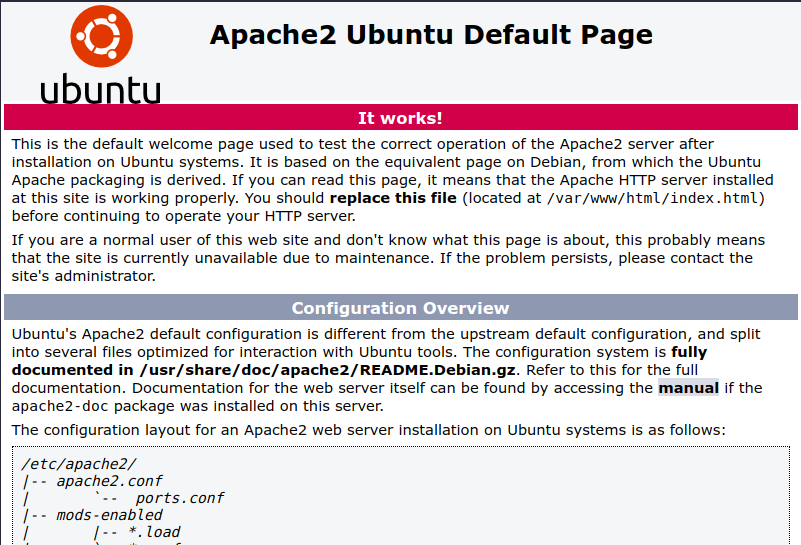


Figure : Default Apache2 page.

Now, this is cool and all, but let’s set us up so we can display our portfolio’s we made in another class. We have FileZilla installed now, so let’s get those website files uploaded and we can move them. We are going to want to move the files from our FTP folder, to /var/www/html.

Now, go ahead and reload the page that we seen the default apache2 page. See what happens!

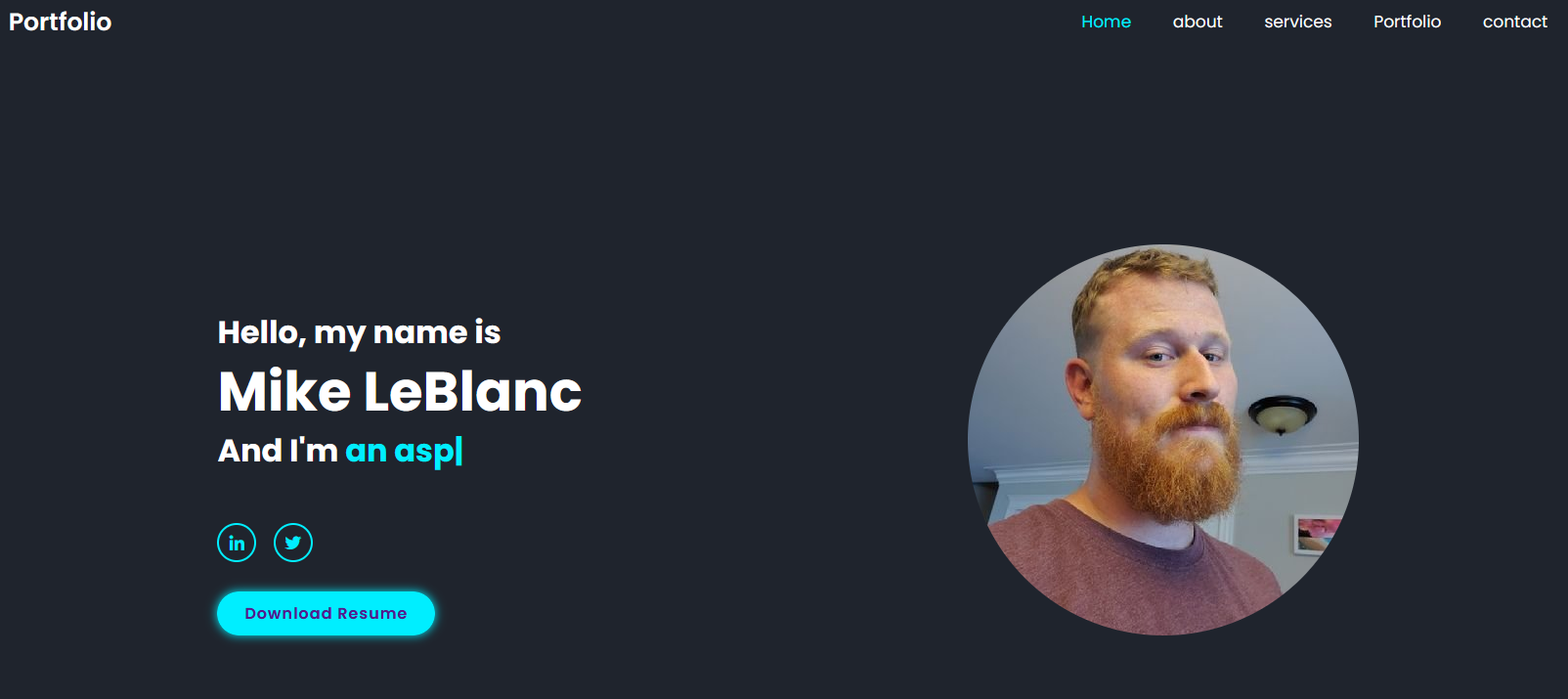


Figure : Personal portfolio on web browser

### MySQL:

With this, we’re going to want to set a password for the root user before we do ANYTHING. This has caused me a lot of headaches and using my snapshot to get it right.

‘sudo mysql’

Now that we’re in the mysql menu, we’re going to set a password for the root user.

ALTER USER ‘root’@’localhost’ IDENTIFIED BY ‘NewPassword’;

Now we can quit and run the secure installation.

‘sudo mysql\_secure\_installation’

Now go through this installation, and when prompted, keep the current password, and say yes to the rest. Once completed we can create a user for ourselves and give us all permissions without signing in as the root, (although that’s still an option and the one I’ve opted for).

In order to sign in, we need to type up:

‘mysql -u UserNameHere -p’

Now enter password.

Same thing for the root. We can enter:

‘mysql -u root -p’

### PHP:

To get started, we will need to install php. This integrates well with MySQL and Apache 2 to access both the web server and the SQL database.

‘sudo apt install php libapache2-mod-php php-mysql’

This installs the prerequisite for php, and the modifications for linking apache2 and MySQL. We just need to restart Apache2 again to ensure the changes have been made and recognized by the daemon.

To test this out, we’d move our personal website away from /var/www/html and instead write a test file for php.

‘sudo vi index.php’

Within this file, we’ll write out the following script to test out everything.

<?php

Phpinfo()

?>

After saving and quitting, reload the portfolio website, and you should see the php information page that outlines versions of php, apache, and MySQL.



Figure :Heading for the latest PHP release.

# Plex Server:

Now, using plex, we can store and replay music and movies that we have bought. With this being a HOME SERVER, we won’t be touching those scary copyright laws.

Let’s go ahead and install Plex.

‘sudo snap install plexmediaserver’ (snap is the other package manager on Ubuntu)

That’s it! Well, other than that pesky firewall. Let’s take care of that. Let’s create a UFW application profile by adding it to the ufw applications folder.

‘sudo vi /etc/ufw/applications.d/plexmediaserver’

Within vi, we’re going to copy/paste this:

[plexmediaserver]

title=Plex Media Server (Standard)

description=The Plex Media Server

ports=32400/tcp|3005/tcp|5353/udp|8324/tcp|32410:32414/udp

[plexmediaserver-dlna]

title=Plex Media Server (DLNA)

description=The Plex Media Server (additional DLNA capability only)

ports=1900/udp|32469/tcp

[plexmediaserver-all]

title=Plex Media Server (Standard + DLNA)

description=The Plex Media Server (with additional DLNA capability)

ports=32400/tcp|3005/tcp|5353/udp|8324/tcp|32410:32414/udp|1900/udp|32469/tcp

Thank developers for copy/paste every day.

Now! Update the profile with:

‘sudo ufw app update plexmediaserver’

This updates UFW and it recognizes there is a new profile made. And we apply those rules with:

‘sudo ufw allow plexmediaserver-all’

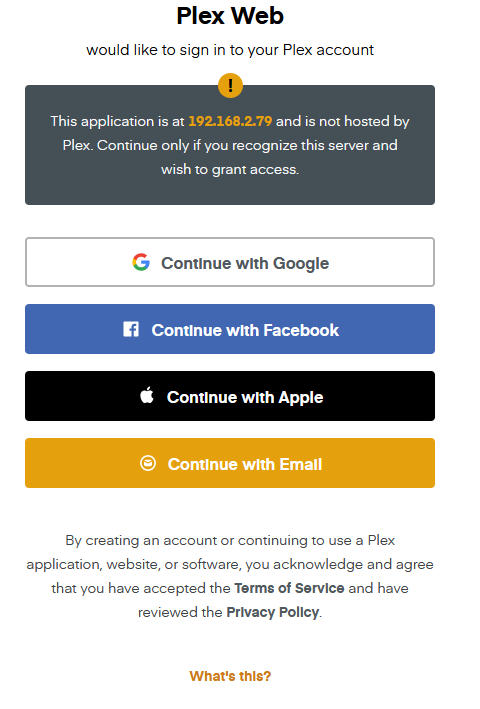


Figure : Plex Media Server as accessed by personal browser.

# Conclusions

This was by far the best assignment! Taking our struggles and turning that into an application that will be used by many of us! I hope my son will get as much enjoyment with getting the computer built, and seeing what his work produces as I have building this server within a virtual environment. This document will be travelling with me as I add to it as iterations of this server come to pass.

Thanks for the great semester. I’ve learned an immense amount about Linux and will be making the jump shortly.