Container Server Knowledgebase

Articles for Ubuntu Server 18.04/LXD, Samba, and LAMP/MediaWiki

Mitch Jean – W0414274 (Team 3)
Network Services Using Linux – OSYS3030
November 27, 2019

Table of Contents

Introduction

The following document is the concatenation of two knowledgebase articles centered around getting a Linux server, Ubuntu in our case, up and running with a couple of LXD containers quickly and easily. No idea what those are? That's fine! I'll briefly walk through the steps here, and then I'll get into more granular detail (with some screenshots) later when I step through the processes within their respective articles. The steps taken will *mostly* be relevant to a server farm environment, but there will be valuable information for *mostly* everyone.

Note – due to formatting, it may be necessary to zoom in on certain screenshots in this document where the terminal text is particularly small

Here's the plan

Installing Ubuntu Server 18.04 – for this one, I'll assume you know (or can learn) how to enter your server's BIOS to boot from a USB stick, if needed

Installing LXD, creating containers – Linux Containers, the foundation this document is built upon! We'll get into much more detail later

Setting up a bridged interface – this will allow your containers to bridge the connection of your host, meaning they'll be on the same network. I'll also go over how to assign the bridged interface you set up to your container(s)

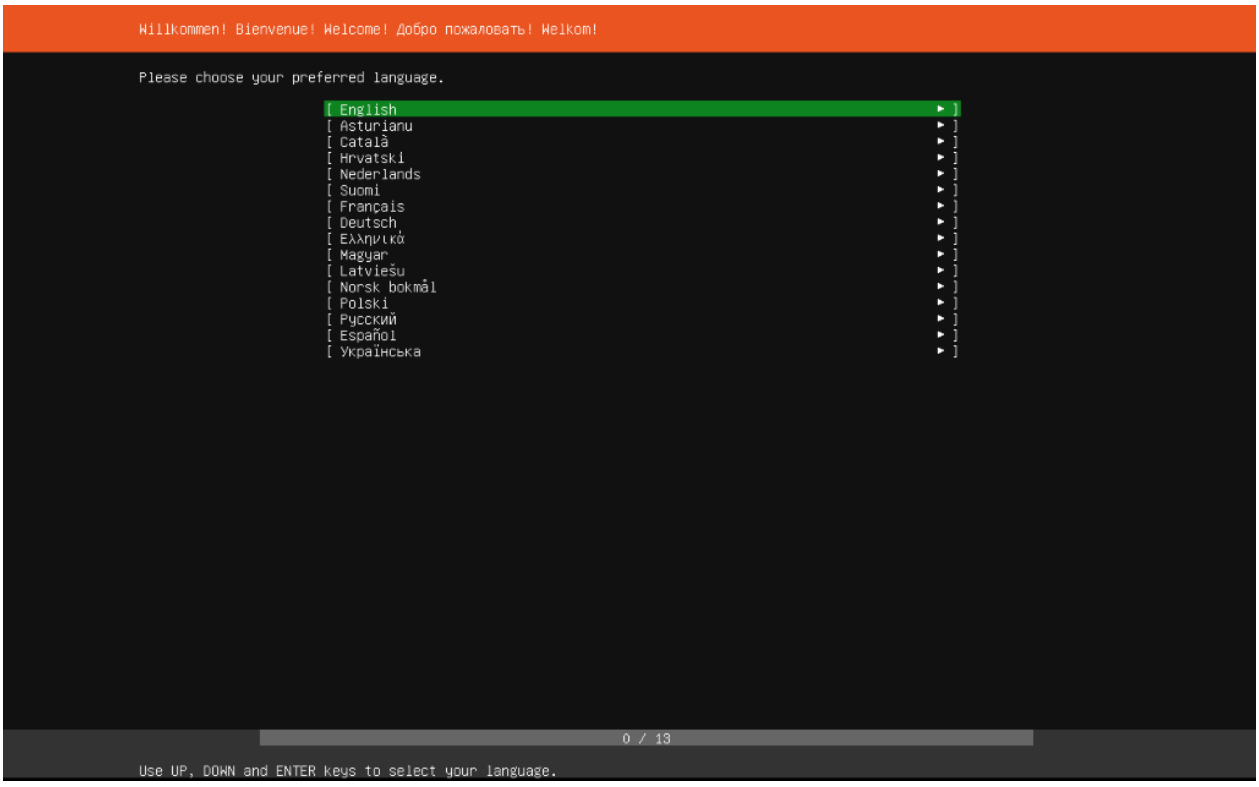
Configuring containers – the bulk of the content lives here. I'll be going over how to set up a simple Mediawiki website on a *LAMP* (Linux, Apache2, MySQL, PHP) stack, as well as installing/configuring a *Samba* share, both of which are served in their own containers. Once again, if this sounds daunting, I'll be taking it step-by-step and trying to explain everything I'm doing as I go

Installing Ubuntu Server 18.04, LXD, Creating Containers and a Bridge

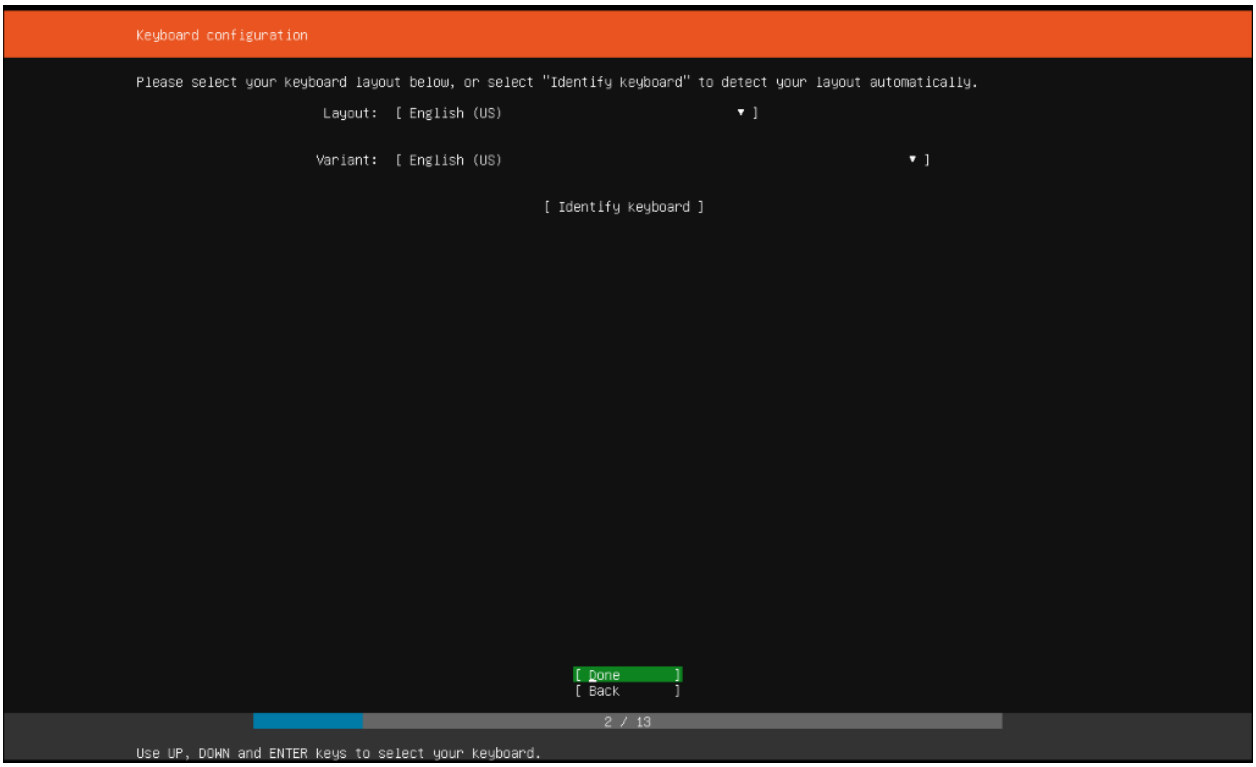
Step 1 – Insert your bootable USB stick containing Ubuntu Server 18.04 into a functional USB port in your system. If you're unfamiliar with how to create bootable media, I'll defer to this tutorial from the official Ubuntu website, which will walk through everything you need to know to get started. Come back here when you're done!

Step 2 – Boot up your system, or reboot if applicable. If your computer boots from the USB stick as intended, you will see some text scrolling down your screen, likely containing a lot of these: [OK] If not, you will need to figure out how to access the computer's BIOS and set it to boot using the USB stick you've created.

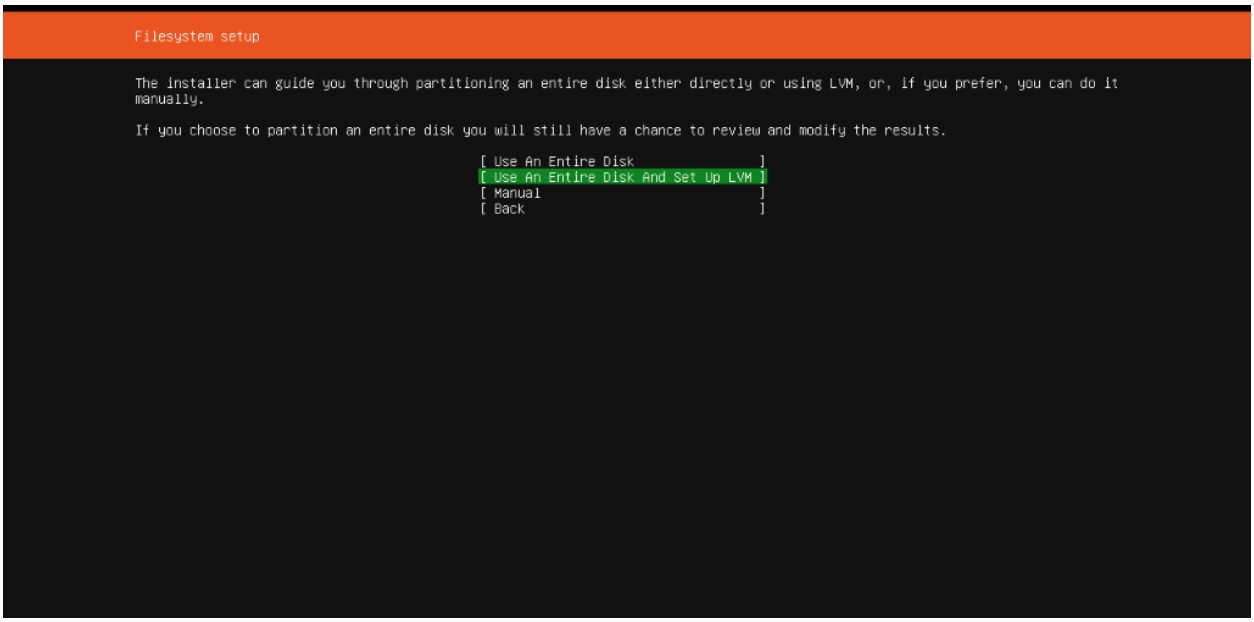
Step 3 – After the text is done scrolling, you should be greeted by the screen below. Use your keyboard (and the helpful directions at the bottom of the screen) to select the language you would prefer.



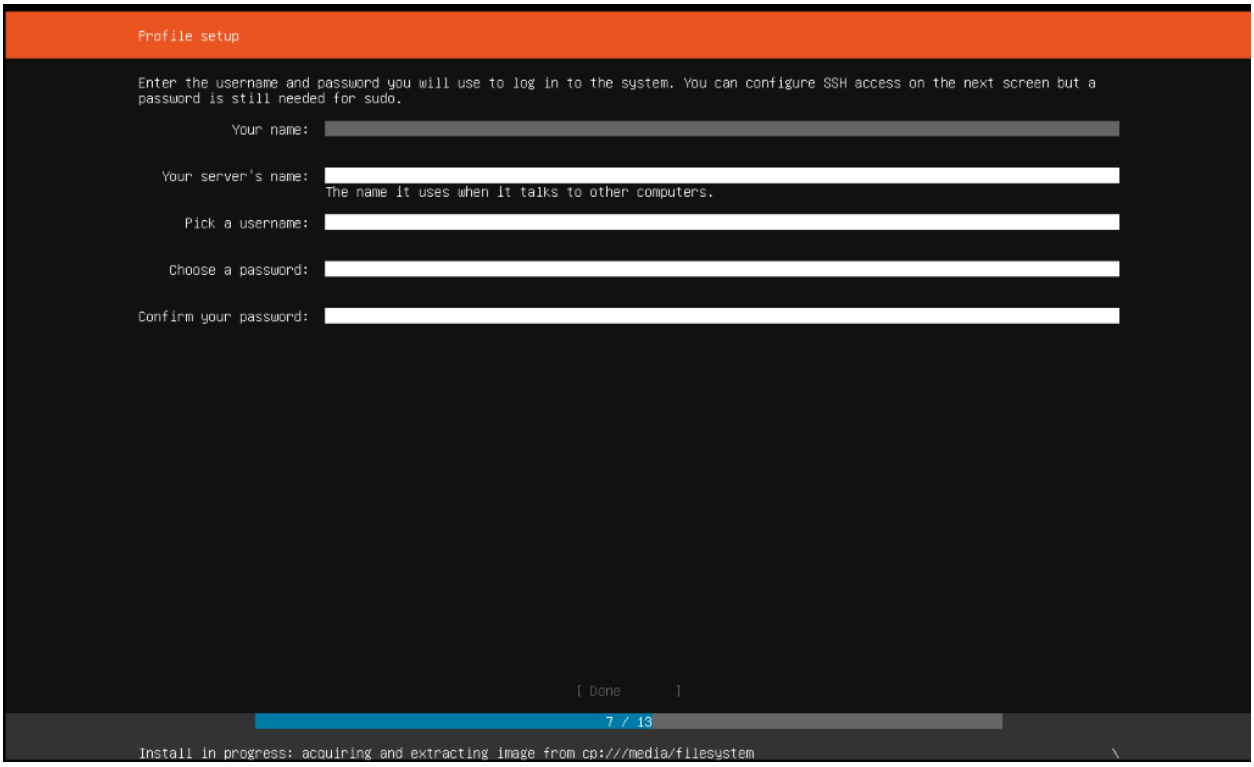
Step 4 –Next, select your preferred regional keyboard layout. After this, you will be prompted for network information: if it's getting a valid IP address via DHCP, you can proceed through. If not, you will have to set it statically within a valid network, or continue through the installer without an internet connection (ergo not receiving updates)



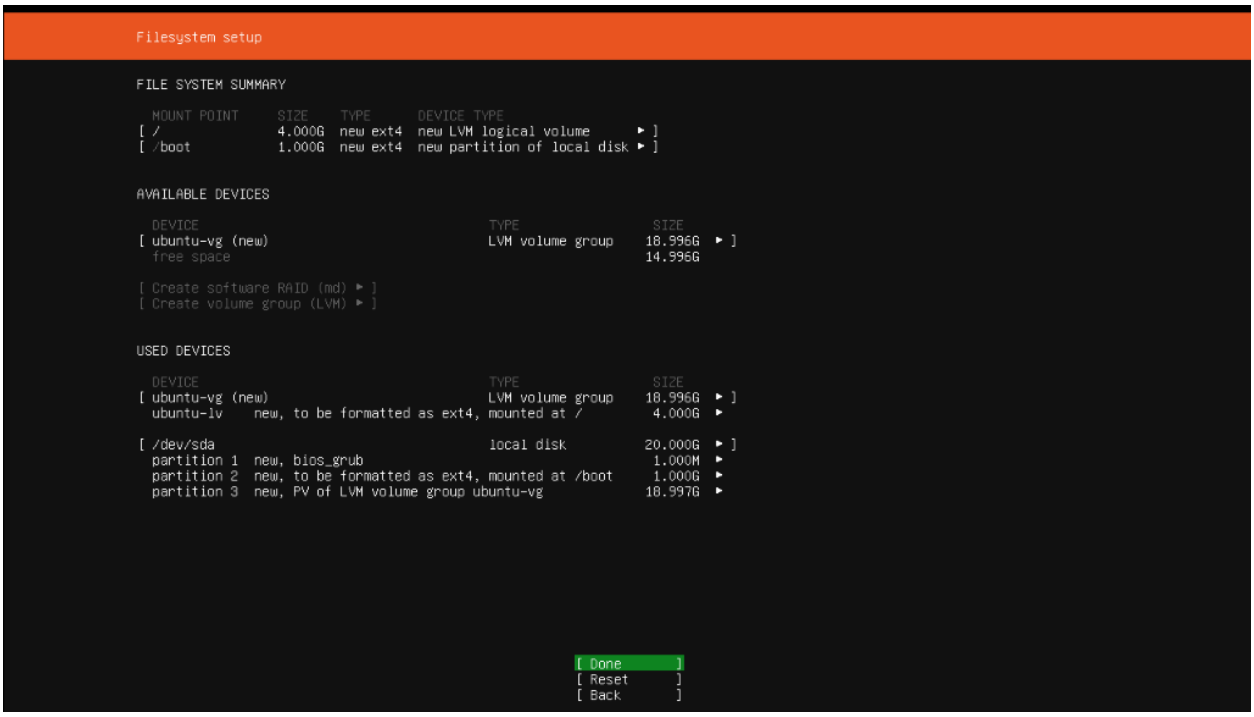
Step 5 – For the purposes of our environment, we'll be choosing 'Use An Entire Disk And Set Up LVM'. You should, too.



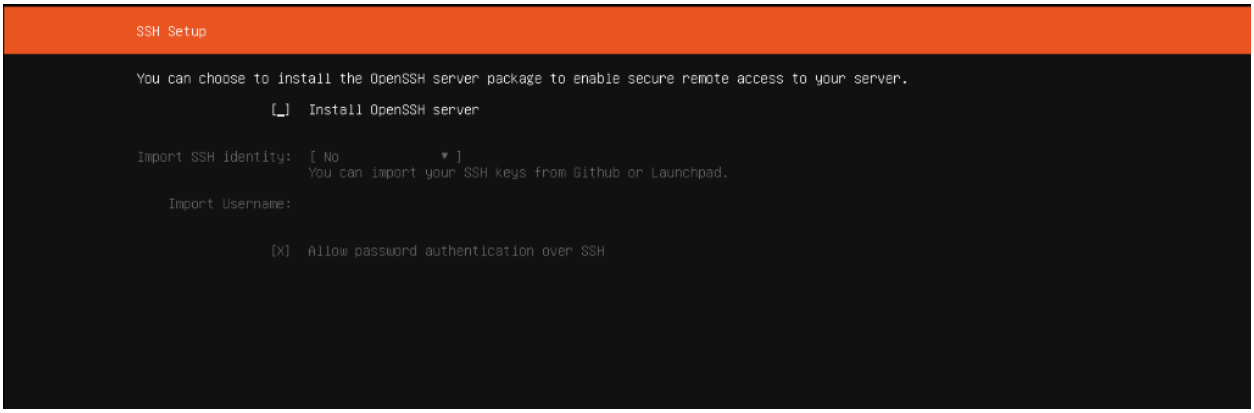
Step 6 – Previous to the screen below, you will be prompted to choose an installation. Select the default/top choice ‘Install Ubuntu’. Next, you will define a new user account, its password and your server’s name.



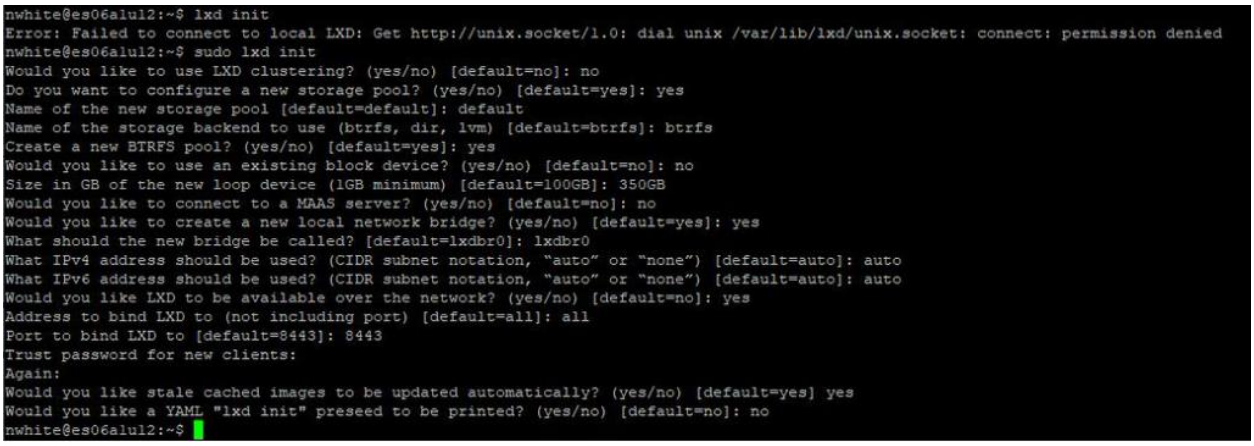
Step 7 – Select the storage device or partition where you’d like to install Ubuntu (hopefully this has been formatted/planned ahead of time), proceeding to the screen below and confirming your configuration



Step 8 – Optionally, install OpenSSH server. This will allow remote access through a command line or a terminal emulator like PuTTY using the credentials you set in Step 6. For more information, check out [this section from the official PuTTY FAQ page](#).



Step 9 – Next, use the command 'lxd init' after your Ubuntu installation has concluded, stepping through the wizard as pictured below. For us, we accepted all defaults except the capacity of the loop device, setting it to 350GB.



Step 10 – Create your first couple of containers! Simply issue the command:
`sudo lxc launch ubuntu:18.04 containername`

For our purposes, issue this command twice using a unique, descriptive name for both containers, as you'll need these in the coming steps when we configure LAMP and Samba!

Step 11 – set up a bridged interface that you will later assign to your containers, putting them on the same network as the host so that they can all communicate. This is explained and walked through very well in [this short guide from Open School Solutions](#), so follow their directions and come back here once you're finished.

Configuring a LAMP and MediaWiki Container

Step 1 – To assign our container to the network bridge we created in the last section/step, issue the following command:
`lxc config device add` *`containername interface`* `nic nictype=bridged parent=`*`bridgeinterface`* `name=`*`interface`*

Step 2 – Next, enter a Bash shell within your container, then install Apache2 using the following commands:

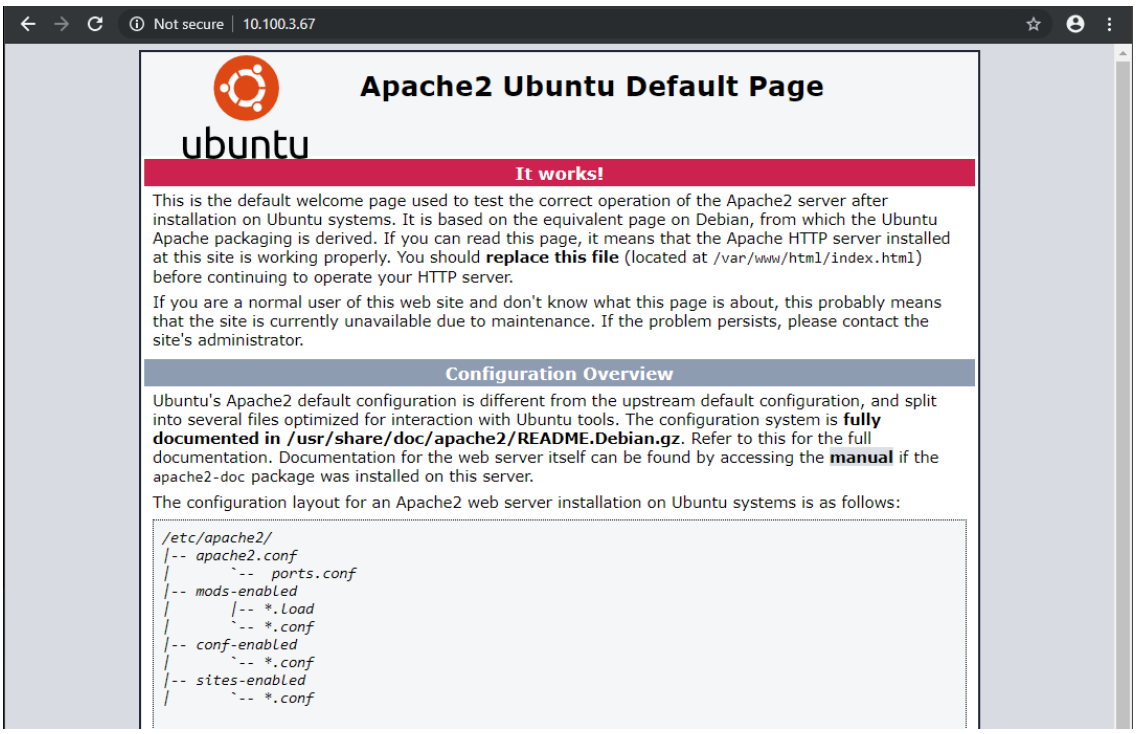
```
mjean@es06a1u12:~$ sudo lxc exec w0414274-lamp -- bash
root@w0414274-lamp:~# sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfreetype6
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
  ssl-cert
Suggested packages:
  www-browser apache2-doc apache2-suexec-pristine | apache2-suexec-custom openssl-blacklist
The following NEW packages will be installed:
```

Note: w0414274-lamp is the name of my container

Step 3 – Allow Apache through your firewall by issuing the following command:

```
root@w0414274-lamp:~# sudo ufw allow in "Apache Full"
Rules updated
Rules updated (v6)
```

Step 4 – To ensure your Apache installation was successful, navigate to your container's IP address in a web browser, and you should see this page:



Step 5 – Next, we'll install MySQL, set our validation policy/root password for connecting, then proceed through the remainder of the wizard using the (default) prompts below:

```
root@w0414274-lamp:~# sudo apt install mysql-server
root@w0414274-lamp:~# sudo mysql_secure_installation

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD plugin?

Press y|Y for Yes, any other key for No: Y

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary                  file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 0
Please set the password for root here.

New password:

Re-enter new password:
```

```
Estimated strength of the password: 50
Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : y
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : n

 ... skipping.


Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : n

 ... skipping.
By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : n

 ... skipping.
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : n

 ... skipping.
All done!
```

Step 6 – Now it's time for PHP! Use the following command to get PHP as well as helper packages to allow it to run on your Apache installation and work with your MySQL database:

```
root@w0414274-lamp:~# sudo apt install php libapache2-mod-php php-mysql
```

Step 7 – Next, open the following file with vi (or another text editor) using this command:

```
sudo vi /etc/apache2/mods-enabled/dir.conf
```

The file will look like the example below, but *index.php* needs to be the first string after *DirectoryIndex*, which is NOT default. After editing, your file should look like this:

```
<IfModule mod_dir.c>
        DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm
</IfModule>
```

Step 8 – To apply our changes, we'll need to restart the Apache service. Do this using the following command:

```
sudo systemctl restart apache2
```

Step 9 – MediaWiki relies on some PHP 'extensions' for it to be able to run. To install and enable these, use the following commands:

```
sudo apt install php7.2-xml
sudo apt install php7.2-mbstring
sudo phpenmod xml
sudo phpenmod mbstring
```

Step 10 – Let's get started on our MediaWiki site. Move to your /tmp directory within the container, grab the MediaWiki 1.33.1 tarball, then extract the contents using the following commands:

```
root@w0414274-lamp:/tmp# cd /tmp
root@w0414274-lamp:/tmp# wget https://releases.wikimedia.org/mediawiki/1.33/mediawiki-1.33.1.tar.gz
```

```
root@w0414274-lamp:/tmp# tar -xvzf /tmp/mediawiki-*.tar.gz
```

Step 11 – Next, we'll make a directory for MediaWiki under /var/lib, and then move the extracted contents from the last step into said directory using the following commands:

```
root@w0414274-lamp:/tmp# sudo mkdir /var/lib/mediawiki
root@w0414274-lamp:/tmp# sudo mv mediawiki-*/* /var/lib/mediawiki
```

Step 12 – Next, we'll create a MySQL user and a database for MediaWiki called 'my_wiki', then give our user using the following commands:

```
root@w0414274-lamp:/tmp# sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.28-0ubuntu0.18.04.4 (Ubuntu)

Copyright (c) 2000, 2019, 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 USER 'mjean'@'localhost' IDENTIFIED BY [redacted];
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
Bye
```

This is a password! Make one of your own.

```
root@w0414274-lamp:/tmp# sudo mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.28-0ubuntu0.18.04.4 (Ubuntu)

Copyright (c) 2000, 2019, 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 my_wiki;
Query OK, 1 row affected (0.00 sec)

mysql> use my_wiki;
Database changed
mysql> GRANT ALL ON my_wiki.* TO 'mjean'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

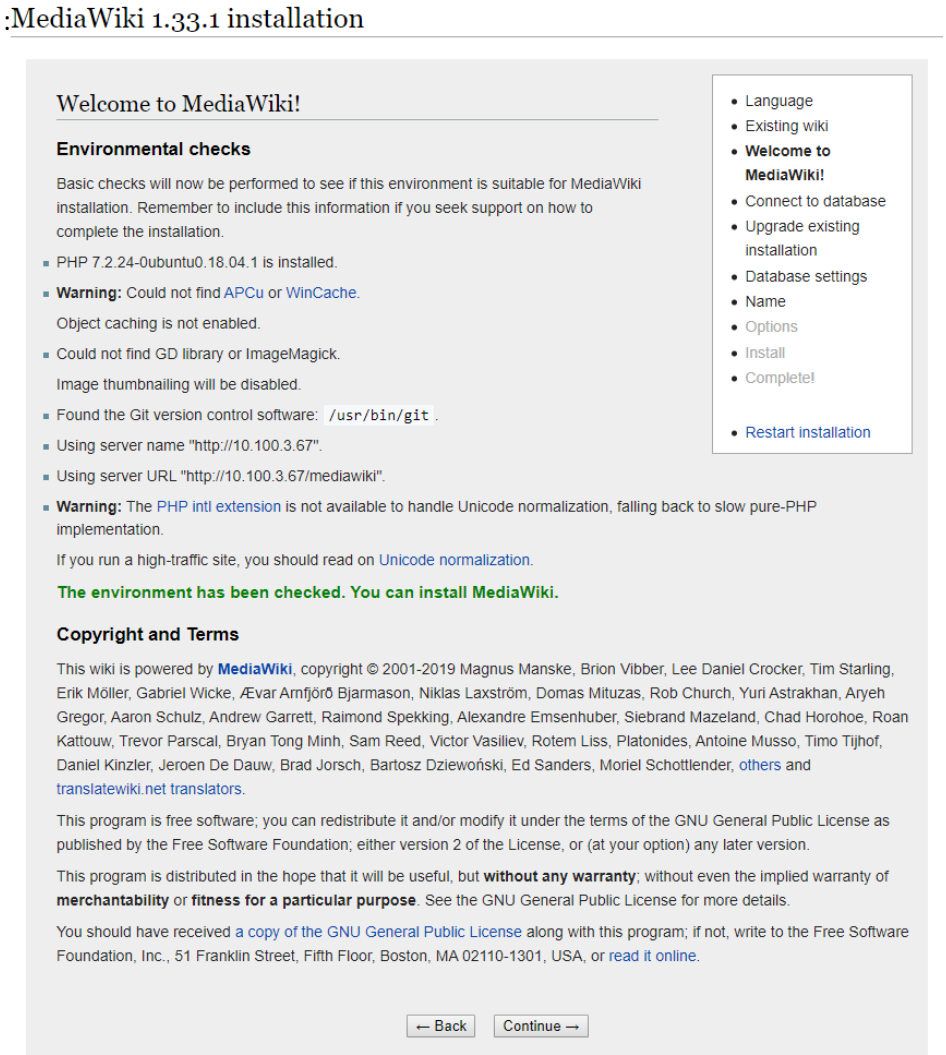
Step 13 – Next, make your container start on server boot using this command:

```
mjean@es06a1u12:~$ sudo lxc config set w0414274-lamp boot.autostart true
```

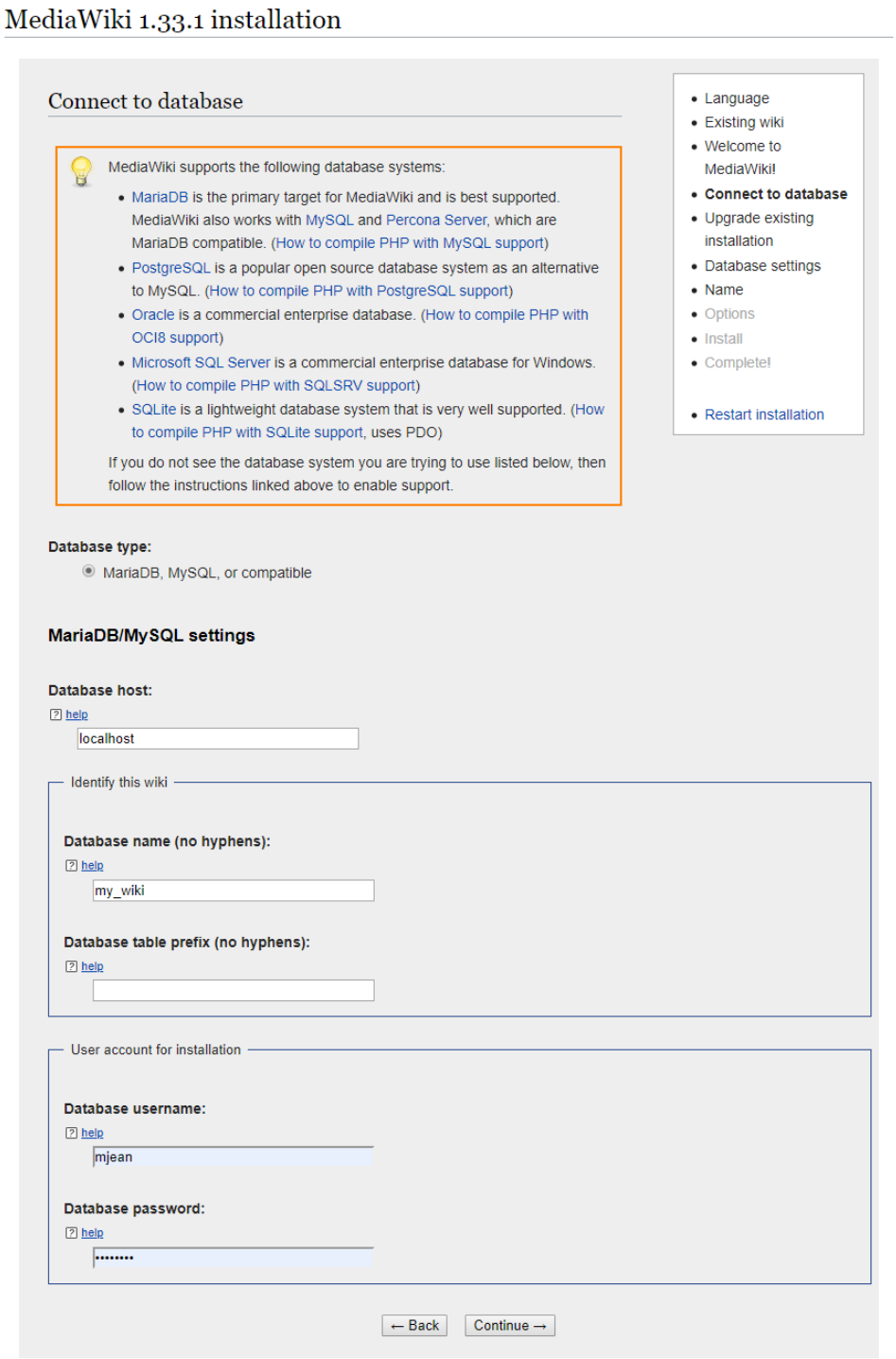
Step 14 – Time to get MediaWiki actually configured. Start by opening a web browser and navigating to the following address (where *containeraddress* is the IP for the container hosting your LAMP stack): **http://*containeraddress*/mediawiki**

You should be greeted with the below screen:

Step 15 – Click the link to 'complete the installation', and continue through this screen if you see "The environment has been checked. You can install MediaWiki."

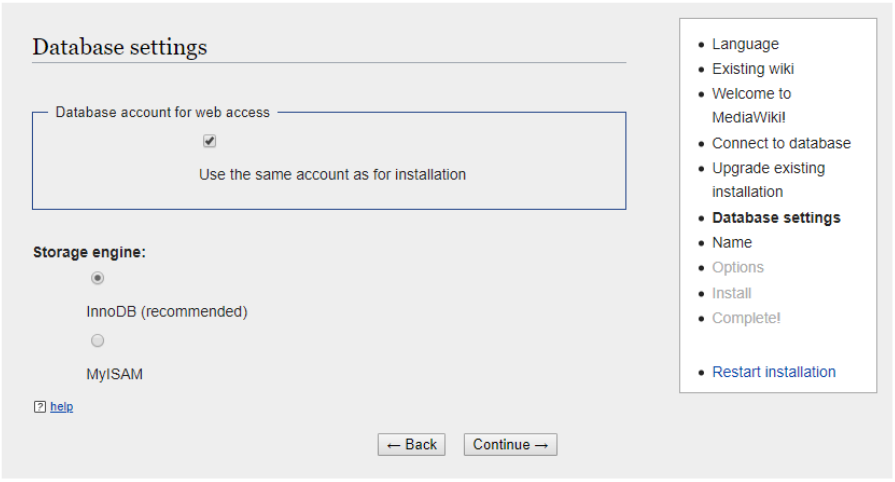


Step 16 – Fill out this form with your SQL database information/credentials, and when prompted, “Use the same account as for installation” as pictured below:

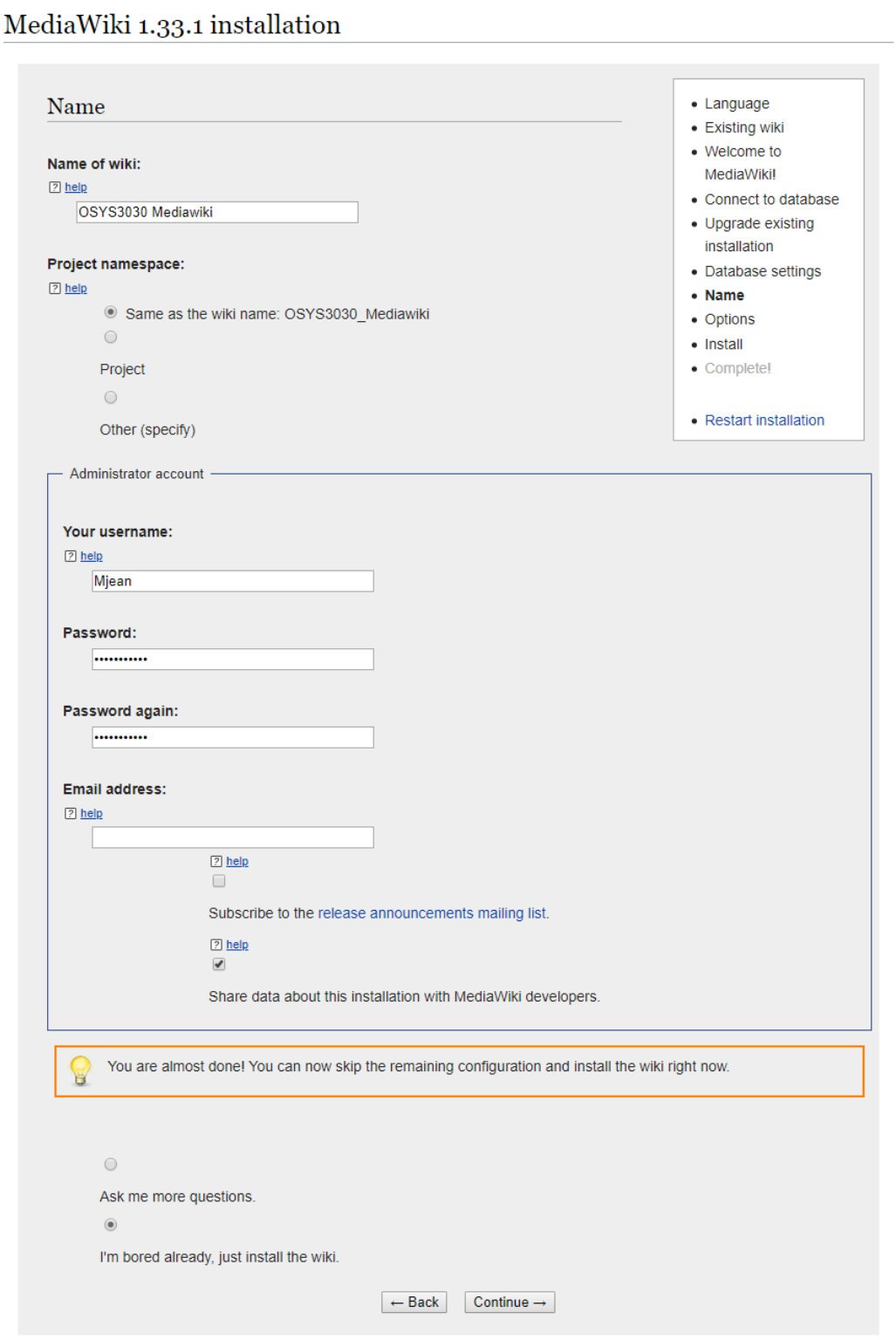


Step continues on next page…

MediaWiki 1.33.1 installation



Step 17 – Next we'll define some basic settings for our Wiki, such as a name and a user account/password to access the backend, as pictured below:



In my case, I was "bored already" and decided to skip the rest of the configuration.

Step 18 – MediaWiki uses a self-created file for its settings, which we'll need to download from our site and then insert it into the /var/lib/mediawiki/

To do this, download the file from using the button below:

Following this, (back in your container) use touch to place an empty file in the aforementioned directory and open it with a text editor using the following commands:

```
root@w0414274-lamp:/var/lib/mediawiki# touch LocalSettings.php
root@w0414274-lamp:/var/lib/mediawiki# sudo vi LocalSettings.php
```

Next, copy the contents of the LocalSettings.php we downloaded earlier (on the computer you accessed the file with), and paste them into the empty file you have open. After, your file should look like this:

root@w0414274-lamp: /var/lib/mediawiki

```
if ( !defined( 'MEDIAWIKI' ) ) {
        exit;
}
## Uncomment this to disable output compression
# $wgDisableOutputCompression = true;
$wgSitename = "OSYS3030 Mediawiki"; $wgMetaNamespace = "OSYS3030_Mediawiki";
## The URL base path to the directory containing the wiki; defaults for all runtime URL paths are based off of
## this. For more information on customizing the URLs (like /w/index.php/Page_title to /wiki/Page_title) please
## see: https://www.mediawiki.org/wiki/Manual:Short_URL
$wgScriptPath = "/mediawiki";
## The protocol and server name to use in fully-qualified URLs
$wgServer = "http://10.100.3.67";
## The URL path to static resources (images, scripts, etc.)
$wgResourceBasePath = $wgScriptPath;
## The URL path to the logo.  Make sure you change this from the default, or else you'll overwrite your logo
## when you upgrade!
$wgLogo = "$wgResourceBasePath/resources/assets/wiki.png";
## UPO means: this is also a user preference option
$wgEnableEmail = true; $wgEnableUserEmail = true; # UPO $wgEmergencyContact = "apache@10.100.3.67";
$wgPasswordSender = "apache@10.100.3.67"; $wgEnotifUserTalk = false; # UPO $wgEnotifWatchlist = false; # UPO
$wgEmailAuthentication = true;
## Database settings
$wgDBtype = "mysql"; $wgDBserver = "localhost"; $wgDBname = "my_wiki"; $wgDBuser = "mjean"; $wgDBpassword =
"Passw0rd";
# MySQL specific settings
$wgDBprefix = "";
# MySQL table options to use during installation or update
$wgDBTableOptions = "ENGINE=InnoDB, DEFAULT CHARSET=binary";
## Shared memory settings
$wgMainCacheType = CACHE_NONE; $wgMemCachedServers = [];
## To enable image uploads, make sure the 'images' directory is writable, then set this to true:
$wgEnableUploads = false;
#$wgUseImageMagick = true; $wgImageMagickConvertCommand = "/usr/bin/convert";
# InstantCommons allows wiki to use images from https://commons.wikimedia.org
$wgUseInstantCommons = false;
# Periodically send a pingback to https://www.mediawiki.org/ with basic data about this MediaWiki instance. The
# Wikimedia Foundation shares this data with MediaWiki developers to help guide future development efforts.
$wgPingback = true;
## If you use ImageMagick (or any other shell command) on a Linux server, this will need to be set to the name
## of an available UTF-8 locale
$wgShellLocale = "C.UTF-8";
## Set $wgCacheDirectory to a writable directory on the web server to make your wiki go slightly faster. The
## directory should not be publicly accessible from the web.
#$wgCacheDirectory = "$IP/cache";
# Site language code, should be one of the list in ./languages/data/Names.php
$wgLanguageCode = "en"; $wgSecretKey = "ed8033d1030772b90033c2171d3530523550c34c29adcc5e475da07c64503a12";
# Changing this will log out all existing sessions.
$wgAuthenticationTokenVersion = "1";
# Site upgrade key. Must be set to a string (default provided) to turn on the web installer while
# LocalSettings.php is in place
$wgUpgradeKey = "fd50adaa54e92604";
## For attaching licensing metadata to pages, and displaying an appropriate copyright notice / icon. GNU Free
## Documentation License and Creative Commons licenses are supported so far.
$wgRightsPage = ""; # Set to the title of a wiki page that describes your license/copyright $wgRightsUrl = "";
$wgRightsText = ""; $wgRightsIcon = "";
# Path to the GNU diff3 utility. Used for conflict resolution.
$wgDiff3 = "/usr/bin/diff3";
## Default skin: you can change the default skin. Use the internal symbolic names, ie 'vector', 'monobook':
$wgDefaultSkin = "vector";
# Enabled skins. The following skins were automatically enabled:
wfLoadSkin( 'MonoBook' ); wfLoadSkin( 'Timeless' ); wfLoadSkin( 'Vector' );
# End of automatically generated settings. Add more configuration options below.
"LocalSettings.php" 71L, 4227C written
```

That's it! You've just installed a functional MediaWiki site on your very own LAMP stack.

Configuring a Samba Container

Step 1 – Now it's time for Samba. Enter your container using the same command as before, but use the container name you set for Samba instead:
`sudo lxc exec containername -- bash`

Step 2 – Use the following command to install the Samba package:

```
root@w0414274-samba:~# sudo apt install samba
```

To verify the integrity of the install issue the following command to ensure the service is running:

```
root@w0414274-samba:~# sudo systemctl status nmbd
● nmbd.service - Samba NMB Daemon
   Loaded: loaded (/lib/systemd/system/nmbd.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2019-11-18 21:27:03 UTC; 2min 44s ago
     Docs: man:nmbd(8)
           man:samba(7)
           man:smb.conf(5)
 Main PID: 2191 (nmbd)
   Status: "nmbd: ready to serve connections..."
    Tasks: 1 (limit: 4915)
   CGroup: /system.slice/nmbd.service
           └─2191 /usr/sbin/nmbd --foreground --no-process-group
```

Step 3 – Next we'll allow Samba through our firewall using the command below:

```
root@w0414274-samba:~# sudo ufw allow 'Samba'
Rules updated
Rules updated (v6)
```

Step 4 – Samba needs its own directory, user/group, and password; for the sake of brevity, just set up an account and group exactly as I've done in my example, swapping 'mjean' for a username of your own:

```
root@w0414274-samba:~# sudo mkdir /samba
root@w0414274-samba:~# sudo chgrp sambashare /samba
root@w0414274-samba:~# sudo useradd -M -d /samba/mjean -s /usr/sbin/nologin -G sambashare mjean

root@w0414274-samba:~#
root@w0414274-samba:~# sudo mkdir /samba/mjean
root@w0414274-samba:~# sudo chown mjean:sambashare /samba/mjean
root@w0414274-samba:~# sudo chmod 2770 /samba/mjean
root@w0414274-samba:~# sudo smbpasswd -a mjean
New SMB password:
Retype new SMB password:
Added user mjean.
root@w0414274-samba:~# sudo smbpasswd -e mjean
Enabled user mjean.
```

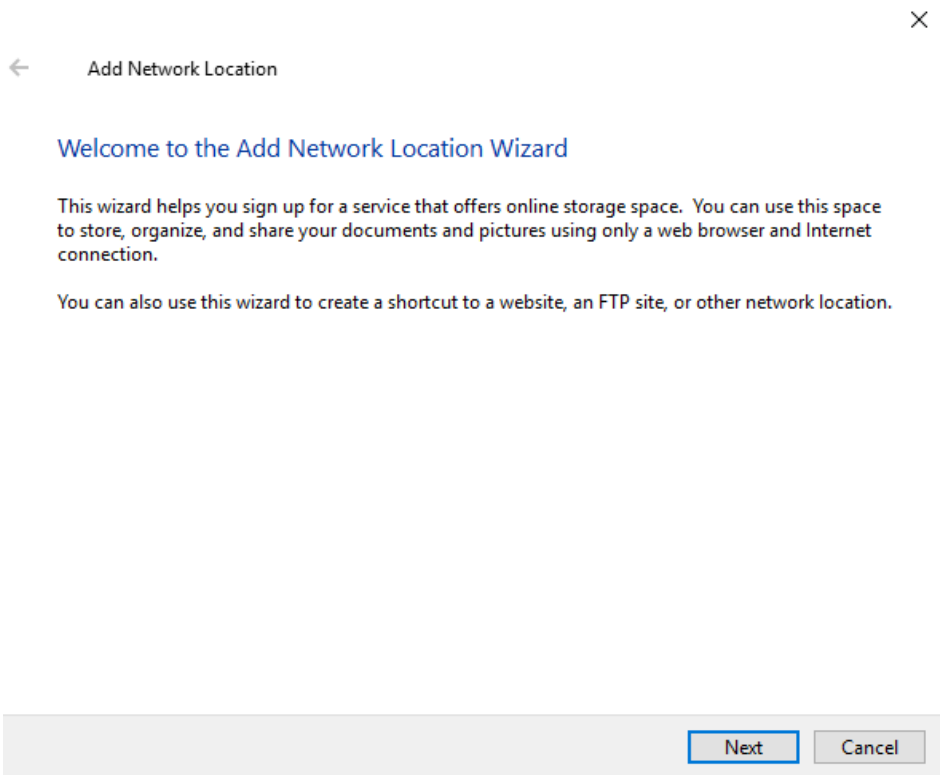
What's going on here is we're creating a user in the container (`-s /usr/sbin/nologin` simply doesn't allow the user to login, since all it's doing is validating access to our share; `-G sambashare mjean` adds the user to the sambashare group), giving it ownership/exclusive rights for /samba (after we created it using mkdir), the setting the aforementioned password.

Step 5 – After this, edit your /etc/samba/smb.conf file, declaring your share as follows:

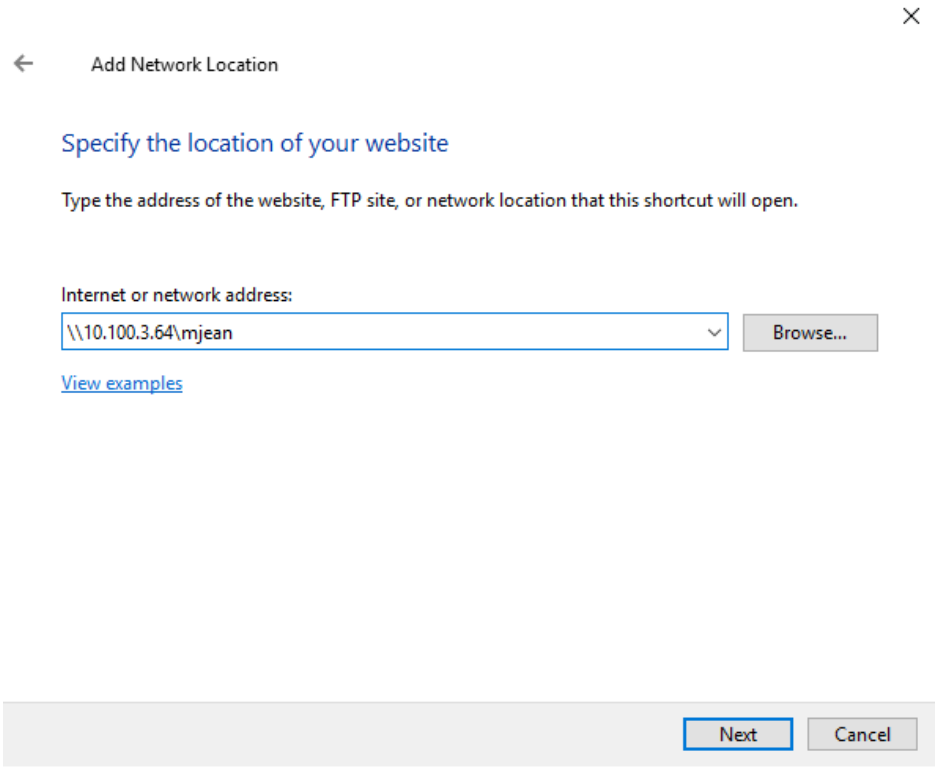
```
[mjean]
    path = /samba/mjean
    browseable = yes
    read only = no
    force create mode = 0660
    force directory mode = 2770
    valid users = mjean
"smb.conf" 274L, 9878C
```

In my example, [mjean] will be the name of my share when I try to access it over the network (we'll get there), your 'path' should reflect the directory we added in the previous step, and 'valid users' should include the user we created.

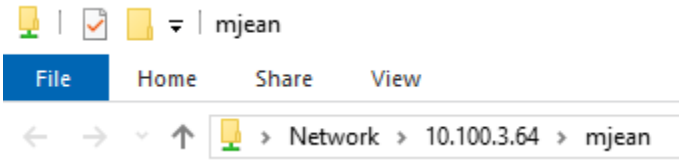
Step 6 – Now let's test our share. In my case, I'll be testing using a Windows system. To connect, navigate to your File Explorer, click 'This PC', then right-click on any white space and select 'Add a network location'. You should be greeted by this screen:



Click next, and type the address of your Samba container, followed by the name of your share, as seen below:



If everything is configured correctly, you should be able to see your share and any files within it inside your File Explorer:



This concludes the knowledgebase article on setting up a container-run Samba share!

References

Drake, M. (2018, April 27) How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 18.04. Retrieved from https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-ubuntu-18-04

Ubuntu Tutorials (n.d.) Create a bootable USB stick on Windows. Retrieved from https://tutorials.ubuntu.com/tutorial/tutorial-create-a-usb-stick-on-windows#0

PuTTY (2019, May 13 – last modified) PuTTY FAQ. Retrieved from https://www.chiark.greenend.org.uk/~sgtatham/putty/faq.html#faq-what

Zefanja, S. (2019, February 8) How To Set Up a Network Bridge for LXD Containers. Retrieved from https://openschoolsolutions.org/set-up-network-bridge-lxd/