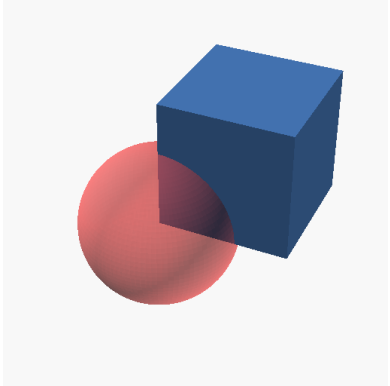

Pi-Zero-Cluster

Table of Contents

1. Document information

Links to Document

Document online

Document Source

Document PDF

2. Overview

Docker is being used to create a cluster that can run a number of applications that are independent of one another.

Figure 1. Picture of the cluster

An example is running an MQTT server (mosquitto) in one container, and another container running a manager to coordinate solar energy and car charging.
The differerent containers can have different dependencies, such as different versions of python, and they do not interact with each other.

The containers only communicate over the network.
The cluster spreads the network across the cluster so that containers do not need to run on the same hardware and can be spread across the cluster.

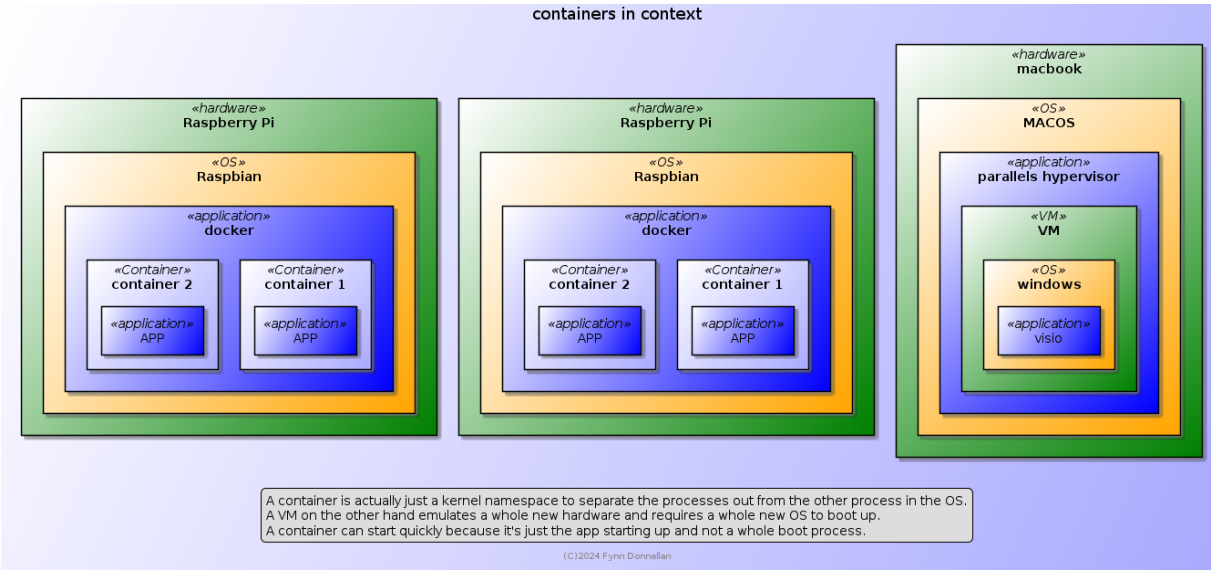

Figure 2. difference between containers and VMs

The reason not to use VMs is that they require too many resources.
Using containers reduces the amount of required resources so that even a small Pi Zero can be used for many applications.

Using Containers as opposed to VMs:

- reduces required memory
- reduces CPU overhead
  - individual containers do not need a full blown OS each
- reduces disk storage
- requires the apps to be able to run on the docker systems OS
  - Can’t run a windows or MACOS app on a linux Docker

2.1. Context (done from a Mac but can be done from Windows too)

The following steps assume your using a UNIX system like a MAC or another Linux system.
If you’re doing this from Windows you might want to try doing some of the steps that involve SSH and git from the Windows Subsystem for Linux.

- https://learn.microsoft.com/en-us/windows/wsl/install

3. Docker

As described above we're using docker here.
This section explains most of the steps to get the base system up and running.

Introductory Video to installing docker on RPi

- https://www.youtube.com/watch?v=qSpfWP-Fgjc

The above is where I got the information on how to do most of the things.

RPi software download

- https://www.raspberrypi.com/software/

This where I installed the Raspberrypi Imager. With this application, you can image a SD-Card to have the proper OS for your raspberrypi installed before you put it in. This will obviously come to hand when we are going to be setting up the cluster's software.

Whilst I was doing this, the diagram was created to show how the cluster is to be connected.

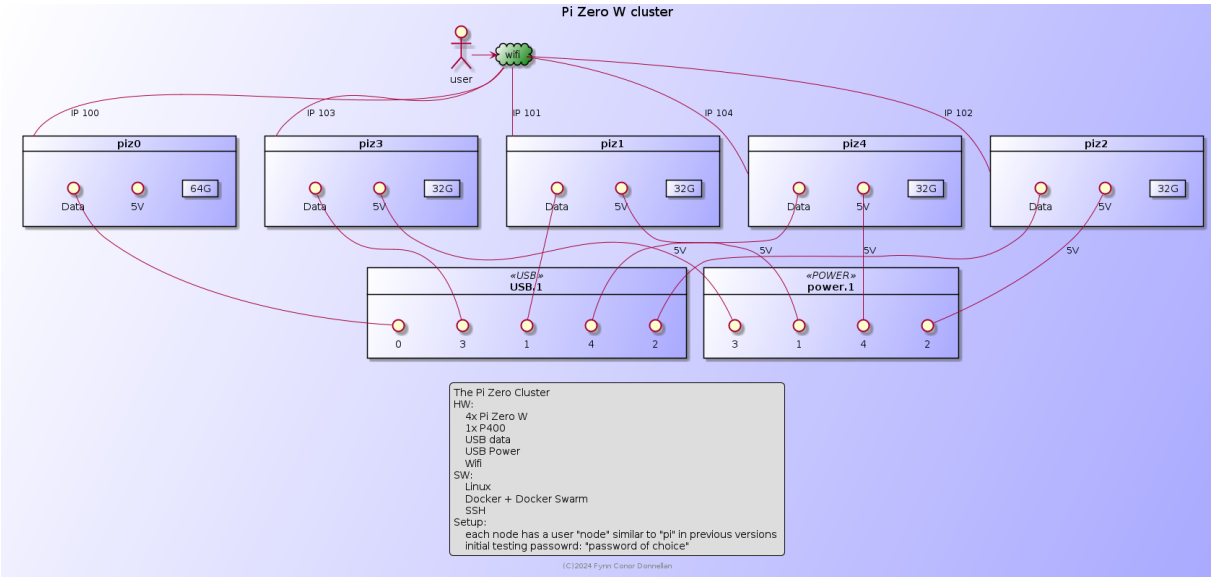

Figure 3. overview of setup

Further tasks after installing the OS

- Assign static IPs to all of the individual piZs
- edit the host file and include important network information
- create a number of ssh keys and connections to all of the piZs.

We then download docker and set it up with the help of this video.

- https://www.youtube.com/watch?v=-jLdkDxCDMo

A few steps might have to be done differently, especially when one gets to the point to initialize the swarm. There we had to log int our own router and assign the static IPs for all of the nodes (piZ1, piZ2, piZ3, piZ4). When doing so, it is important to specify the leasetime to infinite to prevent the IPs from changing later. I would recommend always looking at the settings from the

command "ip a" before changing the IPs and MACs on your router.

4. Gadget ethernet and serial

To make the piZeros into functioning ethernet gadgets so that they are connected by IP over USB and not just wifi the following has to be done.

Adafruit example

- https://learn.adafruit.com/turning-your-raspberry-pi-zero-into-a-usb-gadget/ethernet-tweaks

Image one of the disks to the full extent that is described in this tutorial and then copy all of its contents into a file to image the other 3 nodes. Instead of using the pi itself as the gateway, we used a commander node and it's IP as the router/gateway.

We also did not actually use a serial conection to do any of this, instead using an already established ssh connection, although we did run into some errors along the way and we did have to completely reconfigure the IP etc. of the first node we were working on to ensure the ssh connecton, because it broke a few times. Otherwise this tutorial worked just about fine.

Maybe next time: Pi cluster hat

- https://shop.pimoroni.com/products/cluster-hat?variant=22064645063
    - It may just relieve you of a lot of work and stress with having to use individual power supplies and USB hubs.

5. Regenerating ssh keys

Listing 1. regenerate keys

```
sudo dpkg-reconfigure openssh-server
```

Because we had cloned EVERYTHING from one disk to another, we had also cloned the ssh keys. This meant, logging into any of the nodes would result in logging into the first node. This problem is easily resolved by properly regenerating the ssh keys on every cluster one after the other, by logging into the original first, and regenerating the ssh key for it. Then, you'll want to log into the next node, still using the old ssh key and repeat the process until you are done with every node. You should then be able to log into every node individually again.

6. New Pis for the cluster and re-imagery

We got completely new cluster nodes after nearly a year of not working on this anymore. The new pis have to be re-imaged and set up completely by scratch. Maybe the stuff that we used before will still work, but there is no guarantee. I will also start adding files for 3D-printing all the

stuff that one needs for the cluster if one isn't into buying it. By the way, the new pis are raspberry pi zero 2W. The project started by tinkeriung with 32bit Pi zeros and has moved to the new Pi zero 2 and 64bit.

7. New images for the pis

We decided to image them with the Raspberry OS (Legacy-64bit) Lite.
You will want to get the Raspberry imager for this obviously.

- Allow the pis to access the WiFi and give them SSID and password, as this will make the process way easier.
- Then give them hostnames, e.g. "piZc1/2/3/4".
- In our case, we specify the rest to be located in Germany so that the wifi doesn't break laws and the language is set up
- Set a username and a password for the pi.
  - The username can be the same as the hostname, that shouldn't be a problem.
  - You will need this to enable SSH.

Be carefull when setting keyboard etc as if you don't watch out you'll have the wrong setup and wil find it hard to find the right keys.
Also the pi zero 2W can only do 2.4GHz on wireless so don't try adding it to anything but a 2.4GHz wifi as that will fail.

- When enabling ssh, you will want to make it only accessible with your public key, more of which can be added after you're done configuring.

You can add yours by going to your local machine, from which you will be accessing the cluster, and doing the following:

Listing 2. Get your local ssh key to add to the image

```
cd
cd .ssh
cat id_rsa.pub
```

And then copying that into the respective field in the imager.

8. Basics of ssh

To ssh into your newly setup pis, you will want to do the following:

Listing 3. SSH to a Pi

```
ssh hostnameOfYourPi@usernameOfYourPi
```

If your key has been copied across you will not be asked for a password as unlocking your local key secures access.
It's a very good idea to have a password for your local ssh key when you set it up for that reason.

9. Go to the `cluster-setup` directory of the repository

For the next steps you will want to be in the `cluster-setup` directory of the repository that contains the scripts that have been created to simplify some of the steps.

Listing 4. Clone the repo to your local machine if not already done

```
git clone https://github.com/DonFynn/pi-Zero-Cluster.git
```

Listing 5. Change to the cluster setup directory

```
cd pi-Zero-Cluster
cd cluster-setup
```

10. Update nodes OS

The following shell script gets copied to the systems and can be executed from your local system with the short script after it:

- Updates the package lists
- upgrades the systems software if new versions of installed software is available

Listing 6. Script that gets copied `cluster-setup/initial-setup.sh`

```
#!/usr/bin/env bash

#Upgrgade and update system
sudo apt-get update && sudo apt-get upgrade -y

#Update firmware - not really required but left just in case
#sudo rpi-update

#reboot
sudo reboot
```

Listing 7. short script to update all 4 systems

```
for system in piZc1 piZc2 piZc3 piZc4
    do
    scp ./initial-setup.sh ${system}@${system}:/tmp/
    nohup ssh ${system}@${system} bash /tmp/initial-setup.sh &
done
```

11. raspi-config settings per script

Normally you are asked to set the rasp-config settings for hostname etc interactively with the `rasp-config` tool.

It's also possible to do it via script.

Listing 8. setting raspi-config settings by CLI (non-interactively)

```
# set a new hostname (change it to the one you want)
sudo raspi-config nonint do_hostname NEW_HOSTNAME
# set the wifi country
sudo raspi-config nonint do_wifi_country DE
# set the locale
sudo raspi-config nonint do_change_locale en_US.UTF-8
# reduce amount of GPU mem to minimum
sudo raspi-config nonint do_memory_split 16
# set SSH on
sudo raspi-config nonint do_ssh 0
# set boot to CLI (no autologin)
sudo raspi-config nonint do_boot_behaviour B1
```

Some more of the options can be found in the source code here:

- https://github.com/raspberrypi-ui/rc_gui/blob/master/src/rc_gui.c

This is useful for when you want a cluster to have all the same settings and you don't want to spend a long time navigating menus on each device only to forget a setting anyway.

12. Install docker

The following shell script gets copied to the systems and can be executed from your local system with the short script after it:

- pulls a docker install script from the web
- executes the script and installs docker
- adds the user to the docker group to allow docker to be used without sudo

Listing 9. Script that gets copied `cluster-setup/initial-setup.sh`

```
#!/usr/bin/env bash

# Script to set up a cluster node with the basics

# install docker
sudo curl -sL get.docker.com|bash

# allow docker to be ran without sudo command
sudo usermod -a -G docker $USER

sudo reboot
```

Listing 10. Short script to install docker on all 4 systems

```
for system in piZc1 piZc2 piZc3 piZc4
    do
    scp ./initial-docker.sh ${system}@${system}:/tmp/
    ssh ${system}@${system} bash /tmp/initial-docker.sh
done
```

12.1. Test docker is working

The following script checks that docker is running by checking the docker version:

Listing 11. Short script to get docker version on all 4 systems

```
for system in piZc1 piZc2 piZc3 piZc4
    do
    ssh ${system}@${system} docker --version
done
```

Appendix A: 3D files

The files here are created in openSCAD which is a parametric 3D CAD program for creating solid printable objects.

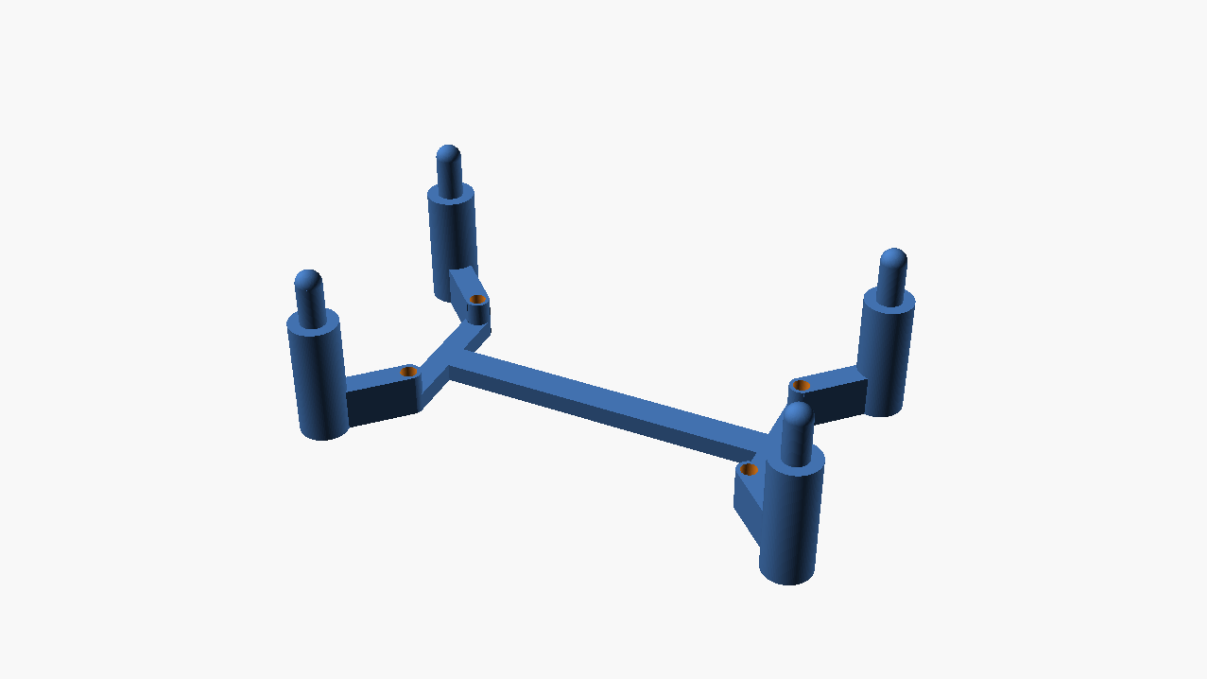

Figure 4. Bracket for the Pi Zero

The above bracket was downloaded here:

- https://www.thingiverse.com/thing:2502615

Figure 5. Base for the power supply and the USB hub

The above was created specifically for this cluster to hold:

- The 4 Pi Zero W2
- The power supply

- The USB hub

Figure 6. Putting it together

The source files, and the STL files, are in the `images` directory in the sub folder `3D`.

If you want to modify the design you will need:

- https://openscad.org/

Then open up the `.scad` files.

If you just want to print the designs just open the `.STL` files with a slicer. The slicewr creates gcode files for specific printers.

Slicer that was used to print the above

- https://ultimaker.com/software/ultimaker-cura/

The brackets and the cluster holder where printed on an Ender 3 printer in PLA.