Installation and use of Panama frog experiment Raspberry Pis

Mario Santos Mira

06/01/2023

Installation

OS installation

For these experiments we will be using the legacy version of Raspberry Pi. The legacy version has support for the picamera python module which will make things easier with my knowledge.

Using the Raspberry Pi Imager, we can select an alternative version of the Raspberry Pi OS. Here we want to select the "Raspberry Pi OS (legacy)" (or the Lite version). Before starting the installation, we need to change some options:

1) Enable SSH, so we can connect to the Pis without an extra monitor and keyboard;
2) Set username and password. Here I assigned each of the Pis a number and standardized both according to number. Usernames are panamaX, and passwords are panamaXpi, where X is the assigned number;
3) Configure wireless LAN. This might be necessary to update and install things on the Pis if there is no available Ethernet internet connection. Just write down the name and password for the connection. Make sure to also select the correct country for Wi-Fi. "eduroam" might not work.
4) Set the time zone.

After all the options are set, select the storage location where the OS will be installed making sure it is the SD card and not some other drive.

Setup

Once the installation is done, we can plug the Pi to power and wait for it to boot. If the Pi connected to a network (wireless or wired) it is possible to find its local IP address. This can be done by accessing the router and seeing which new IP address appeared, or you can use a Linux program with the following command:

```
sudo arp-scan -l
```

This will show all devices connected to the same network as the device running the command. With the local IP address, we can connect to the Pis using an SSH connection. In Linux this is already included in most installations, so you can just run:

```
ssh panamaX@xxx.xxx.xxx.xxx
```

Where X is the number assigned to the specific Pi, and the xxx.xxx.xxx.xxx is whatever IP address it has. This will ask for the password (and a confirmation if you actually trust the Pi when you first connect) and it will log you into the Pi.

Specific for the storage Pi The storage Pi, named panama1, will be assigning IP addresses in our offline local network, so we need to install an extra package:

```
sudo apt install dnsmasq
```

Once installed, we can configure the DHCP server to assign specific IPs. We do this by modifying a specific file:

```
sudo nano /etc/dnsmasq.conf
```

And adding the following at the end of the file:

```
interface=eth0
bind-dynamic
domain-needed
bogus-priv
dhcp-range=192.168.178.150,192.168.178.199,255.255.255.0,12h
```

As specified, the Pi will only work as a router (assigning IPs) when connected with Ethernet (the eth0 part), and it will assign IPs between 192.168.178.150 and 192.168.178.199.

Once added, we restart the server, so the changes can be applied:

```
sudo service dnsmasq restart
```

I followed this page to install this.

All Pis

Static IP address First we will assign static IP addresses to each Pi when using Ethernet. This will make it so that connecting to any of them will be much simpler, and we won't have to always look up their IP addresses. The information can be found here. It just involves modifying one file in each Pi:

```
sudo nano /etc/dhcpcd.conf
```

By adding the following at the end of the file or uncommenting and modifying the already existing section:

```
interface eth0
static ip_address=192.168.178.15X/24
static routers=192.168.178.150
static domain_name_servers=192.168.178.150
```

The X is the number of the Pi minus 1 (so panama1 is 150, panama2 is 151, and panama3 is 152). The router and domain name parts should point to the storage Pi. This Ethernet IP should be setup after a reboot.

Updating and installing packages To update the Pis we need to run two commands. First, we update the package manager:

```
sudo apt update
```

Next we upgrade whatever packages we already have installed:

```
sudo apt upgrade
```

This will ask for a confirmation. You can write "y" and press Enter.

Once updates are finished we can install the last necessary bit. The first one is `screen`. This package allows you to store you terminal instance and disconnect from the Pis without interrupting whatever you were doing. We install this with"

```
sudo apt install screen
```

Confirm all prompts and it will be installed.

Next we need to install a Python package to transfer files securely between the Pis. From here we see the package needs the Rust compiler. So we will install the Rust compiler first with:

```
curl https://sh.rustup.rs -sSf | sh
```

This will ask if you want the default installation. You can type "1" to continue with defaults. You also might need to restart the terminal to have the Rust compiler available. If you are connected via SSH, you will have t disconnect and connect again. Now that Rust is installed, we can install `paramiko` by using:

```
pip install paramiko
```

It should just install without any problems.

Disabling Wi-Fi We can disable the Wi-Fi access in the Pis since, if everything goes right, we won't need it any more, and it is very likely it won't connect to eduroam. To do this we run this command on all Pis:

```
rfkill block wifi
```

To undo this step, we run:

```
rfkill unblock wifi
```

To connect again to a wireless network, we can use the `raspi-config` options to again type the name and password of a Wi-Fi network.

Specific for camera Pis

Enabling the camera The camera Pis also need to have their camera modules enabled. To do this we go into the `raspi-config` options:

```
sudo raspi-config
```

Go into "Interface options" > "Camera" and enable it. It will prompt you to reboot the Pi. Once booted the camera should be operational.

SSH without password With this step we won't need to use passwords each time we want to log into a Pi. The full instructions are found here. We start by generating SSH keys in our laptops:

```
ssh-keygen -t rsa
```

There's no need to choose a passphrase or anything else. Just use all the default options and leave the passphrase empty. Next, go into the ssh directory and confirm the keys were created:

```
cd ~/.ssh/
```

And

```
ls
```

You should see at least `id_rsa` and `id_rsa.pub`. The pub file will be stored in the target Pis so that it "knows" your laptop and allows you in without password. Next we copy this file to the target Pi:

```
ssh-copy-id -i id_rsa.pub panamaX@192.168.178.xxx
```

Once again, the Xs correspond to whatever Pi you are targetting. This should be repeated for each Pi. Once this is done, you can connect with:

```
ssh panamaX@192.168.178.xxx
```

And it should just connect you.

Use

To use all the Pis you will have to connect via SSH. I recommend using a Linux distribution or installing a Linux distribution on your Windows laptop, as seen here. This guide (up to step 4) will allow you to use a Ubuntu terminal as if you had the whole Ubuntu OS.

Connecting to the Pis

At any point, you can connect to any Pi if you know it's username, IP address and password:

```
ssh panamaX@192.168.178.xxx
```

However, we can make things simpler in two ways. First, we can remove the need for a password (as done in the last step of the Installation section). And second, we can remove the need to type all that by creating aliases.

SSH without password (again) See the previous section with the same name. Using the Ubuntu Windows subsystem should be the same process. I haven't tried other programs.

Aliases To create aliases (that is you can create a simpler word that will do the same command) we need to modify the `.bashrc` file.

```
nano ~/.bashrc
```

In this file, at the end, you can add whichever aliases you want. For example, my aliases for the Pis are:

```
alias sshpa1='ssh panama1@192.168.178.150'
alias sshpa2='ssh panama2@192.168.178.151'
alias sshpa3='ssh panama3@192.168.178.152'
```

And with passwordless ssh once I type `sshpa1` and hit Enter, I'm automatically logged into the storage Pi (panama1).