SFS | Clean install and running setup

PowerMonitoring 2020

January 16, 2020

Contents

1 Logger

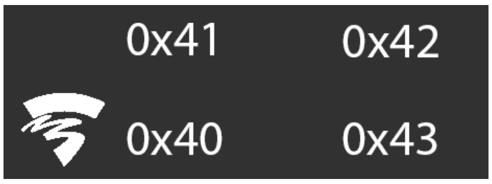
The logger is the part that records all the data from the INA260's, and from the clients, to a log directory. This is the main part of our project.

1.1 Setup

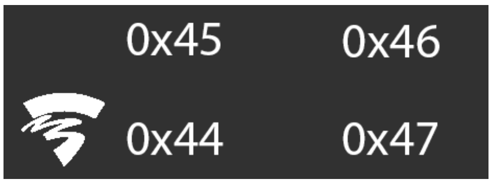
To continue where this project has left of, internet is definitely required. We use a hotspot to do everything that required an internet connection.
The cluster also needs to be setup correctly. Make sure all the power connectors from the black PCB's are connected to the correct Raspberry Pi's. We labelled the Pi's from bottom to top: 0 - 3, with the highest up one being the "g" pi, or Grafana pi. In our setup we conected them to the PCB as follows:

Address	Pi
0x40	0
0x41	1
0x42	2
0x43	3
0x44	G

With the addresses being on the following places:



(a) Bottom PCB



(b) Top PCB

Figure 1: PCB INA connector layout, following page 18 of `http://www.ti.com/lit/ds/symlink/ina260.pdf`

After that, make sure every Pi has it's own ethernet connection to the switch. If everything is setup correctly, it should work with the provided settings file "pis.xml" in the ina260 folder, the file will be required when starting the logger.

NOTE:

- In the current release of Debian Buster, the Pi picks only one of two interfaces (WiFi or Ethernet). Keep this in mind when trying to connect the Pi's to your hotspot.

Now you can make sure the Pi is authorized to access the HvA gitlab servers with an account.
The last step for setup, also the most essential for running, is to add all the Raspberry Pi's that you want to measure to your `/etc/hosts` file. For example, to add `raspberrypi-0`, you would write

```
0.0.0.0 raspberrypi-0.local
```

Where `0.0.0.0` has to be the actual IP address associated with the hostname. We have not been able to find out why this is necessary, but it seems that the logging programme is not able to do a reverse DNS lookup on it's own.

1.2 Installation

To install the logger, make sure you have the data logger pi setup in your ansible hosts file. In our case this file was located in `/etc/ansible/hosts` and contained the following lines:

```
[sgs]
raspberrypi-g.local ansible_user=pi
```

This means that you can run the playbook file `ansible/installG.yaml` with the following command: `ansible-playbook ansible/installG.yaml -l sgs`.

1.3 Running

To run the logger, you can now execute `powermonitoring/ina260/./ina260 -c pis.xml -d /home/pi/od`

2 Client

The client is required to record the temperature of the processing nodes, it does this by sending the data over the network to the logger.

2.1 Setup

The clients don't require much setup, other than an active internet connection, as mentioned in section 1.1. For ease of installation and use, it's also highly recommended to add the Ansible hosts group as mentioned in section 1.2. Only this time the hosts group should only contain the pi's that will run the test programs. In our case this made for a group that looked like this:

```
[sfs]
raspberrypi-0.local ansible_user=pi
raspberrypi-1.local ansible_user=pi
raspberrypi-2.local ansible_user=pi
raspberrypi-3.local ansible_user=pi
```

2.2 Installation

To install the client, we have to follow a similar path to the logger installation. Only this time we run the `ansible/installClient.yaml` playbook, like so: `ansible-playbook install.yaml -l sfs`. This should take care of everything.

2.3 Running

A prerequisite of running the client, is that the logger is already running on the Pi that does the data collection. If not, you **will** get a `connection refused` error. If it is running, you can start the clients by running `ansible -m command -a "powermonitoring/client/src/./echo` `20001 raspberrypi-g.local" sfs`. Here `raspberrypi-g.local` is the hostname of the Raspberry Pi running the logger, and `20001` is the port on which the logger listens.

3 FFTW and MPI

3.1 Library's

The test programs we use to simulate the load a astronomy supercomputer endures rely on a few library's. The FFTW[1] and the OpenMPI[2] library.

The FFTW library is used for the fourier transformations, it has integrated support for multi threading which is what one of the test programs uses. And it also has support for MPI which is what the other FFT test program uses.

The MPI library is used for the communication between the pi's, and requires some know how to properly install. But if you follow this install guide everything should work.

3.2 Setup

For the MPI programs to work it is important that all pi's can ssh to each other, since this is how they communicate. So they all need to be connected to the same network. In our setup two network switches are used to connect all the pi's to a router via ethernet cables, as shown on figure 2.

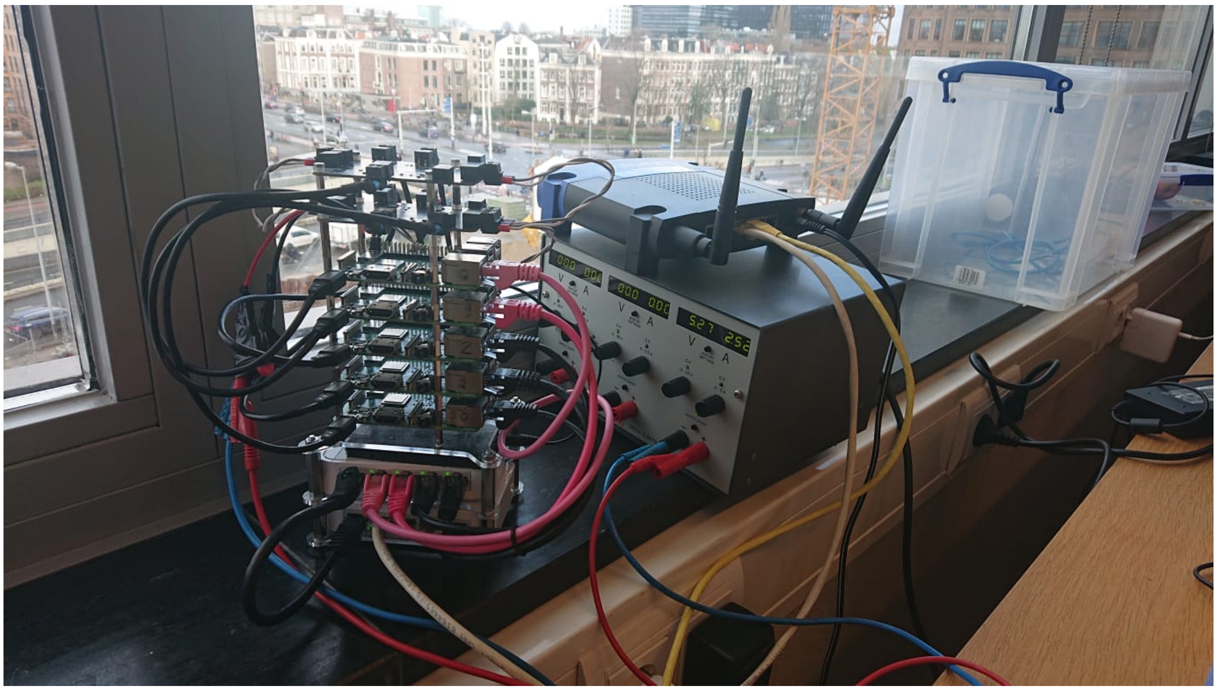


Figure 2: Raspberryi pi's connected to router

The process of giving each pi ssh access to each other has sadly not yet been automated and will have to be done manually. Each pi needs to have a ssh key and it's ID copy'd to all other pi's. To do this use the following commands:

```
ssh-keygen
```

followed up by,

```
ssh-copy-id pi@[ip adress of each pi].local
```

[1]Fastest Fourier In The West

[2]Message Passing Interface

I've made a table you can use to keep track of which pi has ssh access to which other pi.

	pi-0	pi-1	pi-2	pi-3
ssh-keygen				
ssh-copy-id to pi-0				
ssh-copy-id to pi-1				
ssh-copy-id to pi-2				
ssh-copy-id to pi-3				

The ip adress of each pi is also needed for the machinefile, this is a file that MPI uses to know which "machines" it has avalible. A machinefile looks like this, Listing 1

Listing 1: Machinefile for MPI

```
pi@192.168.3.11  node0
pi@192.168.3.16  node1
pi@192.168.3.5   node2
pi@192.168.3.37  node3
```

And is located in the directory of the executable that uses MPI. (it can be located elsewhere but in the directory itself is easiest in my opinion)

3.3 Installation

The install process has been automated using ansible scripts and is therefore very straightforward, we created a host group so the command can be sent to all pi's at once.

To install the library's, make sure you have all the pi's that you want to run the programs on in your ansible hosts file. In our case this file was located in `/etc/ansible/hosts` and contained the following lines:

```
[sfs]
raspberrypi-0.local ansible_user=pi
raspberrypi-1.local ansible_user=pi
raspberrypi-2.local ansible_user=pi
raspberrypi-3.local ansible_user=pi
```

This means that you can run the playbook file `ansible/ansible_install_fftw.yaml` with the following command: `ansible-playbook ansible/ansible_install_fftw.yaml -l sfs`.

After this you can run the playbook `ansible/ansible_install_Test_programs.yaml` with the following command:
`ansible-playbook ansible/ansible_install_Test_programs.yaml -l sfs`.