Raspberry Pi Configuration v2.0

Here is a list of Raspberry Pi images for LEaRN sensors:

- Pre-shrunken Raspbian Lite with version 0.5.1 of the LEaRN sensor firmware installed, **with Balena WiFi Connect Installed to support WiFi installations and click-through EULA** (for both large and small sensor boxes)
    - - learn-rpi-1_5_2-preshrunk.img
      - SHA256: 84c5d3f3749bec1ca86a0cc9a00f8b6822af7a9f3a8267afc8184a49e176056c
      - MD5: d3b01eb130946328294dbf78f85ad3cc
    - !!! Note that configuration YML files for 0.4.x are not compatible with 0.5.0+ (and vice versa) !!!

- Pre-shrunken Raspbian Lite with version 0.5.1 of the LEaRN sensor firmware installed (for both large and small sensor boxes)
    - learn-rpi-1_4_2-preshrunk.img
        - SHA256: 14c0ffc81d414abf4446b034ebe5b13d7ff2349212005f80c521a4a9d9c149ae
        - MD5: 75469d9a09ac769b2061befff4b8989a
    - !!! Note that configuration YML files for 0.4.x are not compatible with 0.5.0+ (and vice versa) !!!

- Pre-shrunken Raspbian Lite with version 0.4.4 of the LEaRN sensor firmware installed (for both large and small sensor boxes)
    - learn-rpi-1_3-preshrunk.img
        - SHA256: 5aa5eec8d5fe3612daf784b7a3e1f39ac0af4584e55f08a71d2aabe8aa70aef8
        - MD5: 59b66c4d693cee01a49ddd40b5aabeda

Burn image to SD card using `dd` on Linux (which will take MUCH less time than using a Windows machine):

```
sudo dd if=learn-rpi-1_4_2-preshrunk.img of=/dev/sdb bs=4M conv=sync status=progress
```

More details for burning the image to an SD card can be found here:

- https://www.raspberrypi.org/documentation/installation/installing-images/linux.md

Instructions for creating a shrunken Raspberry Pi image can be found here:

https://gist.github.com/selimnairb/ba819d936d0d3c9ab9d5a46adf887842

Setup Ethernet network connection with static IP (optional, will do DHCP by default)

Edit /etc/dhcpcd.conf

Configuration File

```
# Note: Under "nohook lookup-hostname" text

interface eth0

static ip_address=<ip>/16
static routers=<ip_gateway>
static domain_name_servers=<ip_dns>
```

Edit /etc/network/interfaces

Configuration File

```
# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

iface eth0 inet static
  address <ip address>
  netmask <netmask>
  gateway <gateway>
  dns-nameservers <dns>
```

Disable Bluetooth and WiFi

Edit /etc/modprobe.d/raspi-blacklist.conf

Configuration File

```
# WiFi
blacklist brcmfmac
blacklist brcmutil
# Bluetooth
blacklist btbcm
blacklist hci_uar
```

Setup WiFi (Optional)

Make sure WiFi network exists/can be seen by Raspberry Pi:

```
sudo iwlist wlan0 scan | grep MYSSID
```

Where MYSSID should be replaced with the SSID you wish to use

Add WiFi SSID and password to /etc/wpa_supplicant/wpa_supplicant.conf:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=US

network={
        ssid="MYSSID"
        psk="MYPASSWORD"
}
```

However, for a private network, you will have to do:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=US

network={
        ssid="MYSSID"
        scan_ssid=1
        psk="MYPASSWORD"
}
```

Disable Bluetooth

Edit /etc/modprobe.d/raspi-blacklist.conf

Configuration File

```
# Bluetooth
blacklist btbcm
blacklist hci_uar
```

Update OS

Update and upgrade your local repository in the Raspian Jessie Lite OS:

Commands

```
sudo apt-get update
sudo apt-get upgrade
```

If on a CGI network, add CGI CA cert to trust store:

- Copy `CGIFederalRootCA.crt` to `/usr/share/ca-certificates/CGIFederalRootCA.crt`
- `echo CGIFederalRootCA.crt >> /etc/ca-certificates.conf`
- `update-ca-certificates`

If when running tools like apt-get you see errors like:

perl: warning: Please check that your locale settings:
LANGUAGE = (unset),
LC_ALL = (unset),
LC_CTYPE = "UTF-8",
LANG = "en_GB.UTF-8"
are supported and installed on your system.

Do the following:

sudo -s
export LANGUAGE=en_US.UTF-8
export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8
locale-gen en_US.UTF-8
dpkg-reconfigure locales

Enable auto updates

First, install `unattended-upgrades`:

```
sudo apt-get install unattended-upgrades
```

Edit /etc/apt/apt.conf.d/50unattended-upgrades, making sure that `Unattended-Upgrade::Origins-Pattern` has the following enabled:

```
"origin=Debian,codename=${distro_codename},label=Debian-Security";
"origin=Debian,codename=${distro_codename},label=main";
"o=Raspbian,a=stable,n=stretch,l=Raspbian,c=main";
"o=Raspbian,a=stable,n=stretch,l=Raspbian,c=non-free";
"o=Raspberry Pi Foundation,a=stable,n=stretch,l=Raspberry Pi Foundation,c=ui";
"o=Raspberry Pi Foundation,a=stable,n=stretch,l=Raspberry Pi Foundation,c=main";
"o=Raspbian,a=stable,n=stretch,l=Raspbian,c=rpi";
```

The edit `/etc/apt/apt.conf.d/20auto-upgrades` to look like:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";
```

Install depdencies

```
sudo apt-get install python3
sudo apt-get install python3-rpi.gpio
sudo apt-get install python3-wheel
sudo apt-get install python3-pip
sudo apt-get install git
```

Install Python dependencies

Commands

```
sudo easy_install-3 -U pip

sudo wget https://pypi.python.org/packages/49/df/50aa1999ab9bde74656c2919d9c0c085fd2b3775fd3eca826012bef76d8c
/requests-2.18.4-py2.py3-none-any.whl#md5=eb9be71cc41fd73a51a7c9cd1adde5de

sudo wget https://pypi.python.org/packages/97/8a/d710f792d6f6ecc089c5e55b66e66c3f2f35516a1ede5a8f54c13350ffb0
/requests_toolbelt-0.8.0-py2.py3-none-any.whl#md5=aab9f93459410330d292753e398d51b3

sudo pip3 install requests-2.18.4-py2.py3-none-any.whl
sudo pip3 install requests_toolbelt-0.8.0-py2.py3-none-any.whl
```

Clone the LEaRN repository and install sensor code

Commands

```
git clone https://github.com/learnlafayette/sensors.git
cd sensors/sensors/
sudo python3 setup.py install
cd
sudo rm -rf sensors
```

Configure sensor sampler and transmitter as services

Setup Sensor and Transmit services:

Create **/etc/systemd/system/sensor.service** and add the following contents:

Configuration File

```
[Unit]
Description=Sensor sampler and spooling service
After=multi-user.target

[Service]
Type=simple
Environment="LEARN_YAML_PATH=/home/pi/sensor.yml"
ExecStart=/usr/local/bin/sensor_raspi_sample

[Install]
WantedBy=multi-user.target
```

Create **/etc/systemd/system/transmit.service** and add the following contents:

Configuration File

```
[Unit]
Description=Sensor data transmit service
After=network.target

[Service]
Type=idle
Environment="LEARN_YAML_PATH=/home/pi/sensor.yml"
ExecStart=/usr/local/bin/sensor_transmit

[Install]
WantedBy=network.target
```

Set permissions and configure systemd for the new services:

Commands

```
chmod 644 /etc/systemd/system/sensor.service
chmod 644 /etc/systemd/system/transmit.service
systemctl enable sensor.service
systemctl enable transmit.service
```

Setup sensor YAML file

Place sensor YAML file in **/home/pi/sensor.yml**. Make sure its permissions are correctly set:

```
chmod 644 /home/pi/sensor.yml
```

Contents of config should be something like:

```
# Example LEaRN sensor configuration example for a stationary sensor
# Note: Don't use tab characters in this file as it blows up the YAML parser.

logging:
  logger_path: /var/log/sensor.log
  level_file: WARNING
  level_console: WARNING
spooler:
  db_path: /var/spool/sensor.sqlite
thing:
  id: 3f271e1f-c779-48ee-9737-4dbd018f2af4
  location_id: 5069f182-6d6d-4ed0-9492-e1bbf1b2a2d1
sensors:
  - type: mq131
    observed_properties:
      - name: ozone
        datastream_id: be3601ca-4456-45e2-aa4d-d5db3918d989
        properties:
          - Ro: 2.8599
  - type: dht11
    observed_properties:
      - name: air_temperature
        datastream_id: 64bc0f97-b1c0-49a5-873a-0062fa0fb39c
      - name: relative_humidity
        datastream_id: c275936a-b755-463f-a315-12d1c978bb1f
transports:
  - type: https
    properties:
      auth_url: https://dev2-sta-api.learnlafayette.com/SensorThingsService/auth/login
      url: https://dev2-sta-api.learnlafayette.com/SensorThingsService/v1.0/
      jwt_id: 56f9dfee-69f9-4987-91b4-8bbdace1e2ae
      jwt_key: 30f4c588-57c6-4956-b809-4cc71aef221f
      jwt_token_ttl_minutes: 15
      transmit_interval_seconds: 15
      verify_ssl: true
```

To reduce wear on the fragile flash storage of the Raspberry Pi, it is recommended to set the logging level for file logging (**level_file**, which in the above is logged to **/var/log/sensor.log**) and for console logging (**level_console**, which will be captured by **/var/log/syslog**) to "WARNING". If you are debugging, you can set one of these to "DEBUG", but for production, these should be set to "WARNING" or "ERROR".

Calibrate MQ131 sensor

The **Ro** property of the **MQ131** sensor must be calibrated. To calibrate the sensor do the following:

- Turn on the sensor and let it run for 24-hours
- Run the command **sudo calibrate_mq131**
    - Run this command 3 times in quick succession and average the results (to four decimal places)
    - This value is **Ro**, which should be put in sensor.yml

Calibrating the **Ro** parameter of the MQ131 is recommended, but optional. If not specified, a default value of 2.501 will be used.

Configure peripherals

```
sudo nano /boot/config.txt
```

Make sure the following lines appear in config.txt:

```
dtparam=i2c_arm=on
dtparam=spi=on
enable_uart=1
```

Also disable audio by commenting out the line "dtparam=audio=on".

Setup Cron job to restart sampler and transmitter services (in case they crash):

Commands

```
 sudo crontab -e
```

Configuration File

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

# */1 * * * * sudo -u pi scp /var/log/sensor.log <Where log file gets copied to every minute>

*/13 * * * * sudo systemctl restart transmit.service
*/29 * * * * sudo systemctl restart sensor.service
0 3 * * * sudo shutdown -r +`shuf -i 0-50 -n 1`
```

We restart **transmit.service** more frequently, every 13 minutes, because we have seen the transmit service crash ocassionally during testing. We have fixed several causes of these crashes, but to be cautious, we automatically restart the transmit server frequently. This way we do not have a long gap between delivery of subsequent observation. Note that even if the transmit service does crash, observations will not be lost as the sensor sampling service is totally independent and will continue generating observations. However observation delivery will be delayed until the transmit service restarts.

We restart **sensor.service** less frequently, every 29 minutes, because we have not observed many/any crashes during testing. Still, we restart the sensor sampler from time to time to make sure we don't miss out on the opportunity to sample data for too long.

It is good practice to use different prime numbers for the above restart intervals. This way, there will never be a time where both services are trying to start at the same time. This will reduce the chances of overloading the Raspberry Pi and more importantly reduce the chances of race conditions arising over contention for shared resources (e.g. the spooler database used to store observations before transmission).

The last line reboots the Raspberry Pi once per day at 3:00am with a random offset of M minutes where m is between 0 and 50.

Check the status of the new services running after rebooting:

```
sudo systemctl status sensor.service
sudo systemctl status transmit.service
```

Check the log to see the application at work:

```
tail -f /var/log/sensor.log
```