DHT sensors reading with Raspberry Pi

Peixun Shen

2020/7/24

Introduction (RPi_DHTsensors)

- This material is used to show to read, store, display and transfer DHT sensor temperature and humidity data with RPi and host server.
- We use the Adafruit_Python_DHT package to read data, the influxdb and grafana to display data, the rsync + initify to transfer data from clients to host server.
- The corresponding codes can be found in: https://github.com/Hep-dog/RPi_DHTsensors
- Main softwares link:
  - Adafruit_Python_DHT: https://github.com/adafruit/Adafruit_Python_DHT
  - Influxdb and Grafana: https://www.terminalbytes.com/temperature-using-raspberry-pi-grafana/

The usage of this package has been introduced in the README, and this doc shows some main steps and results during the installation and running.

1. Installation (clients and server)

1. git clone https://github.com/Hep-dog/RPi_DHTsensors
2. Using the Installation/autoInstalling.sh to install:
   a. Set the workarea: $ echo “DHTsensors_workarea=‘….’ ” >> ~/.bashrc
   b. $ source ~/.bashrc
   c. $ cd $DHTsensors_workarea/Installation/autoInstallationg.sh
      (the sentences for Adafruit_DHT should be commented for Linux host server)

Packages needed:

RPi clients: Adafruit_DHT, Influxdb and Grafana (optional), postfix and inotify

Packages for host server: Influxdb and Grafana

Note: the python version should be python2.+ (less than python3)

```
pi@raspberrypi:~/Work $ git clone https://github.com/Hep-dog/RPi_DHTsensors.git
正克隆到 'RPi_DHTsensors'...
remote: Enumerating objects: 137, done.
remote: Counting objects: 100% (137/137), done.
remote: Compressing objects: 100% (86/86), done.
remote: Total 137 (delta 40), reused 131 (delta 34), pack-reused 0
接收对象中：100% (137/137), 133.15 KiB | 296.00 KiB/s, 完成.
处理 delta 中：100% (40/40), 完成.
pi@raspberrypi:~/Work $ cd RPi_DHTsensors/
pi@raspberrypi:~/Work/RPi_DHTsensors $ echo "export DHTsensors_workarea='/home/pi/Work/RPi_DHTsensors'" >> ~/.bashrc
pi@raspberrypi:~/Work/RPi_DHTsensors $ source ~/.bashrc
pi@raspberrypi:~/Work/RPi_DHTsensors $ echo $DHTsensors_workarea
/home/pi/Work/RPi_DHTsensors
pi@raspberrypi:~/Work/RPi_DHTsensors $
```

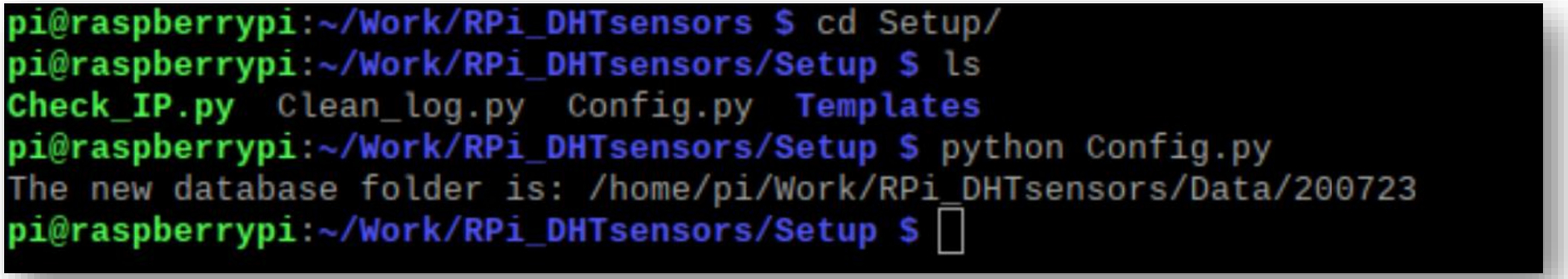
```
pi@raspberrypi:~/Work/RPi_DHTsensors $ ls
Adafruit_Python_DHT  Collection  Crontab  Display  Installation  Parameters  README.md  Setup  Sync
pi@raspberrypi:~/Work/RPi_DHTsensors $ cd Installation/
pi@raspberrypi:~/Work/RPi_DHTsensors/Installation $ ls
autoInstalling.sh
pi@raspberrypi:~/Work/RPi_DHTsensors/Installation $ vim autoInstalling.sh
pi@raspberrypi:~/Work/RPi_DHTsensors/Installation $ source autoInstalling.sh
获取:1 http://archive.raspberrypi.org/debian buster InRelease [32.6 kB]
获取:2 http://archive.raspberrypi.org/debian buster/main armhf Packages [330 kB]
命中:3 http://raspbian.raspberrypi.org/raspbian buster InRelease
已下载 363 kB，耗时 4秒 (95.9 kB/s)
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
有 75 个软件包可以升级。请执行 'apt list --upgradable' 来查看它们。
OK
deb https://repos.influxdata.com/debian buster stable
命中:1 http://raspbian.raspberrypi.org/raspbian buster InRelease
获取:2 https://repos.influxdata.com/debian buster InRelease [4,731 B]
获取:3 https://repos.influxdata.com/debian buster/stable armhf Packages [929 B]
```

Packages Setup (clients and server)

- The scripts in Setup are used to setup several template files, and copy them to each directory for further using.
- The workarea should be defined as environment variable, and then:

  $ python Config.py

- The Clean_log.py and Check_IP.py are useful scripts. The first one is used to clear the log file (typically in /var/log directory), and the latter is used to get the dynamic IP for Rpi client (When the telegraf method is used to transfer raw data to influxdb, the IP address should be set in the influxdb.conf).



Parameters for data collection and displaying

- The Parameters directory holds the default parameters for data collection (Paras_coll.py) and display (Paras_db.py). The sub-dir “Templetes” in Setup directory has the basic script templates with default parameters, which are activated after the workarea defining and Config.py running).
- Usually, the default values are generated automatically. User needn’t to change them.

```
a. For data collection, we use the class named "collectionModule" in Collection dir, to read and save the raw data.
There are several parameters should be set firstly, which is defined in the "Paras_coll.py" file in Parameters dir.

    sensor = sensor    // Name of sensor for AdafruitDHT, should be DHT11, DHT22 or AM2302
    sensor_gpios = [ ] // the gpio BCM number of the sensor
    meas = meas[ ]     // Name for measurement in influxdb
    outputs = [ ]      // Data files for saving the humi and temp values

b. For data transmission from raw data to influxdb database, and for displaying later. We use the class named "Display"
to transfer the raw data to infludb. Serveral parameters should be set firstly, which the default values are defined in
the "Para_db.py" file in Parameters dir.

    host = host  // The IP address for influxdb. The default value is ok, which can be obtained with the function "Check_IP"
    port = port  // Port for influxdb, default is ok
    user = user      // User for influxdb, could be null
    passwd = passwd  // Password for User in influxdb, could be null
    dbname = dbname  // Name of the database
    meas = meas[ ]   // Name for measurement in influxdb
    outputs = [ ]    // Data files for saving the humi and temp values

Notes: Multi sensors can be used simultanously. For this case the number of measurements , sensor_gpios and outputs need to keep same!
```

Data collection in RPi clients

- The directory “Collection” holds the script for RPi read and store the temperature and humidity data, the default path for raw data is the sub-dir “data” in “Collection”.
- User can collect data manually to test the program (in Collection dir):

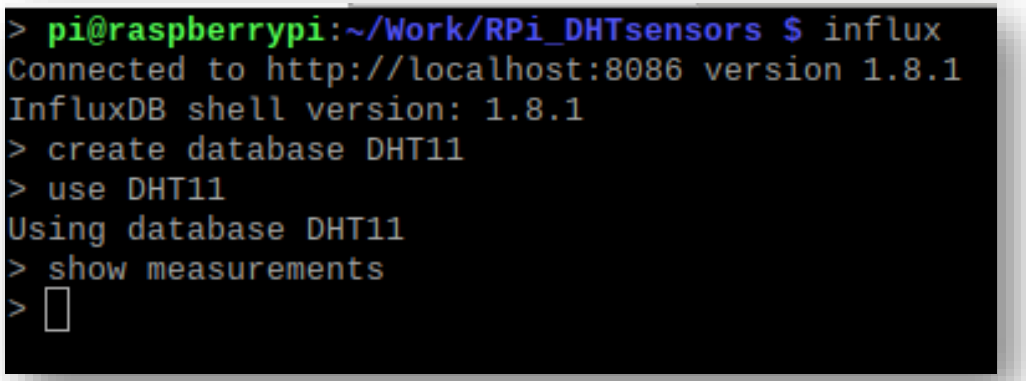
  $ python runCollection.py

  If the program is correctly running, the dat files in “data” should have temp and humi data now!

```
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection $ mkdir data
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection $ python runCollection.py
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection $ cd data/
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection/data $ ls
dht11_1.dat  dht11_2.dat  dht11_3.dat
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection/data $ cat dht11_1.dat
1595498019773541888 Temp=28.0C and Humidity=48.0%
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection/data $ date -d @1595498019
2020年 07月 23日 星期四 17:53:39 CST
pi@raspberrypi:~/Work/RPi_DHTsensors/Collection/data $
```

Data displaying I

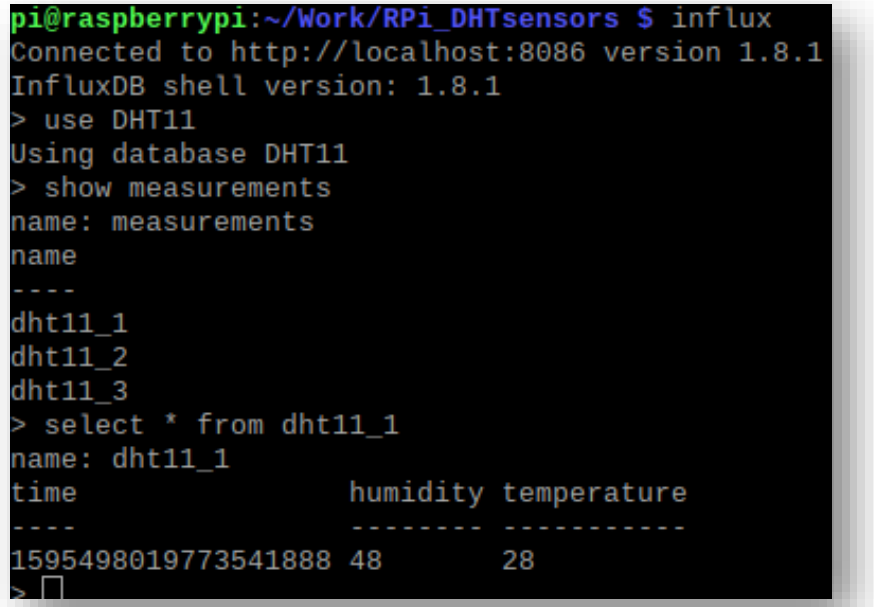
- When the data collection step is correct, we can display it with Influxdb and Grafana now.
- We should create a new database in the influxdb firstly (no data in database in this step).



- Then run the “rundb.py” in “Display”

  $ python rundb.py

The name of database should to be same as the one in Parameters/Paras_db.py! And you can see the database in Influxdb has data now:



Data displaying II

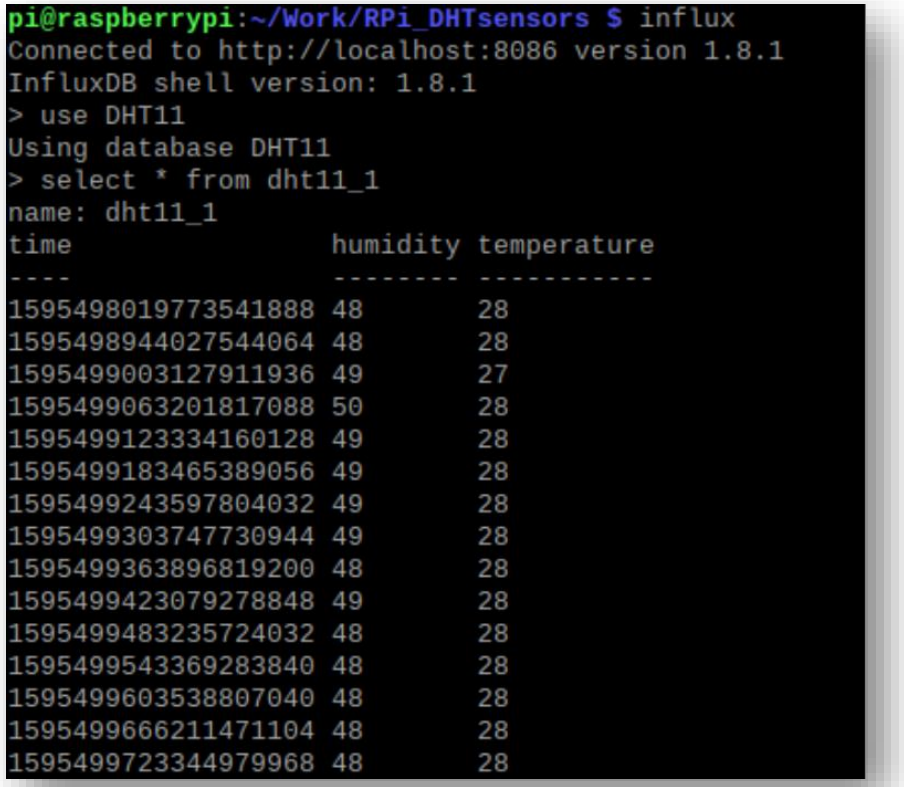
- If the tests of data collection and displaying steps are correct, we can do them in a automatically way by using crontab.
- The “Crontab” directory holds the script to run the data collection and displaying steps in crontab. We should add the schedule in crontab (* * * * * means run them every minute):

```
$ crontab -e
Adding "* * * * * /bin/bash workarea/Crontab/doCollection.sh";
Adding "* * * * * /bin/bash workarea/Crontab/doDisplay.sh";
```

- For linux host server, i.e. ubuntu in my laptop, there are some problems when using crontab to run the data collection and displaying. They are caused by the difference between crontab env and user shell env, and can be fixed by adding the following configures in the top of crontab:

```
$ SEHLL=/bin/bash
$ BASH_ENV=~/.bashrc
$ PATH=/sbin:/bin:/usr:/sbin:/usr/bin
$ ...
```

- After the setup of schedule in crontab, one should check its status.
- If the data collection and displaying processes working, there will be data stream in the Influxdb database now!



Data displaying III

Tutorial for using influxdb in Grafana:
https://www.cnblogs.com/laraine/p/11188770.html

- Once the Influxda has data stream, we can display the data with Grafana:

1. Set influxdb database in Grafana(localhost:3000)

2. Create and set dashboards

Data Synchronizing (clients to host)

- The raw data is stored and displayed in RPi clients, and we want to transfer the data to host server for safety.
- This can be done using the rsync + initify to transfer data in real time (The client and host server should be in the same local network link. If not, the frp method can be used, which is beyond this doc).
- The scripts in Sync directory are used for the data synchronizing. Users can follow the commands listed in the “readme” in Sync dir:

  https://github.com/Hep-dog/RPi_DHTsensors/blob/master/Sync/readme

Note: the IPs parameters for linux host server and RPi clients should be set!

trans.sh

```
#!/bin/bash
remote_ip=192.168.43.84
/usr/bin/inotifywait -mrq -e modify,delete,create,attrib,move /home/pi/Work/RPi_DHTsensors/Collection/d
do
    sudo rsync -av --password-file=/etc/rsyncd.secrets /home/pi/Work/RPi_DHTsensors/Collection/data/ pi
    sudo echo "`date +'%F %T'` have $events" >> /tmp/rsync.log 2>&1
done
```

Host IP address

Rsyncd.confg

```
uid = root
gid = root

[dht11]
path = /home/pi/Work/RPi_DHTsensors/Collection/data
ignore errors
read only = no
auth users = pi
secrets file = /etc/rsyncd.secrets
hosts allow = 192.168.43.80, 192.168.43.100, 192.168.43.190
```

Clients IP address

Raw Data backup (clients and host)

Since we use the rsync to synchronize data from RPi clients to Ubuntu server, the data in host will be lost if those in clients are lost! So we add the script in directory "Backup" to backup data day by day. In this way, the raw data could be keep the size for latest few days to save storage, because all data are backup in server. One should add the running of the backup script to crontab with per day schedule.

```
Adding the following command to crontab, just like the data collection and displaying:
        $ * * * * * /bin/bash workarea/Crontab/doBackup.sh
```