Installation on Debian-Based Systems

Qosium Probe can be installed by using Debian package management system. This works on Debian, Ubuntu, Raspbian (Raspberry Pi), and other Debian-based Linux distributions. Installation can be done in terminal or by using package manager applications like the Software Center and Synaptic.

Table of Contents

1. Preparations

First, sign in to your account and access your downloads page. Then download Qosium Probe for the target machine. The package is typically named as `QosiumProbe_<version_details>.deb`, where the `<version_details>` part depends on version to another and may contain many identifiers.

2. Package Pre-checks (Optional)

The Debian package can be verified by opening Terminal and running:

```
dpkg --info QosiumProbe_<version_details>.deb
```

To see what will be installed and where to, run:

```
dpkg --contents QosiumProbe_<version_details>.deb
```

3. Installation

To install a fresh or upgrade an existing Qosium Probe in a machine, open Terminal and run:

```
sudo dpkg -i  QosiumProbe_<version_details>.deb
```

It is also possible to use *apt* for installation. This method installs automatically all dependencies. Installation using apt-get happens the following way:

```
sudo apt-get install ./QosiumProbe_<version_details>.deb
```

The installation process asks all the relevant details, e.g., whether or not to install Qosium Probe as a Systemd service, etc.

After installation, you can check which version of Qosium Probe is installed by running:

```
dpkg -l qosiumprobe
```

4. Uninstallation

To remove Qosium Probe from the device, run:

```
sudo dpkg -r qosiumprobe
```

5. GNSS Setup

In Linux, GNSS positioning is based on gpsd and time sync ntpd. The following guide expects that you have a compatible GNSS receiver.

5.1. Prerequisites

First, you need a compatible *GNSS* unit. Any GNSS with PPS signal that works in Linux should apply. However, if you did not purchase the GNSS unit from Kaitotek, modifications to the configuration may be needed to match with the wiring of the GNSS unit.

The following packages need to be installed (Ubuntu):

- gpsd
- ntp
- setserial
- pps-tools (optional, recommended for debugging problems)
- gpsd-clients (optional, recommended for debugging problems)

You need NTP with PPS clock support. By default, it is provided with latest updates of Ubuntu xenial (16.04 LTS) and yakkety (16.10) and zesty (17.04). In Ubuntu, the PPS is enabled in NTP 4.2.8p4 or later. If your system has a version of NTP without PPS support, compile it yourself with PPS clock (22) enabled. This can be done by enabling all clocks during configuration phase: `./configure --enable-all-clocks`.

This guide assumes that kernel level PPS is supported. If you have a modern distribution this should not be a problem, but for example some old CentOS distributions do not have kernel level PPS support. In this case special *shmpps* application with NMEA(20) and SHM(28) NTP drivers and special configuration might help you to get application level PPS support.

In order to reach a good result, your machine should have an integrated serial port or at least a PCI-based serial port adapter for the PPS signal. If, e.g., a USB serial port adapter is in use, the synchronization accuracy will be much worse.

5.2. Step-by-Step Instructions

This chapter assumes that the serial port location is */dev/ttyS0* (an integrated serial port for PPS signal) and USB port is */dev/ttyUSB0* (for NMEA message reception). Port numbers may vary depending on your system, so please check and change them accordingly.

To install the required packages, type:

```
sudo apt-get install gpsd gpsd-clients ntp pps-tools setserial
```

Connect the USB and serial cables. It might be useful to check in which port the USB device is attached:

```
dmesg
```

To start *gpsd* type:

```
sudo service gpsd start
```

Then, check that GNSS reception is working:

```
xgps
```

You should see satellites appearing. It might take few minutes initially (as long as 12 minutes in the worst

case). Please wait until the fix is got. If nothing happens, exit the xgps and check that gpsd is using the correct USB port by typing:

```
sudo dpkg-reconfigure gpsd
sudo service gpsd restart
```

When the GNSS basic reception is working fine, enable the serial driver for PPS (18). In case there are several serial ports, please modify the port number */dev/ttyS<port>* to match the correct one.

```
sudo ldattach 18 /dev/ttyS0
```

/dev/pps0 device should appear. If there are several */dev/pps<port>* devices or you want to ensure that the PPS is working type:

```
sudo ppstest /dev/pps0
```

If you see output appearing every second, the PPS works. Press *Ctrl+C* to stop. If you don't see anything after `ok, found 1 source(s), now start fetching data...`, please check the device is properly connected and port number is correctly selected.

Enable serial port **low_latency** mode to ensure low jitter:

```
sudo setserial /dev/ttyS0 low_latency
```

Configure NTP by editing */etc/ntp.conf*.

Disable the default servers by commenting them out, if you want to use only GPS time. Example:

```
# Use servers from the NTP Pool Project. Approved by Ubuntu Technical Board
# on 2011-02-08 (LP: #104525). See http://www.pool.ntp.org/join.html for
# more information.
#server 0.ubuntu.pool.ntp.org
#server 1.ubuntu.pool.ntp.org
#server 2.ubuntu.pool.ntp.org
#server 3.ubuntu.pool.ntp.org
```

Add the following lines:

```
#PPS
server 127.127.22.0 minpoll 4 maxpoll 4
fudge 127.127.22.0 flag2 0
#SHM
server 127.127.28.0 prefer
fudge 127.127.28.0 time1 0.55 refid GPS
```

Again, if the */dev/pps<port>* device number is not 0, you should modify the number in PPS lines accordingly. Example (PPS device */dev/pps1*):

```
#PPS
server 127.127.22.1 minpoll 4 maxpoll 4
fudge 127.127.22.1 flag2 0
```

Start the NTP

```
sudo service ntp start
```

5.3. Checking NTP Status

Wait for the clock to be synchronized. To check the status type:

```
ntpq -p
```

Typical outputs are presented next.

1. The clock is correctly synchronized (see the marks o and *):

```
remote           refid           st t when poll reach   delay   offset  jitter
==============================================================================
oPPS(1)          .PPS.            0 l   11   16  377    0.000   -0.011   0.019
*SHM(0)          .GPS.            0 l   42   64  377    0.000    6.741  15.645
```

2. Not (yet) synchronized. Please wait or recheck the configuration. Also, if the clock is off too much, NTP could refuse synching. In this case, do initial synching manually.

```
remote           refid           st t when poll reach   delay   offset  jitter
==============================================================================
PPS(1)          .PPS.            0 l    -   16    0    0.000    0.000   0.000
SHM(0)          .GPS.            0 l    -   64    0    0.000    0.000   0.000
```

3. SHM only synchronized. Please wait for the PPS sync or recheck the PPS configuration.

```
remote           refid           st t when poll reach   delay   offset  jitter
==============================================================================
PPS(1)          .PPS.            0 l    -   16    0    0.000    0.000   0.000
*SHM(0)         .GPS.            0 l    3   64    1    0.000  -14.525   0.000
```

4. Clock sources dropped as falseticker due to poor accuracy. Please check that the GPS signal is strong enough and restart NTP.

```
remote           refid           st t when poll reach   delay   offset  jitter
==============================================================================
xPPS(1)          .PPS.            0 l    -   16    0    0.000    0.000   0.000
xSHM(0)          .GPS.            0 l    -   64    0    0.000    0.000   0.000
```

5. No PPS clock. Please check that PPS (22) is enabled in *ntp.conf* and that NTP has PPS driver support.

```
remote           refid           st t when poll reach   delay   offset  jitter
==============================================================================
*SHM(0)          .GPS.            0 l    3   64    1    0.000  -14.525   0.000
```

6. NTP not running:

```
ntpq: read: Connection refused
```

6. Clock Synchronization in Linux

This section assists you in clock synchronization setup on Linux-based systems.

6.1. Timestamping Accuracy

In Linux, the **timestamp mode** parameter, like in Windows, does not exist. This is not a problem, since the timestamping process in Linux has already good enough resolution. Also, Linux systems have typically *NTP* running with parameters suiting also to measurements, while, of course, it is still likely that the parameters are not optimal, and tuning might be useful. Especially, select as close NTP server (in terms of network delay) as possible. In addition, check that the *minimum and maximum poll values* are small enough. How to tweak the NTP parameters can change from one distribution to another.

Besides NTP, *PTP* is typically also available, enabling better accuracy for the system clock synchronization. In good conditions (a lightly loaded LAN), PTP can yield a microsecond-level accuracy.

6.2. PTP in Linux

If *GNSS* clock source is not available, PTP is the second best option to synchronize the machines where you run Qosium Probe. In the simplest case, PTP works in a way that there is a Master device and then Slave devices around the network synchronize to that. Thus, if you have a server in the network, which already has an accurate system time, and is accessible, set that as the Master, and let the other machines synchronize into that. If, however, you have only two devices communicating directly, you can select either one of them as the Master.

A good PTP solution is *PTPd*. Setting a Master service with *PTPd* is done as follows:

```
ptpd --masteronly --interface <network interface over which PTP synch messages will be
sent>
```

PTP Slave service is set as:

```
ptpd --slaveonly --interface <network interface over which PTP synch messages will be
sent>
```

In addition, it is useful to use `--verbose` argument to see that Slave finds a Master and also to see the observed time drift. There's no need to switch the system timesyncd service off. Everything should start working directly. PTP messages are sent by default as multicast. If you want to use unicast mode instead, use the following arguments in both the Master and the Slave: `--unicast` and `--unicast-destinations`.

7. Glossary

Global Navigation Satellite System

A general term under which all the different global satellite navigation systems (e.g., GPS, GLONASS, Galileo, BDS) fall.

Network Time Protocol

A very common protocol for synchronizing the clocks of devices across a network.

Precision Time Protocol

A protocol for synchronizing the clocks of devices across a network. The reached synchronization accuracy is typically considerably better than with NTP.