

Installing Probe

These instructions comprise Qosium Probe installation to different operating systems.

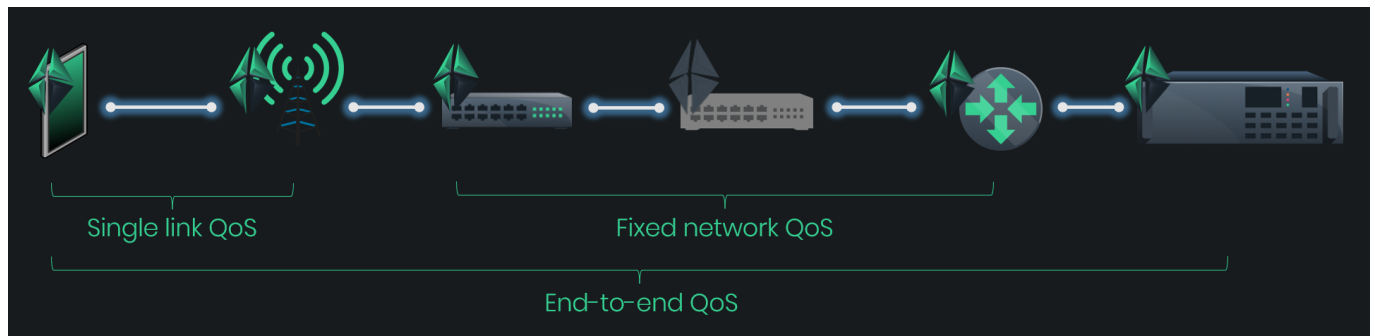
Table of Contents

- 1. Taking Qosium Probe into Use 4
- 2. System Requirements 4
 - 2.1. Operating System 4
 - 2.2. Equipment 4
 - 2.3. Additional Software 5
 - 2.4. External GNSS Receiver 5
- 3. Installation on Windows 5
 - 3.1. Installing a Packet Capture Library on Windows 6
 - 3.1.1. Npcap 6
 - 3.1.2. Win10Pcap 7
 - 3.1.3. WinPcap 7
 - 3.1.4. Parameterization: Timestamping Mode 7
 - 3.2. Clock Synchronization in Windows 8
 - 3.2.1. Introduction 8
 - 3.2.2. NTP Client: Automatic Configuration 8
 - 3.2.3. NTP Client: Manual Configuration 9
 - 3.2.4. NTP Server Configuration 9
- 4. Installation on Debian-Based Systems 10
 - 4.1. Preparations 10
 - 4.2. Package Pre-checks (Optional) 10
 - 4.3. Installation 10
 - 4.4. Uninstallation 10
 - 4.5. GNSS Setup 11
 - 4.5.1. Prerequisites 11
 - 4.5.2. Step-by-Step Instructions for ntp 11
 - 4.5.2.1. Checking the Status 13
 - 4.5.3. Step-by-Step Instructions for chrony 14
 - 4.6. Clock Synchronization in Linux 17
 - 4.6.1. Timestamping Accuracy 17
 - 4.6.2. PTP in Linux 17
- 5. Installation on Android 18
 - 5.1. Rooting 18
 - 5.2. Installation 18
- 6. Installation on Red Hat Based Operating Systems 19
 - 6.1. Preparations 19
 - 6.2. Installation 19
 - 6.3. Uninstall 19
- 7. OpenWRT 19
 - 7.1. Preparations 20
 - 7.2. Installation 20
 - 7.2.1. Install from tarball 20
 - 7.3. Uninstall 20
 - 7.3.1. Uninstall with tarball installation 20
- 8. Installation on Other Operating Systems 21
 - 8.1. MacOS 21
 - 8.2. Equipment of Axis Communications® 21
 - 8.3. Other Linux Systems & FreeBSD 21

9. Glossary 22

1. Taking Qosium Probe into Use

The locations of Qosium Probes in a network determine the measurement possibilities, that is the measurement paths, as illustrated in the figure below. Qosium Probes are installed to nodes from which measurement results are wanted to be collected. As a passive agent, Qosium Probe needs to see the network traffic being measured. If a network device allows installing new software, Qosium Probe is installed directly to it. In case a device is closed, Probe is suggested to be installed as close as possible to it, or by mirroring traffic to a separate measurement device in the network where Qosium Probe is running.



Qosium Probes come with a registration feature, which helps identify devices available for measurement. It is possible to set a unique identifier for Probes. This is useful when IP addresses of network devices are dynamic. With the unique identifier, a network node can be identified despite the IP address it is currently assigned with.

2. System Requirements

2.1. Operating System

Most of the Windows and Linux based operating systems are supported. See platform-specific instructions for further information.

2.2. Equipment

- The measurement point platform
 - Computer / network device / embedded system, or similar
 - Two devices needed for an end-to-end measurement
- Performance considerations
 - Storage space: < 10 MB
 - Minimum reasonable amount of available system memory: ~ 20 MB
 - Higher system memory is recommended for enabling measurements of higher packet rates
 - Low memory restricts the number of simultaneous measurement controllers per Probe.
 - The processing power of the computer sets the limit to the packet rate of what can be analyzed
- Network interfaces, which are supported by the used packet capturing library

2.3. Additional Software

- A packet capturing library
 - Npcap, WinPcap, or Win10Pcap on Windows
 - libpcap on Linux and other systems

2.4. External GNSS Receiver

With an external GNSS receiver, accurate time can be got in addition to location. This allows synchronization of the system clocks of the measurement points accurately, enabling high-accuracy delay measurements.

- GNSS receiver device providing PPS output and supporting NMEA or TSIP-protocol. Tested with:
 - Garmin GPS18x LVC
 - Trimble Acutime
- PPS (Pulse Per Second) to Serial interface converter
- Serial port in the PC
 - PCI-based serial port adapters will most likely also work well enough
 - USB-based serial port adapters are not recommended: tests have shown them to cause a milli-second level error to the PPS pulse



The delay measurement is the only one affected by the clock synchronization. Regardless of the synchronization of the clocks, other QoS statistics can be safely measured.



Satellite clock synchronization is currently not supported in Windows, but if this is an interesting feature for you, contact Kaitotek, since this is in our road map.

3. Installation on Windows

Qosium has an easy-to-use installer available for Windows. The installation process is very straightforward and is completed typically within a minute.

Qosium's Windows installer contains typically the Qosium *GP* including **Qosium Probe**, **Qosium Scope**, **Qosium Scopemon**, and **Qosium Scope Lite**. A separate installation package only for Qosium Probe can be provided as well if needed.

In some cases, the customer has their own installation routines including software packaging. Thus, if needed, Kaitotek can also deliver Qosium Probe's pure binary files without the installation package.

Follow these steps to install Qosium:

1. Install [a packet capture library](#), if not already installed
 2. [Sign in](#) to your account
 3. Download the installer from [your downloads page](#)
 4. Launch the installer: an installation wizard guides you through the installation process.
- At the end of the installation, the wizard asks for permission to set up the timing related parameters. If the clock synchronization settings of the environment are already handled, you can ignore this, but if not, it is recommended to allow Qosium to perform the setup.

- If you chose to ignore the timing settings, take a look at, still, the [packet capturer timestamping parameterization](#), since that needs some attention.

After completing these steps, Qosium Probe has been successfully installed to your system.

3.1. Installing a Packet Capture Library on Windows

The typical installation package of Qosium does not contain a packet capturing library. If a such library has not been installed in the system earlier, it has to be installed manually. There is also some parameterization to consider.

In Windows, there are at least three supported alternatives for packet capturing, which are discussed in the following sections.

Probe 1.9.2.0

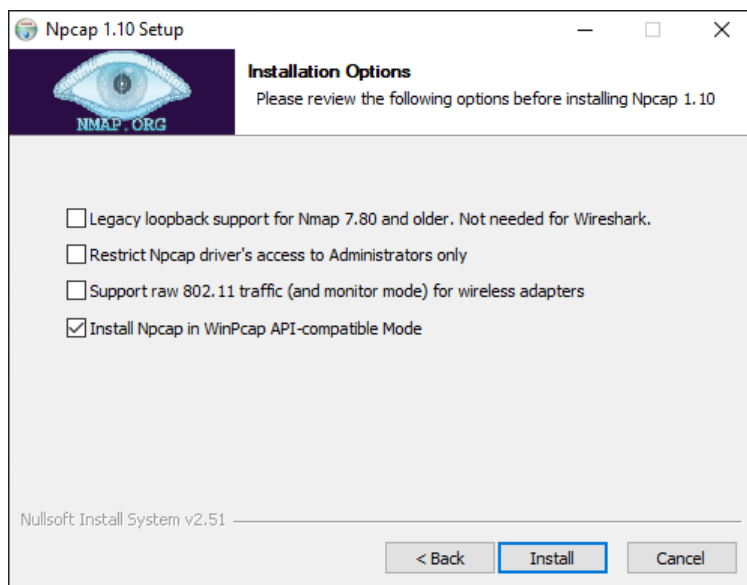
Only Npcap is supported from this version forward.

3.1.1. Npcap

Npcap is the recommended packet capturing library to use with Qosium. It is originally based on WinPcap, but it is under continuous development, unlike WinPcap. Npcap uses *NDIS* 6.x, so it should support also those *NIC*'s that WinPcap no longer does. Please bear in mind that in contrast to WinPcap, Npcap's free use is not unlimited. Thus, please check the license conditions of Npcap if you are considering using it.

[Download Npcap](#)

If you choose Npcap, the older versions of Qosium (Probe version before 1.9.2.0 and Scope version before 2.8.1.5) require it to be installed in WinPcap-compatible mode. See below what needs to be checked in the installation wizard. The other options shown in the figure are up to the needs – they are neither limited nor required by Qosium.



If you need to make modifications to the driver's parameters, restart the driver, e.g., from the command prompt (with superuser rights) to enable the changes.

To shut down:

```
net stop npcap
```

To start:

```
net start npcap
```

3.1.2. Win10Pcap

Another alternative for a packet capturing driver for Windows is Win10Pcap. Like Npcap, this is also based on WinPcap but has the support for the NDIS 6.x driver model. Win10Pcap works directly with the older versions of Qosium. Win10Pcap is fully free to use since it is under the GPLv2 license. It is, however, likely that the driver is no longer developed: the newest version is from Oct. 8, 2015. In any case, it is still newer than the original WinPcap and seems to work.

One technical downside is that it is not confirmed whether the different timestamp modes are supported or not in Win10Pcap. At least, with tests carried out, we haven't yet been successful in modifying the timestamp mode.

[Download Win10Pcap](#)

3.1.3. WinPcap

WinPcap (version 4.1.3) is one option for the older versions of Qosium. Despite the fact that it has not been updated for years, and uses the nowadays deprecated NDIS 5.x, it still works with most of the system configurations at least in Windows 10. Thus, if your NIC works with WinPcap, there is no imminent reason to stop using it.

[Download WinPcap](#)

3.1.4. Parameterization: Timestamping Mode

Npcap and WinPcap support several timestamp modes. With the default mode 0, the absolute accuracy is "a one-shot accuracy", since system performance counters are used for providing high timestamp resolution, but the absolute time is not synchronized. This means that once the driver is turned on, the absolute time is once got from the system clock, but after this, system performance counters will be used for updating the time. Hence, the absolute time begins to drift, and the accuracy of Qosium's delay measurement gets worse as a function of measurement time. However, the resolution is high – even better than in Linux – leading to highly accurate jitter calculation.

If you are interested in one-way delay measurements, it is recommended to change the timestamp mode. One option supported by all known versions of Npcap and WinPcap is 2, in which case, the system clock is used for timestamping packets. This relieves the one-shot accuracy problem but has a downside that it makes the accuracy of jitter calculation worse. This is thanks to the Windows' system clock resolution that is typically only 1 ms. In fact, the default Windows' system clock resolution is as bad as 15.6 ms, but when Qosium is in use, it increases the resolution to 1 ms. For most of the practical applications this 1 ms resolution is already enough, but of course, it is very far from the μ s-level resolution that can be reached with timestamp mode 0.

The newer versions of Npcap support also timestamp mode 4. In this, the system clock is used for absolute timing, but system performance counters are used to calculate the accurate time between the sparse clock ticks. If you can use a new Npcap, this is the recommended mode, since it provides highly accurate timestamps without losing the absolute timing synchronization.

If you allow, Qosium's installation wizard detects automatically the installed Pcap version and selects

timestamp mode 4 when available and mode 2 otherwise. If not, here are the instructions on how to do it manually. The timestamping mode can be changed in the registry. Prior to this, the packet capturing driver must be stopped.

1. Stop the measurement first (and all SW using Pcap, such as Wireshark in addition to Qosium)
2. Open the command prompt and stop the packet capturing driver as explained earlier in the context of Npcap
3. Open *Registry Editor* and navigate to
WinPcap: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NPF`
Npcap (below 0.9985): `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\npf\Parameters`
Npcap (version 0.9985 and higher):
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\npcap\Parameters`
4. Locate the parameter **TimestampMode**. If it does not exist, create it (REG_DWORD).
5. Change/set the mode to the desired value
6. Start the packet capturing driver

Probe 1.9.2.0

From this version forward, manual timestamp mode parameterization is no longer needed. Qosium Probe is itself capable of selecting always the best available mode.

3.2. Clock Synchronization in Windows

Windows has an inbuilt NTP client, so there is no need to install another NTP software. However, the standard settings in the NTP client are only meant for rough system timing. These settings can be modified in order to get better results.

3.2.1. Introduction

System clock synchronization in Windows is carried out by its own *NTP* client. In the Services-list it can be found with the name Windows Time and as a driver, it is called *W32Time*. The default parameters of the Windows NTP client are not at their best for accurate measurement purposes. By tweaking the parameters, typical NTP absolute clock accuracy of some milliseconds can be reached. This requires a high-quality network connection with low jitter to the NTP server. If synchronization is carried out over wireless, such as WiFi, the accuracy will be worse.

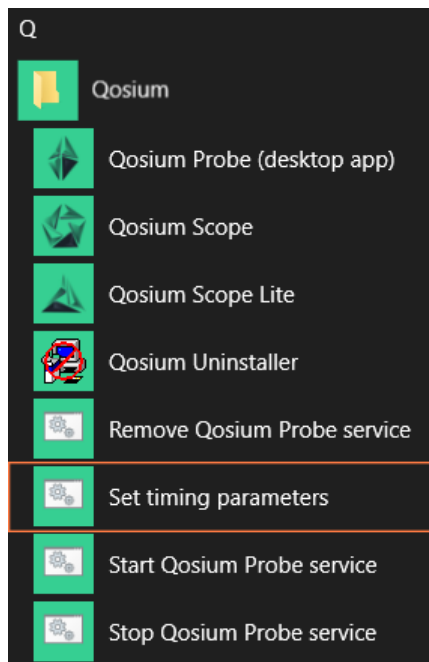
With Qosium's installation package, a set of Windows NTP parameters is also given. By using these, typically a reasonable accuracy for measurement is reached. Those are discussed in the next section.

If the goal is to perform highly accurate delay measurements, other clock synchronization methods are recommended. By using an external *GNSS* clock with Qosium, it is possible to reach about 50 μ s absolute timing accuracy for delay measurements with 1 μ s jitter resolution (at its best). This, however, requires a custom synchronization driver, which is currently at an experimental state. If you are interested in this, please contact Kaitotek's support.

The W32Time driver in the newer Windows 10 based systems supports also *PTP*. This leads to much better accuracy when compared to NTP. If you are interested in this, contact Kaitotek's support for instructions.

3.2.2. NTP Client: Automatic Configuration

Qosium's installation wizard will ask at the end permission to tweak the Windows' NTP client settings. If you did not allow this, it can be done later from the shortcut. Remember, you need to run the script with superuser rights.



3.2.3. NTP Client: Manual Configuration

Windows NTP client parameters can also be set manually. Qosium's installation package contains a registry file that can be taken into use easily. It is found at the Qosium's installation folder with name `win_timing_conf_<code>.reg`, where the `<code>` part is dependent on the distribution. Before activating the new settings, you can also edit them by a text editor. Alternatively, you can set the values directly with Registry Editor.

To activate the parameters from the file:

1. Open Windows Command Prompt with superuser (system administrator) rights
2. Stop the W32Time service by typing: `net stop w32time`
3. Go to the Qosium's installation folder by using, e.g., File Explorer
4. Double-click the file `win_timing_conf_<code>.reg` and allow it to be entered to Registry
5. At the Command Prompt, start the W32Time service by typing: `net start w32time`

Please note that it might take some minutes before the synchronization takes effect.



There might also be other timing clients installed to the system. In this case, in order to use the Windows' own NTP client, find the other clients and shut them down.

3.2.4. NTP Server Configuration

Windows NTP can also be used in server mode to distribute time to NTP clients. The NTP server is often, however, turned off by default. To check and turn on the NTP server, do the following:

1. Open Windows Registry Editor tool
2. Go to:
Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpServer
3. Check that the value **Enabled** is set to 1. If not, change it.
4. Close Registry Editor

If you made changes, you need to restart the Windows Time service. This can be done in many ways, e.g.,

by using Command Prompt:

1. Open Windows Command Prompt with superuser rights
2. Stop the W32Time service by typing: `net stop w32time`
3. Start the W32Time service by typing: `net start w32time`

4. 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.

4.1. 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.

4.2. 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
```

4.3. 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
```

4.4. Uninstallation

To remove Qosium Probe from the device, run:

```
sudo dpkg -r qosiumprobe
```

4.5. GNSS Setup

In Linux, GNSS positioning is enabled by `gpsd` and GNSS-based time synchronization is enabled by `ntp` or `chrony`. The following guide expects that you have a compatible GNSS receiver.

4.5.1. Prerequisites

Using *GNSS* for timing is a two-phased process. First, to get coarse clock synchronization within the correct second, the *full time* is taken from the NMEA messages fed by the GNSS device. Then, the *PPS* signal is used to fine-tune clock synchronization within one second, reaching even microsecond-level accuracy. You could also take the full time from another source, like that provided with *NTP*, but since GNSS also delivers the full time, it is convenient to use that.

To start with, you need a compatible GNSS unit. Any GNSS with PPS output 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` or `chrony`
- `setserial`
- `pps-tools` (optional, recommended for debugging problems)
- `gpsd-clients` (optional, recommended for debugging problems)

You need `ntp` or `chrony` with PPS clock support. By default, all newer `ntp` and `chrony` implementations should support PPS. Bear in mind that `ntp` and `chrony` with GNSS/PPS are used just as tools to adjust the clock, not to control the clock with NTP.

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.

4.5.2. Step-by-Step Instructions for `ntp`

This section gives the instructions when you are using `ntp`. If you are using `chrony`, see the next section.

This section 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
```

Sometimes you need to tell *gpsd* manually the device you intend to use. To do this, modify the configuration file at */etc/default/gpsd*. There, add your device(s) to the device list, e.g.,

```
...  
DEVICES="/dev/ttyUSB0"  
...
```

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 pps0test /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.

Enabling the serial port **low_latency** mode can improve 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
```

4.5.2.1. Checking the 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 1  11  16  377  0.000  -0.011  0.019
*SHM(0)         .GPS.          0 1  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 1 - 16  0    0.000  0.000  0.000
SHM(0)          .GPS.         0 1 - 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 1 - 16  0    0.000  0.000  0.000
*SHM(0)         .GPS.         0 1  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 1 - 16  0    0.000  0.000  0.000
xSHM(0)         .GPS.         0 1 - 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 1  3  64  1    0.000 -14.525 0.000
```

6. NTP not running:

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

4.5.3. Step-by-Step Instructions for chrony

This section gives the GNSS positioning setup and timing synchronization instructions when you are using chrony.

In the following instructions, it is assumed that the PPS signal is connected to the device's integrated serial port at */dev/ttyS0*, and NMEA messages are received through a USB-serial converter via */dev/ttyUSB0*. Port numbers may vary depending on your system, so please check and change them accordingly.

If you have *ntp* installed and you wish to start using *chrony*, remove *ntp* first. To install the required packages, type:

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

Just in case, if *gpsd* was already installed, and is running, stop it:

```
sudo systemctl stop gpsd  
sudo systemctl stop gpsd.socket
```

Set the NMEA message reception speed correctly according to your settings, e.g., 4800 bauds:

```
sudo stty -F /dev/ttyUSB0 4800
```

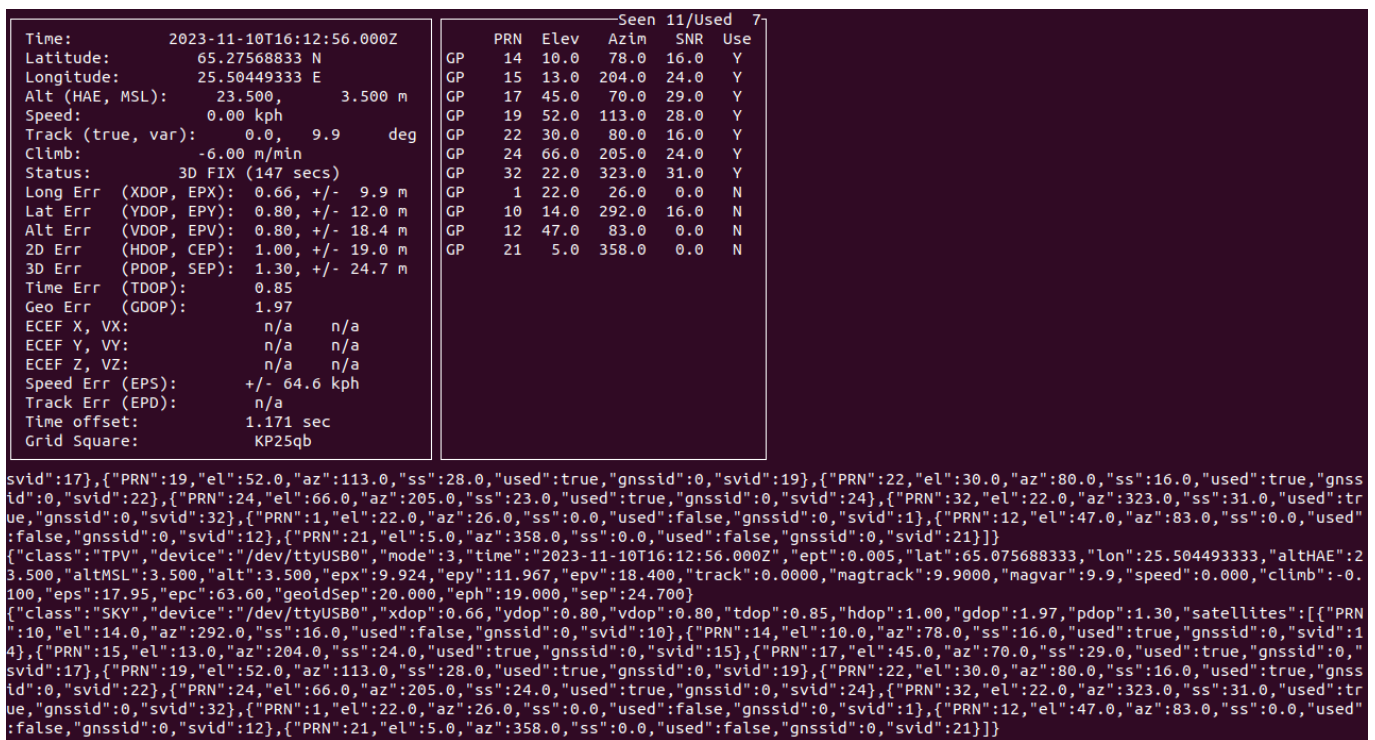
Start gpsds manually:

```
sudo gpsd -n /dev/ttyUSB0
```

Now you should be receiving the NMEA message data. Check it:

```
cgps
```

You should see something like the following:



The messages in the lower part of the terminal show the NMEA data. The upper-right box lists the found satellites, and the upper-left box shows the data extracted. If you don't see any data, check the connections and the previous steps.

If you do get data, check that the Status in the upper-left box shows *fix* like in the example figure. If you don't have fix yet, wait for it. It will take some time to synchronize with GNSS (as long as 12 minutes in the worst case during the cold start). After you get fix, you also get the location and the time information shown in the upper part of the box.

Now, you have successfully set up the GNSS positioning for your device. When parameterized, Qosium Probe can read location directly from gpsd. Next, we will configure clock synchronization and start with PPS. Attach a line discipline to the PPS serial port to create a PPS source:

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

The PPS source will appear as a new device, typically: `/dev/pps0`. However, sometimes there might already be PPS devices in the device list. The newly created PPS source will be the last of the PPS devices shown in the device list. To check the device list, type:

```
ls /dev
```

To verify that you are receiving the PPS data, e.g., for `/dev/pps0`, type:

```
sudo ppsstest /dev/pps0
```

When it works, you see the timestamps for the rising and lower edges of the PPS pulse arriving once per second. If you don't, check the connections and the previous steps.

Enabling the serial port **low_latency** mode can improve jitter:

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

Now we are ready to configure chrony. Open the configuration file `/etc/chrony/chrony.conf` for editing and add there the following lines:

```
refclock PPS /dev/pps0 lock NMEA  
refclock SHM 0 offset 0.5 delay 0.2 refid NMEA noselect
```

The first line ties the reference clock to a PPS source at `/dev/pps0` and locks it to another reference lock, called `NMEA`, providing the full time. The second line parameterizes the NTP shared memory driver, `SHM`, capable of receiving timing samples from another processes, like `gpsd` now. The first parameter is the shared memory segment being 0, typically. The rest of the parameters are:

- **offset** (seconds): specifies a correction if there is a delay in the time source. Now, we only need to hit the correct second, so setting 0.5 seconds here decreases the likelihood of locking to an incorrect second.
- **delay** (seconds): sets the NTP delay of the source, relating to the source selection algorithm.
- **refid**: specifies the reference ID of the refclock. We have now set here `NMEA` for convenience, but it is not a reserved name. You can set there any four-ASCII character ID you wish, but remember to refer to the same ID in the PPS reference clock.
- **noselect**: means that this source is never selected as the dominating source for clock synchronization.

Start the chrony service:

```
sudo systemctl start chrony
```

Now, everything should be operational. You can check the situation with instructions:

```
chronyc sources -v
```

When synchronization is working, you should see something like the following:


```
210 Number of sources = 2
```

```
.-- Source mode '^' = server, '=' = peer, '#' = local clock.
/ .- Source state '*' = current synced, '+' = combined, '-' = not combined,
| / '?' = unreachable, 'x' = time may be in error, '~' = time too variable.
||                                     | - xxxx [ yyyy ] +/- zzzz
|| Reachability register (octal) -.   | xxxx = adjusted offset,
|| Log2(Polling interval) --.       | yyyy = measured offset,
||                                     | zzzz = estimated error.
||                                     |
MS Name/IP address          Stratum Poll Reach LastRx Last sample
=====
#* PPS0                      0  4   377   17  +589ns[+1251ns] +/-  486ns
#? NMEA                      0  4   377   15  +41ms[ +41ms] +/- 104ms
```

As seen, there is * next to PPS, meaning it is now in use and working. The ? in the NMEA line only means that it is not in use, but as there are samples received, PPS has initially used it to fetch the full time. Otherwise, PPS would not claim to be synchronized. You might also see some NTP servers below, depending on your chrony configuration, but they are not in use either because of our chrony settings. If you do not get sync for PPS after a while, check your cables, settings, and the earlier steps.

4.6. Clock Synchronization in Linux

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

4.6.1. 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. If you are aiming for microsecond-level accuracy, and fixed connections for PTP are not possible, [GNSS clock synchronization is recommended](#).

4.6.2. 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`. So, in the case of unicast, you need to tell the Master by parameters who is allowed to connect. If a particular Slave is not on the Master's list, synchronization will not work.

Sometimes PTP is desired to be tied into a *PTP domain*. This is done with the switch `--domain`. Remember that if the Master uses a certain domain, you need to use the same domain in the Slave.

In addition to command line parameters, PTPd can be configured by using a configuration file.

5. Installation on Android

Qosium Probe can be installed on Android devices, such as smartphones and tablets. For Probe to be able to capture traffic, it needs to be run with superuser rights. Thus, you need to root the device or have another way to install software as a system service. This article explains rooting and installation process of Qosium Probe for Android.

5.1. Rooting

If you have a way to install Qosium Probe as a system service or you already have superuser rights (root permissions) to the Android device, you can skip this part of the instruction.

The first step in Qosium installation is to acquire superuser permissions in Android device. The so called *rooting* is always device specific. Some manufacturers provide support for unlocking their devices, while others do not. In both cases, it is advisable to search reliable instructions and carefully follow them.



Kaitotek cannot and will not take any responsibility on the possible damage caused by rooting procedures. However, we have experience on a set of devices and can give recommendations or instructions managing those devices.



Rooting usually requires unlocking bootloader, which leads to voiding the warranty on the phone! It is advisable to dedicate a specific device for Qosium installation and not to use personal devices.

5.2. Installation

Qosium installer comes as an Android APK package. The installation follows the normal process that comes to the packages outside the Google Play store. There are a couple of different methods for performing the installation. The tested methods are using a specific APK Installer application (available at Google Play store) and by using `adb install` command (adb is part of the Android SDK).

The following instructions make use of the APK Installer application:

1. Download the Qosium Probe's installation package (typically named as `QosiumProbe-<version_information>.apk` to the Android device for example, from a memory card, or with a browser from you Kaitotek account
2. Open APK Installer and install the just downloaded package

3. Locate **QosiumProbe** application from the application menu and start it
4. If previously installed Qosium Probe agent process is running on background, close it by tapping the **Stop** button
5. If Qosium Probe agent was previously installed remove it by tapping **Uninstall** button
6. Execute the Qosium Probe agent installation procedure by tapping **Install** button

6. Installation on Red Hat Based Operating Systems

Qosium Probe installation package is provided as a tar.gz package and the installation happens in command line interface. The same installation method works on CentOS, Red Hat, and other operating systems that are similar to Red Hat.

6.1. Preparations

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

6.2. Installation

First, extract the tar.gz package by opening a *Terminal*:

```
tar -zxvf QosiumProbe_<version_details>.tar.gz
```

Then, go into the extracted directory by typing:

```
cd QosiumProbe
```

Run the install script with superuser rights:

```
./install-QosiumProbe.sh
```

The installation procedure asks whether you want to set Qosium Probe as a Systemd service or to be started manually.

The installed Qosium Probe files can be found at `/opt/QosiumProbe/`.

6.3. Uninstall

The uninstall script is located in the same extracted folder where the `install-QosiumProbe.sh` was. To uninstall Qosium Probe, run the following in Terminal with superuser privileges:

```
uninstall-QosiumProbe.sh
```

7. OpenWRT

Qosium Probe installation package is provided as an ipk package to be managed with the opkg package manager.

7.1. Preparations

First, [sign in](#) to your account and access [your downloads page](#). Then download Qosium products for the target machine. The package is typically named as `qosiumProbe_<version_details>.ipk`, where the `<version_details>` part includes the Qosium Probe version and other delivery-specific identifiers.

7.2. Installation

Installation happens through the `opkg` package manager. Transfer the install file to the device and run:

```
opkg install qosiumProbe_<version_details>.ipk
```

The installation procedure asks whether you want to set Qosium Probe as a background service (creates a `procd` init script to `/etc/init.d/`) or to be started manually. You can also set a unique *Service ID* for the Probe.

The installed Qosium Probe files can be found at `/opt/QosiumProbe/`.

7.2.1. Install from tarball

If you wish to get the installer as a tarball file, the installation happens in the following way:

First, move the tarball file to the device and extract the compressed tar file:

```
tar -zxvf QosiumProbe_<version_details>.tar.gz
```

Then, go into the extracted directory by typing:

```
cd QosiumProbe
```

Run the install script with superuser privileges:

```
./install-QosiumProbe.sh
```

7.3. Uninstall

Uninstallation happens with `opkg`:

```
opkg remove QosiumProbe
```

7.3.1. Uninstall with tarball installation

If you installed Probe from a tar.gz file, the uninstall script is located in the same extracted folder where the `install-QosiumProbe.sh` was. To uninstall Qosium Probe, run the following in command line with superuser privileges:

```
uninstall-QosiumProbe.sh
```

8. Installation on Other Operating Systems

Qosium Probe can be installed to a large number of different operating systems. Below you can see instruction for some of them. Kaitotek will provide you with more instructions regarding these operating systems upon delivery.

8.1. MacOS

Qosium Probe installer is provided as a DMG, mountable disk image, file. Installation happens over a graphical window by drag and dropping Qosium Probe to *Applications* directory.

8.2. Equipment of Axis Communications©

Qosium Probe is provided as an official Axis software installation package. The installation is carried out by using the standard software installation mechanisms of Axis.

8.3. Other Linux Systems & FreeBSD

Qosium Probe is provided as a *tar.gz* package, typically named as `QosiumProbe_<version_details>.tar.gz`, where the `<version_details>` part depends on version to another and may contain many identifiers.

First, extract the tar.gz package in a command line interface:

```
tar -zxvf QosiumProbe_<version_details>.tar.gz
```

Then, go into the extracted directory by typing:

```
cd QosiumProbe
```

Run the install script as superuser:

```
./install-QosiumProbe.sh
```

The Qosium Probe files can then be found from `/opt/QosiumProbe/`.

9. Glossary

General Package

Qosium General Package contains the basic Qosium components required for measurements and monitoring.

Network Driver Interface Specification

NDIS on Windowsin ohjelmointirajapinta verkkokortelle

[Wikipedia article on Network Driver Interface Specification](#)

Network Interface Card

A piece of hardware which offers a device a networking interface.

Network Time Protocol

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

Global Navigation Satellite System

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

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.

Pulse per second

A square-like electrical signal that is used in accurate clock synchronization