

2. August 2021

**Apprentice Project – set up a stratum 1 ntp timeserver with an RPi 4**

This time I’ll write in English as I followed an excellent guide from Johannes Weber from Weberblogs to set up an Stratum 1 ntp timeserver with an Raspberry Pi and GPS. My blogpost is a direct "pingback", so English will be best. *I think that teachers and in-company trainers with some ideas can easiely build a small project around the guide.*

In my opinion it can be very suitable for apprentices in IT professions as well as all professions with an electronics background. Maybe as a small project with some pupils, too. So I wrote down my Ideas at the end of the article.

**My configuring Process**

I tried Johannes setup with an RPi 4 and 5.10 kernel on an summer 2021 recent raspian os. So I followed this guide: <https://weberblog.net/ntp-server-via-gps-on-a-raspberry-pi/>. With some soldering (yes, I’m really out of practice…) that works nearly out of the box. Some changes I have seen due the newer operating system I will document here.

While configuring the gpsd the interface ist not /dev/AMA0 it is /dev/ttyS0 everything else with the gpsd is fine.

Johannes mentioned, to set a symbolic link to set up a service for gpsd and ntpd. That is out of date. The gpsd service as well as the ntpd service is already there, when you install from a recent raspian repositority. So after compiling the latest version of ntpd (thats a really good idea!) you’ll just have to set the package for apt-get on hold, as Johannes wrote. To enable and start the services then just type:

sudo systemctl enable ntp && sudo systemctl start ntp

sudo systemctl enable ntp && sudo systemctl start ntp

The enable` command makes the service start automatically after a system restart, and the `start command will start it immediately.

Then you have to make the GPS and PPS related configs in /etc/ntp.conf.

## That is the PPS and GPS part

# pps-gpio /dev/pps0

server 127.127.22.0 minpoll 4 maxpoll 4

fudge 127.127.22.0 refid PPS

# gpsd clock via shm

server 127.127.28.0 minpoll 4 maxpoll 4 prefer

fudge 127.127.28.0 time1 +0.135 refid GPS flag1 1

And well, it works. So far so good. Nevertheless I had to make some other changes in the standard ntpd config file /etc/ntp.conf. It may be useful for people who followed the guide, as well. Several time queries worked well until the moment as the ntp server gets some load (maybe 5 to 10 queries per second). Than the clients will get a 500 error and you’ll will see a lot of timeouts. I searched for that behavior for a while. The solution was to comment out two directives in /etc/ntp.conf. The directives *kod & limited*. After that the server worked as expected even with load.

# By default, exchange time with everybody, but don’t allow configuration.

restrict -4 default notrap nomodify nopeer noquery # kod limited

restrict -6 default notrap nomodify nopeer noquery # kod limited

Here is my hole /etc/ntp.conf file

# /etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

# Leap seconds definition provided by tzdata

leapfile /usr/share/zoneinfo/leap-seconds.list

# Enable this if you want statistics to be logged.

#statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats

filegen loopstats file loopstats type day enable

filegen peerstats file peerstats type day enable

filegen clockstats file clockstats type day enable

# You do need to talk to an NTP server or two (or three).

#server ntp.your-provider.example

# Some IPv4 and IPv6 Stratum 1 servers for timesyncing and backup if GPS is offline

server ptbtime1.ptb.de prefer

server ptbtime2.ptb.de prefer

server ptbtime3.ptb.de prefer

server ntp0.ipv6.fau.de prefer

server ntp1.ipv6.fau.de prefer

server rustime01.rus.uni-stuttgart.de prefer

server ntp01.sixtopia.net prefer

# Note that "restrict" applies to both servers and clients, so a configuration

# that might be intended to block requests from certain clients could also end

# up blocking replies from your own upstream servers.

# By default, exchange time with everybody, but don't allow configuration.

restrict -4 default notrap nomodify nopeer noquery

# kod limited

restrict -6 default notrap nomodify nopeer noquery

# kod limited

# Local users may interrogate the ntp server more closely.

restrict 127.0.0.1

restrict ::1

# Needed for adding pool entries

restrict source notrap nomodify noquery

# Clients from this (example!) subnet have unlimited access, but only if

# cryptographically authenticated.

#restrict 192.168.123.0 mask 255.255.255.0 notrust

# If you want to provide time to your local subnet, change the next line.

# (Again, the address is an example only.)

#broadcast 192.168.123.255

# If you want to listen to time broadcasts on your local subnet, de-comment the

# next lines. Please do this only if you trust everybody on the network!

#disable auth

#broadcastclient

## That is the PPS and GPS part

# pps-gpio /dev/pps0

server 127.127.22.0 minpoll 4 maxpoll 4

fudge 127.127.22.0 refid PPS

# gpsd clock via shm

server 127.127.28.0 minpoll 4 maxpoll 4 prefer

fudge 127.127.28.0 time1 +0.135 refid GPS flag1 1

**Nginx proxyerver for ntpd – yes, you can**

Furtermore I proxied the RPi4 ntp service through an nginx proxy server on an other machine. That could be interesting for all of you which don’t want to expose the whole (firewalled) RPi4 to the internet. As well you can do some load balancing if you have a lot of clients and/or more than one timeserver. In my case its because I only have one public IPv4 that I can use for my servers.

I used the nginx servers *stream* directive with a small custom logging format to do that. Here is my workling config (just set the proper IPs).

stream {

#set Log Format

log\_format upstream ‘$remote\_addr [$time\_local] ‘

‘$protocol $status $bytes\_sent $bytes\_received ‘

‘$session\_time ‘

‘$upstream\_addr’;

# ntp upstream pool

upstream time {

server 192.168.xx.1:123;

# More servers for load balancing

#server 192.168.xx.2:123;

#server 192.168.xx.3:123;

# Finally a local statum 2 server as backup as none of the above servers response

# maybe you can use your routers included timeserver (eg openwrt or fritzbox) for that

server 192.168.xx.4:123 backup;

}

# Time server proxy

server {

listen 123 udp;

listen [::]:123 udp;

# Comment out the log after success cause it will be HUGE!

access\_log /var/log/nginx/time.access upstream;

error\_log /var/log/nginx/time.error;

proxy\_timeout 5s;

proxy\_requests 1;

proxy\_responses 1;

proxy\_pass time;

}

}

I highly recommend to turn of logging after you checked that all works well. The ntp is such a small protocol and the log will be huge in a couple of hours if you expose the timeserver to pool.ntp.org an get some load.

Finnaly my stratum 1 timeserver works well but is somewhat ugly. Nobody will see that, cause it lies on the ceiling and looks like that:

[Ein Bild, das Elektrische Leitungen, Werkzeug, Kabel, Elektronik enthält.

KI-generierte Inhalte können fehlerhaft sein.](https://www.ulrichivens.de/wp-content/uploads/2021/08/PXL_20210731_131410649-scaled.jpg)

**Ideas for the youngsters**

So what are my ideas for the apprentices, trainees and pupils. I think you’ll will have a wide variety of possible tasks and projekts:

1. Figure out how ntp works.
2. Building the server from scratch with a full documentation
3. Make a beautiful housing (eg. 3D printing).
4. Try different antennas. What is the influence on systems jitter and offset
5. Use other time sources like dcf77 or an own pulse generator
6. Take a closer look on time syncing with computers. For what scenarios do you need a highly precice time?
7. Take a look on proxiing udp services
8. Make measurments with the signals
9. Monitoring the servers health and get some statistics
10. …

**Where to find my server?**

You can find my server via IPv4 and IPv6 with the hostname **time.eldshort.de** and – as well – it is included in the ntppool project within the pool ***de.pool.ntp.org***. Right now it handles round about 10-20 queries per second with 2% system load. Due the fact that the protocol is very leightweight it uses about 10 Megayte per 90 minutes, so even with a slow internet access you can expose your timeserver without loosing bandwidth. Have fun with it.

**Licence**

Blogpost "Apprentice Projekt – set up a stratum 1 ntp timeserver with an RPi 4" from Ulrich Ivens is licensed under [**CC BY-SA 4.0**](https://creativecommons.org/licenses/by-sa/4.0/).

Picture of the post: ["Pocket watch"](https://www.flickr.com/photos/56832361@N00/3203909371) by [vidalia\_11](https://www.flickr.com/photos/56832361@N00) is licensed under [CC BY 2.0](https://creativecommons.org/licenses/by/2.0/?ref=ccsearch&atype=html)