GPSD Installation Manual

For Ubuntu 16.04LTS

Navisys Technology Crop.

GPSD Installation Manual

For Ubuntu 16.04LTS

Installation

Install gpsd

$ sudo apt-get update

$ sudo apt-get install gpsd-clients gpsd

Plugin GPS receiver

$ dmesg | tail -n 5

Find the receiver was really mounted to

Edit file "/etc/default/gpsd"

Default settings for the gpsd init script and the hotplug wrapper.

# Start the gpsd daemon automatically at boot time
START_DAEMON="true"

# Use USB hotplugging to add new USB devices automatically to the daemon
USBAUTO="true"

# Devices gpsd should collect to at boot time.
# They need to be read/writeable, either by user gpsd or the group dialout.
DEVICES="/dev/ttyUSB0"

# Other options you want to pass to gpsd
GPSD_OPTIONS="-n"

Restart gpsd

$ service gpsd restart

Check gpsd is running

$ ps aux | grep gpsd

Use the gpspipe command to get gpsd data:

$ gpspipe -w -n 5

```
eric@eric-VirtualBox:~$ gpspipe -w -n 5
{"class":"VERSION","release":"3.15","rev":"3.15-2build1","proto_major":3,"proto_
minor":11}
{"class":"DEVICES","devices":[{"class":"DEVICE","path":"/dev/ttyUSB0","driver":"
SiRF","subtype":"-\u0006GSD4e_4.1.2-P1_RPATCH.05-NS23 06/23/2014 174","activated
":"2017-01-19T05:43:37.520Z","flags":1,"native":1,"bps":4800,"parity":"N","stopb
its":1,"cycle":1.00}]}
{"class":"WATCH","enable":true,"json":true,"nmea":false,"raw":0,"scaled":false,"
timing":false,"split24":false,"pps":false}
{"class":"TPV","device":"/dev/ttyUSB0","mode":3,"time":"2017-01-19T05:43:38.000Z
","ept":0.005,"lat":24.773696667,"lon":121.007343333,"alt":120.800,"epx":2.054,"
epy":2.376,"epv":9.775,"track":0.0000,"speed":0.000,"climb":0.006,"eps":0.04,"ep
c":0.15}
{"class":"TPV","device":"/dev/ttyUSB0","mode":3,"time":"2017-01-19T05:43:39.000Z
","ept":0.005,"lat":24.773696667,"lon":121.007343333,"alt":120.800,"epx":2.054,"
epy":2.376,"epv":9.200,"track":0.0000,"speed":0.000,"climb":0.000,"eps":4.75,"ep
c":18.98}
```

$ gpspipe -w -n 5 | grep -m 1 TPV

```
eric@eric-VirtualBox: ~
eric@eric-VirtualBox:~$ gpspipe -w -n 5 | grep -m 1 TPV
{"class":"TPV","device":"/dev/ttyUSB0","mode":3,"epv":32.200}
eric@eric-VirtualBox:~$ gpspipe -w -n 5 | grep -m 1 TPV
{"class":"TPV","device":"/dev/ttyUSB0","mode":3,"time":"2017-01-19T05:57:34.000Z
","ept":0.005,"lat":24.773698330,"lon":121.007350005,"alt":120.800,"epx":1.952,"
epy":2.395,"epv":8.050,"track":0.0000,"speed":0.000,"climb":0.000,"eps":4.79,"ep
c":16.10}
```

Install ntp

$ sudo apt-get install ntp

Edit file "/etc/ntp.conf"

pool us.pool.ntp.org iburst

driftfile /var/lib/ntp/ntp.drift
logfile /var/log/ntp.log

restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
restrict 127.0.0.1 mask 255.255.255.0
restrict -6 ::1

GPS Serial data reference (NTP0)
server 127.127.28.0
fudge 127.127.28.0 time1 0.9999 refid GPS

GPS PPS reference (NTP1)
server 127.127.28.1 prefer
fudge 127.127.28.1 refid PPS

Restart ntpd

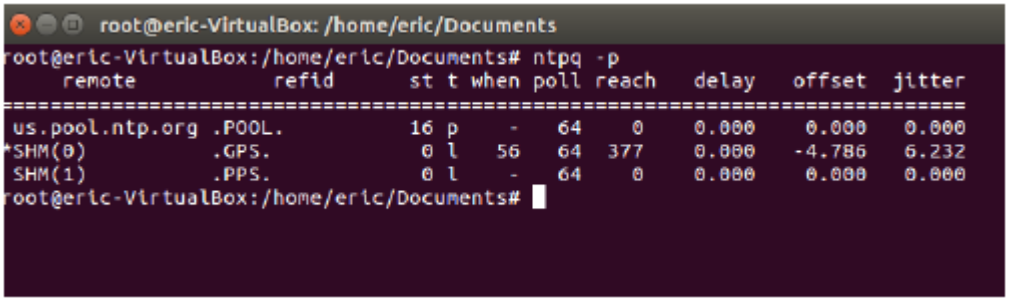
$ service ntp restart

If the difference is too large , "ntpd" won't correct the clock
(may can use "crond" to check and correct big difference first)

Detail method see http://www.catb.org/gpsd/gpsd-time-service-howto.html

Check if work normal

$ ntpq –p


root@eric-VirtualBox: /home/eric/Documents
root@eric-VirtualBox:/home/eric/Documents# ntpq -p
remote refid st t when poll reach delay offset jitter
us.pool.ntp.org .POOL. 16 p - 64 0 0.000 0.000 0.000
*SHM(0) .GPS. 0 l 56 64 377 0.000 -4.786 6.232
SHM(1) .PPS. 0 l - 64 0 0.000 0.000 0.000
root@eric-VirtualBox:/home/eric/Documents#

Retrieve GPS data

Interface

Shell Script: use tools like "gpspipe" to retrieve and parsing data

Socket interface: default is 2947

Shared-memory interface

D-bus broadcasts

Developer Tools

C, C++, Python, Java, Perl

Detail see http://www.catb.org/gpsd/client-howto.html

Install library

```
$ sudo apt-get update
$ sudo apt-get install libgps-dev
```

Example code

```
#include <gps.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <math.h>

int main() {

int rc;

struct timeval tv;

struct gps_data_t gps_data;

if ((rc = gps_open("localhost", "2947", &gps_data)) == -1) {

   printf("code: %d, reason: %s\n", rc, gps_errstr(rc));

   return EXIT_FAILURE;

}

gps_stream(&gps_data, WATCH_ENABLE | WATCH_JSON, NULL);

while (1) {

   /* wait for 2 seconds to receive data */

   if (gps_waiting (&gps_data, 2000000)) {
```

```
    /* read data */
    if ((rc = gps_read(&gps_data)) == -1) {
      printf("error occured reading gps data. code: %d, reason: %s\n", rc, gps_errstr(rc));
    } else {
      /* Display data from the GPS receiver. */
      if ((gps_data.status == STATUS_FIX) &&
        (gps_data.fix.mode == MODE_2D || gps_data.fix.mode == MODE_3D) &&
        !isnan(gps_data.fix.latitude) &&
        !isnan(gps_data.fix.longitude)) {
          //gettimeofday(&tv, NULL); EDIT: tv.tv_sec isn't actually the timestamp!
          printf("latitude: %f, longitude: %f, speed: %f, timestamp: %ld\n", gps_data.fix.latitude, gps_data.fix.longitude, gps_data.fix.speed,
gps_data.fix.time); //EDIT: Replaced tv.tv_sec with gps_data.fix.time
      } else {
        printf("no GPS data available\n");
      }
    }
  }

  sleep(3);
}
/* When you are done... */
gps_stream(&gps_data, WATCH_DISABLE, NULL);
gps_close (&gps_data);

return EXIT_SUCCESS;
}
```

compile it by running gcc -o gps filename.c -lm -lgps