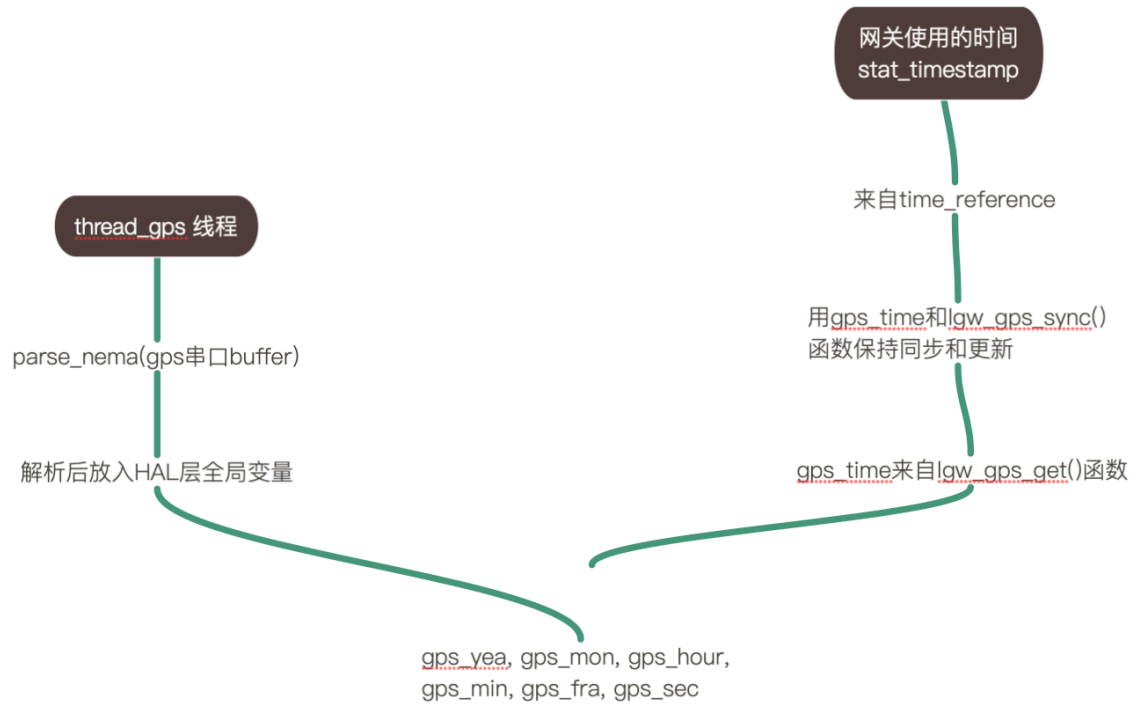
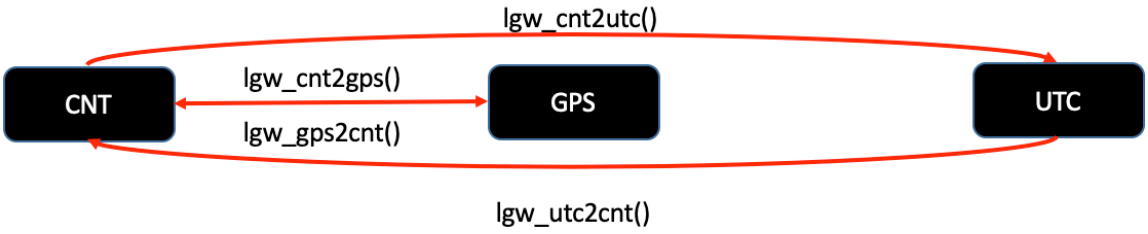
LoRa网关时间flow

葛鑫 2019-1-2



网关的所有时间都来自GPS，但是具体到不同使用场景的时，可能会使用内部计数器，可能使用到UTC时间

因此也具有如下关系



它们之间的转换关系通过一个时间同步线程来进行同步，也就是`thrid_gps`和`thrid_valid`

`thread_gps`不断从串口读取接收到的gps报文，用parser解析，更新底层全局gps时间变量

`thread_valid`不断判断当前time_reference是否有效，修正晶振偏差（calculate XTAL correction）

网关的时间流很复杂，并且众多函数的算法一一都看也很难定位错误，且一些错误校准算法大致一看也看不懂。所以我觉得直接去检验这三个时间，查看它们的变化规律是否线性就可以判断网关的时间转换算法到底是不是有错。

每收到一个包，网关会打上时间戳

```
JSON up: {"rxpk":[{"tmst":322954380,"time":"2019-01-02T19:53:24.747868000Z","tmms":1230465223748,"chan":3,"rfch":0,"freq":433.175000,"stat":1,"modu":"LORA","datr":"SF12BW125","codr":"4/5","lsnr":6.2,"rssi":-51,"size":15,"data":"QKRQegGAHwADfRk7nj0T"}]}
```

`tmst`是内部计数器， `time`是上行收到的UTC时间， `tmms`是gps时间（从1980年1月6日到现在的毫秒数）

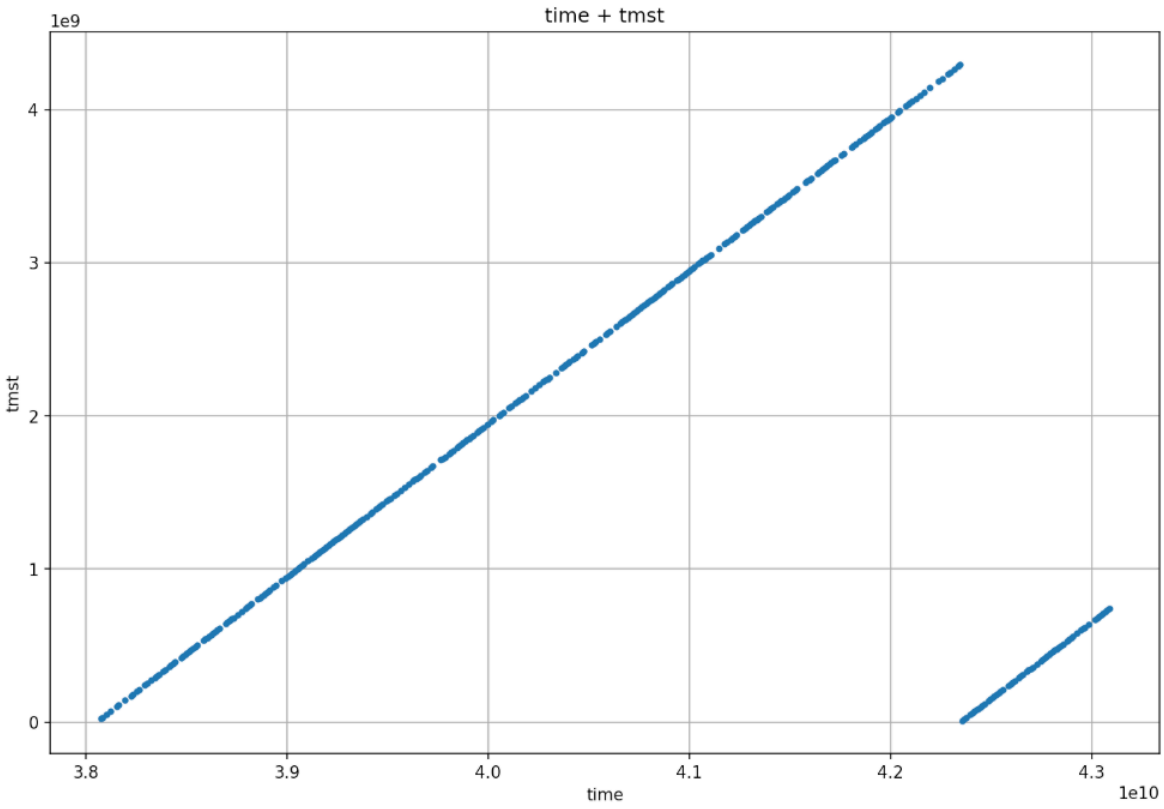
其中代码里`time`字段的变量是这样得到的

```
j = lgw_cnt2utc(local_ref, p->count_us, &pkt_utc_time);
                if (j == LGW_GPS_SUCCESS) {
                    /* split the UNIX timestamp to its calendar components */
                    x = gmtime(&(pkt_utc_time.tv_sec));
                    ...
                }
```

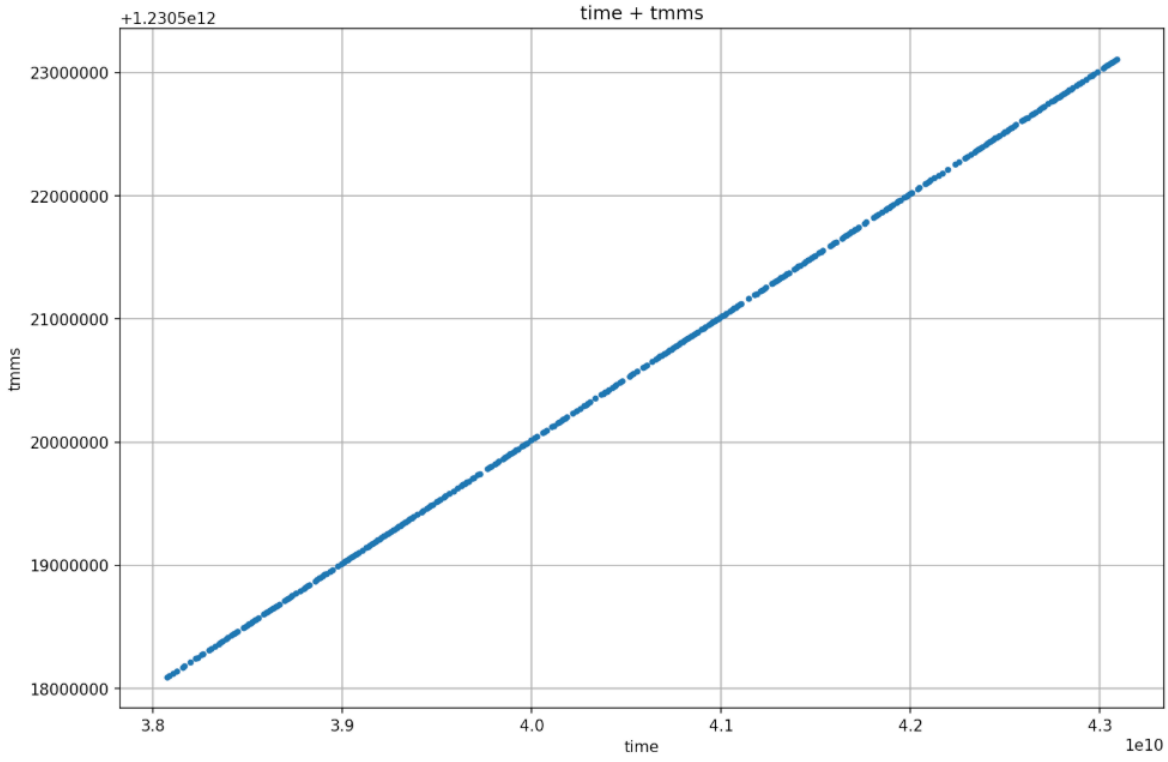
网关的时间全部来自GPS，在某个阶段一定做了GPS与CNT的同步，此时又做了cnt2utc的同步，最终得到了`time`

- 测试方法，终端定时发包，网关记录打印信息，拿到每一个包的三个字段`tmst`， `time`和`tmms`，两两比较，查看线性程度，如果转换算法有错，一定会有点偏离直线，或所有点的趋势不在直线上。
- 测试结果

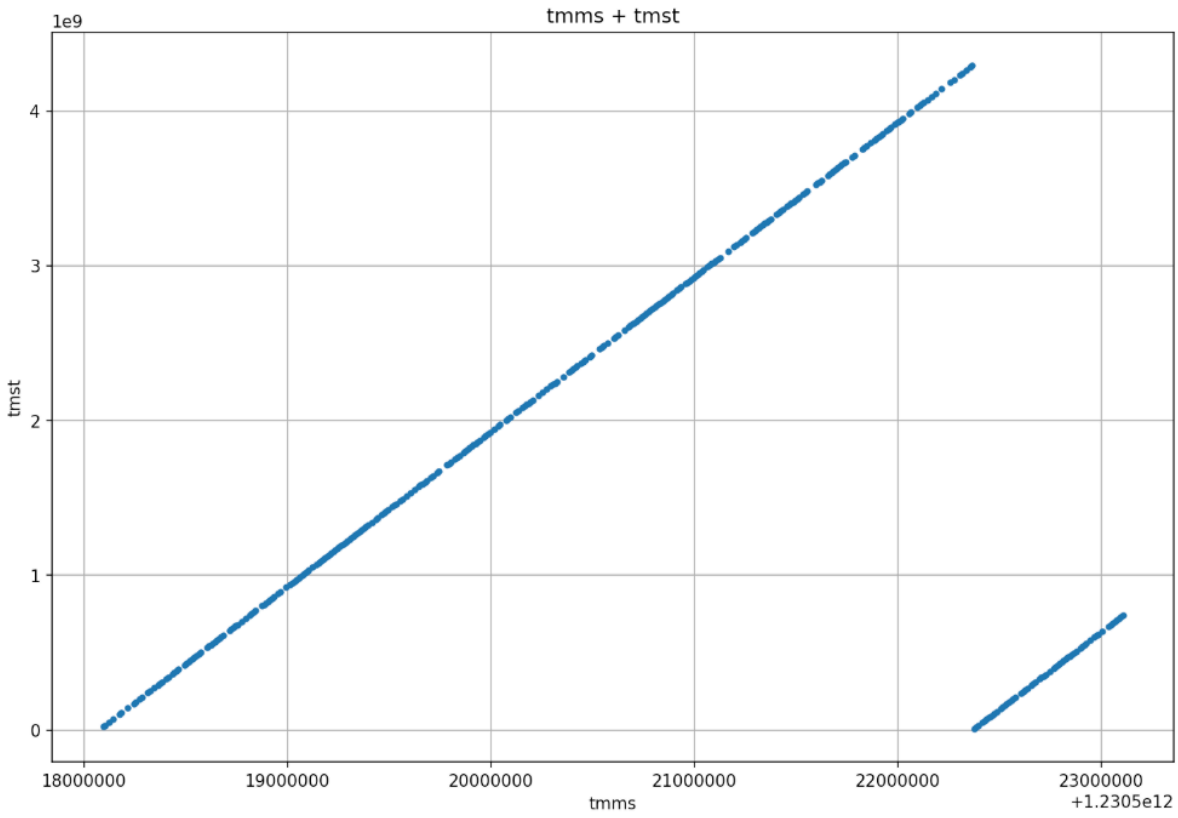
- `tmst`与`time`对比图



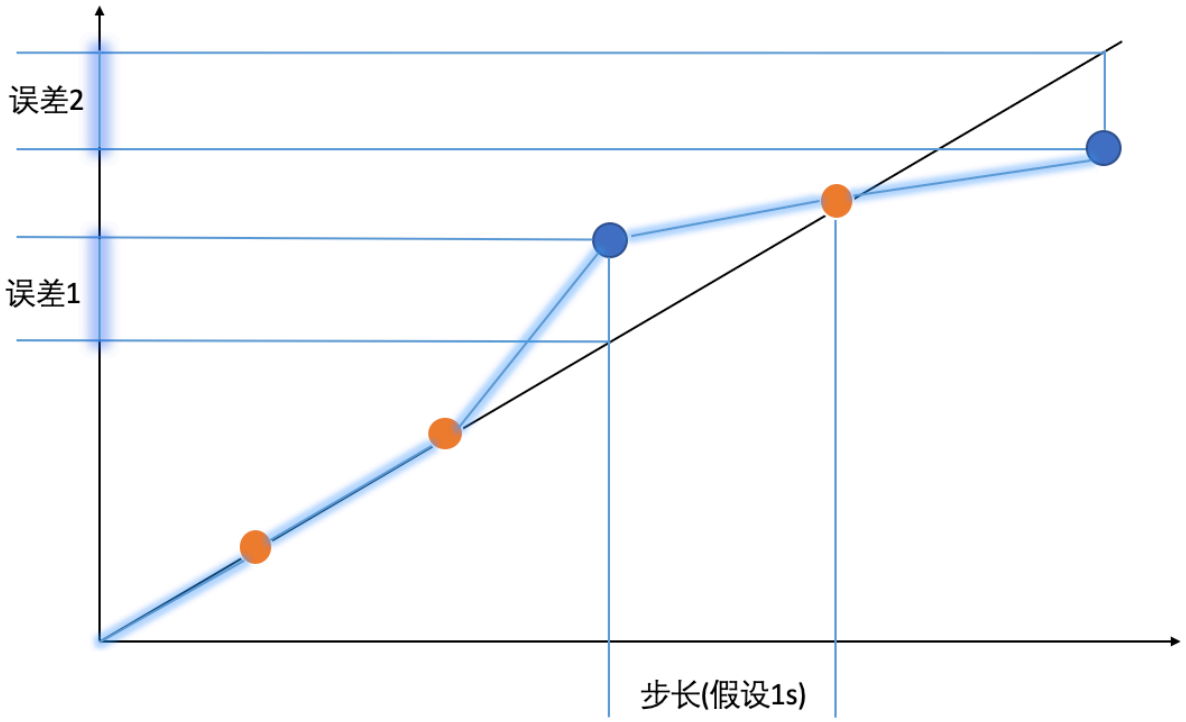
- `time`与`tmms`对比图



- `tmst`与`tmms`对比图



但是由于时间精度需要很高，需要计算能产生的最大误差才能有说服力，因此使用下面的方法去检查最大的误差



经过确认，时间同步线程的更新间隔的确是**1000ms**

结果如下

```
tmst为X time为Y
```

```
avr slop = 1.0
max slop = 1.0517345360804133
min slop = 0.9111464014513971
最坏情况下，每秒同步一次造成的误差如下
对 time 的影响为51.734536080413335 ms/ -88.85359854860286 ms

tmst为X tmms为Y
avr slop = 0.001
max slop = 0.0010001079361018276
min slop = 0.000999928826419633
最坏情况下，每秒同步一次造成的误差如下
对 tmms 的影响为0.10793610182762217 ms/ -0.0711735803670191 ms

time为X tmms为Y
avr slop = 0.001
max slop = 0.001097600077575732
min slop = 0.0009508011485639861
最坏情况下，每秒同步一次造成的误差如下
对 tmms 的影响为97.60007757573207 ms/ -49.198851436013896 ms
```

结论：

似乎有`time`字段的地方的确有精度问题，下一步需要查看下行发送要使用到哪些时间转换函数，以及同步线程的同步周期大概是是多少。如果非常快的话那么基本是不会有问题。如果不是这样，那么有`utc`存在的时间转换函数的算法的确有问题，需要查找。

5.12 DeviceTime commands (DeviceTimeReq, DeviceTimeAns)

This MAC command is only available if the device is activated on a LoRaWAN1.1 compatible Network Server. LoRaWAN1.0 servers do not implement this MAC command.

With the ***DeviceTimeReq*** command, an end-device may request from the network the current network date and time. The request has no payload.

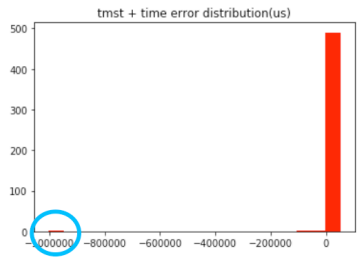
With the ***DeviceTimeAns*** command, the Network Server provides the network date and time to the end device. The time provided is the network time captured at the end of the uplink transmission. The command has a 5 bytes payload defined as follows:

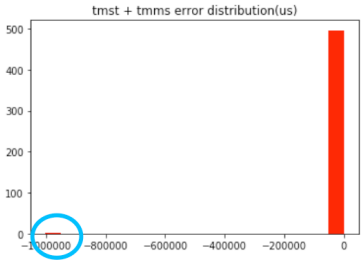
Size (bytes)	4	1
DeviceTimeAns Payload	32-bit unsigned integer : Seconds since epoch*	8bits unsigned integer: fractional-second in ½^8 second steps

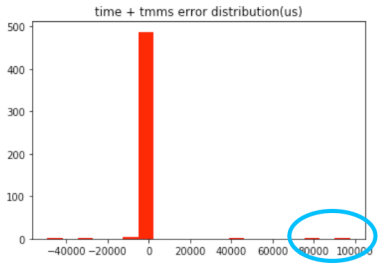
Figure 39 : DeviceTimeAns payload format

The time provided by the network MUST have a worst case accuracy of +/-100mSec.

这说明在**最坏情况下**，网关的时间精度满足协议需要，但对于pingslot 30ms的时间精度来说就难以满足要求，因此claas b实现难度很大。所以需要观察一下误差分布。

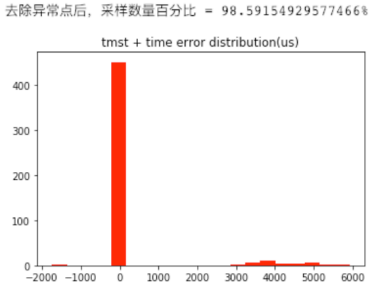


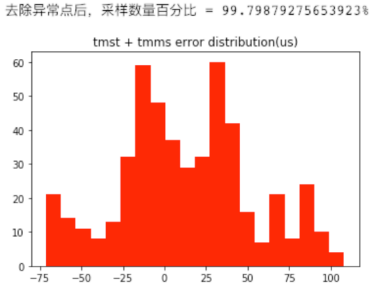


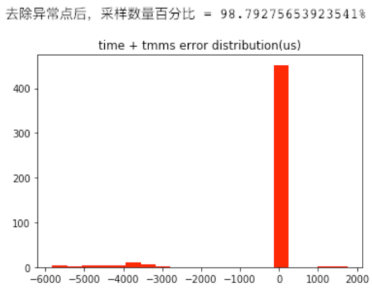


在497个采样点中，大部分时候，时间误差都在0左右，存在很少部分的“异常点”拖累了误差平均值，并且让误差分布图不够直观，因此，去掉这些点。

异常点判定准则: +-30ms







结论

至少有98.59%的采样点，时间精度误差在 $\pm 30ms$ 内

至少有91.14%的采样点，时间精度误差在 $\pm 3ms$内

至少有90.74%的采样点，时间精度误差在 $\pm 0.3ms$内

至少有55.73%的采样点，时间精度误差在 $\pm 0.03ms$内

因此，我们可以接受，网关的时间同步精度在绝大部分情况下，满足class B模式的需要。