Timekeeping for the masses

How to keep your stuff on-time

Rudi van Drunen (NLUUG) 21-05-2024

Agenda

- Intro
- Timekeeping methods
  - NTP
  - NTS
  - PtP
- Details on PtP
- Demo
- Questions
- Lunch Time (and General Assembly)


BACK
TO THE FUTURE

Me

Rudi van Drunen

- Network Engineer by day
  - LVNL
- Hardware hacker and Time Geek by night
  - GPS synced
  - DCF synced
  - LORAN(2) synced (in the works, using SDR)
  - Rubidium time standard @home
  - Earlier works: Stratum0 op een RasPi (NLUUG) 19-11-2015
    https://www.youtube.com/watch?v=cQlylYtWxr8

- (Likes the concept of the flux capacitor, but does not yet understand it)

Dreaming of

the real timenuts….

- https://www.wired.com/2008/01/gallery-time-hackers/

Why

Intro

- synchronization
  - monitoring
  - security
  - telecom
    - E1
    - Mobile
  - broadcast
    - Audio / video sync

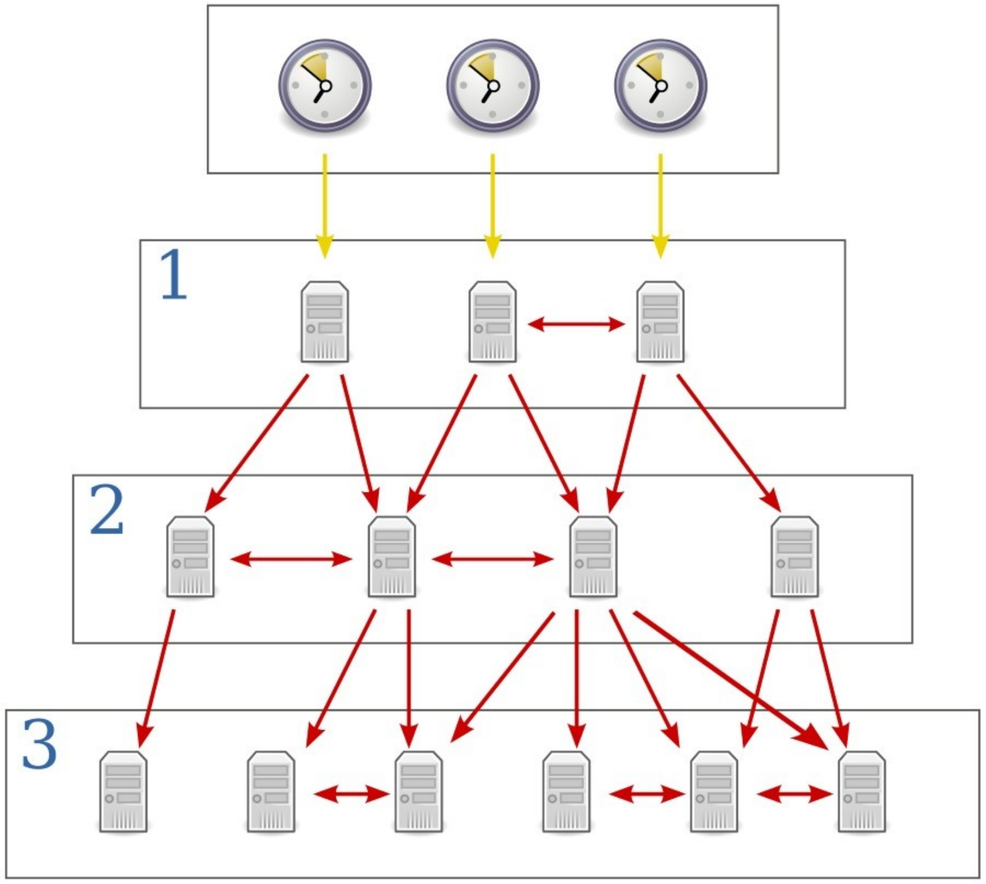
Time

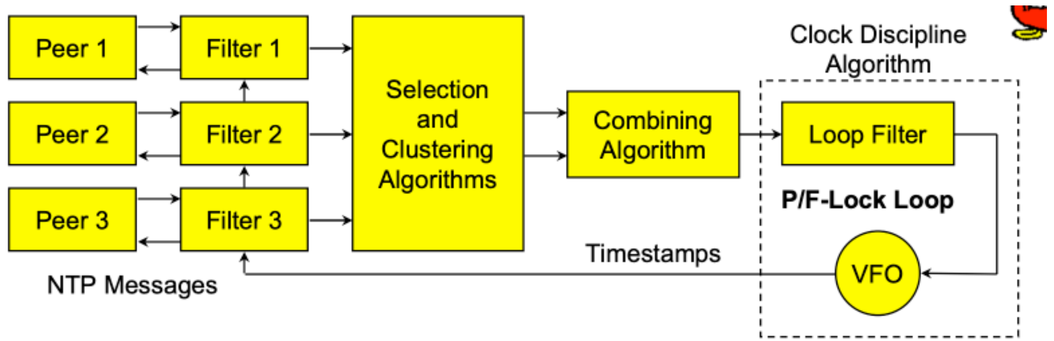
- Absolute time
  - Get from GPS / DCF
- Relative time
  - TCXO
  - Refclock
  - Time standard

How

- NTP (network time Protocol)
  - microseconds
  - NTP + NTS (TLS : authentication)
- NTPsec
  - rewrite optimizing code + more secure
- PtP
  - sub-microsecond

NTP architecture

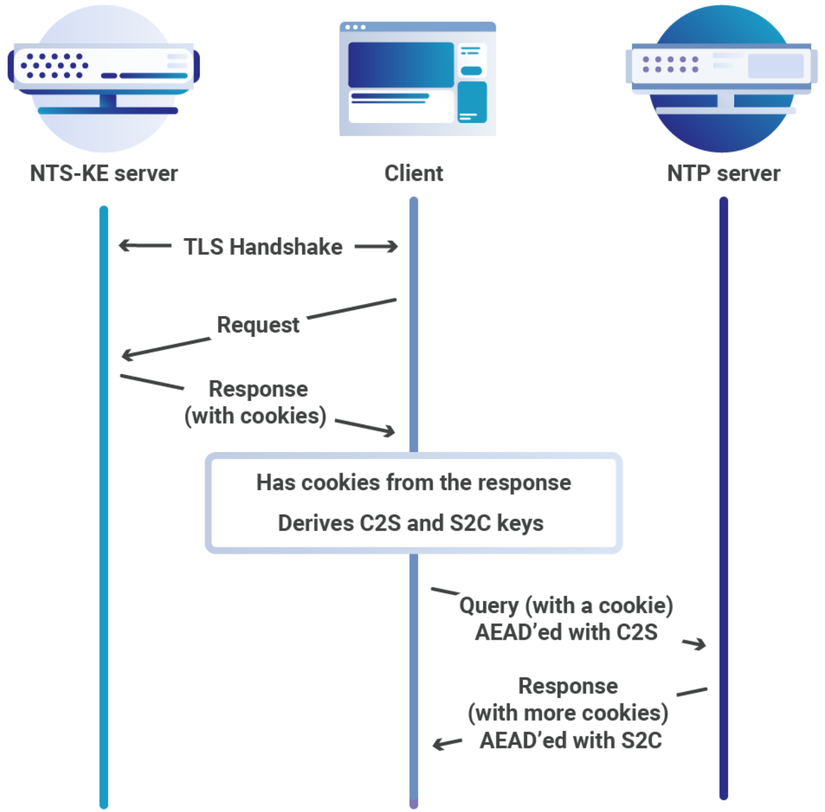




NTS

NTP extension

- rfc 8915
  - needs keyserver
  - needs server support
  - encrypted response
  - mitm mitigation



- https://www.netnod.se/sites/default/files/2021-01/Netnod_NTS_Whitepaper_2020.pdf

CLOUDFLARE

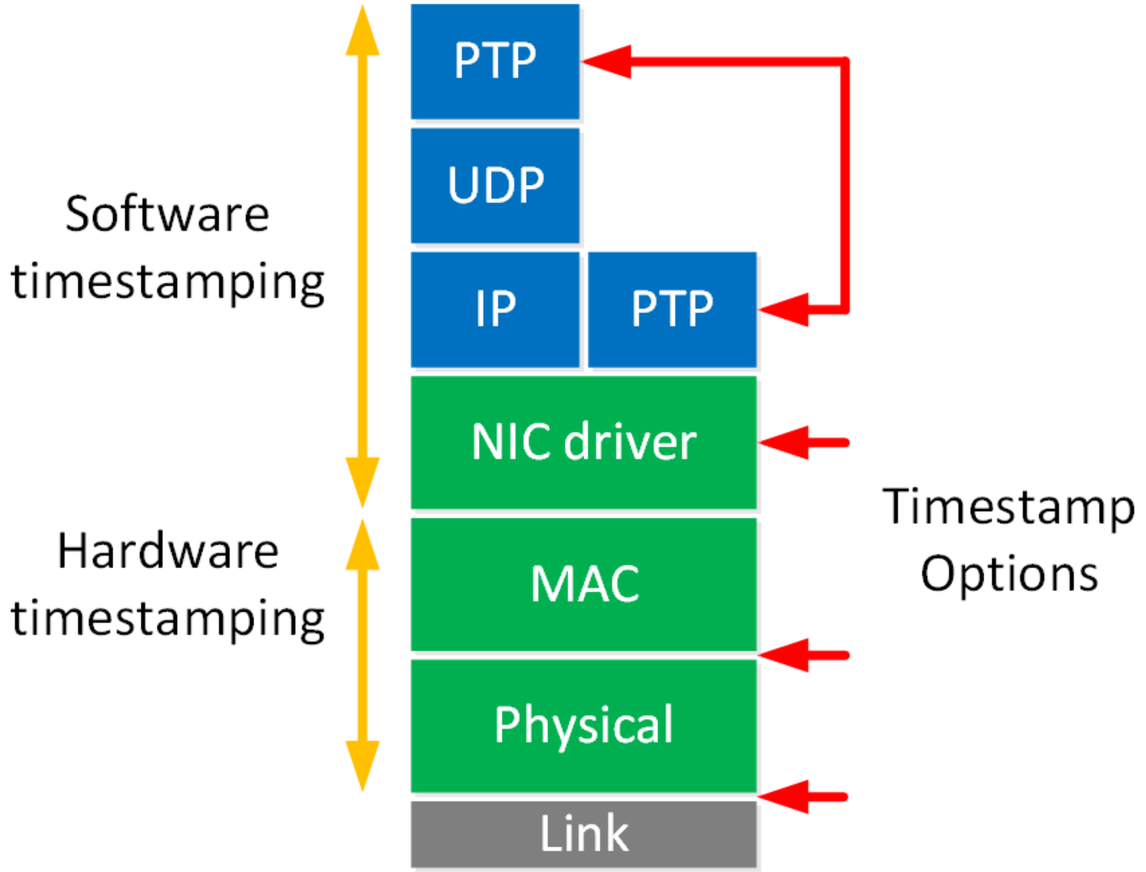
PtP

Precision Time Protocol

- https://www.ieee802.org/1/files/public/docs2009/as-mjt-clause-7-d0-1009.pdf
- IEEE1588 PtP
  - Hardware support
  - Layer 2 en Layer 3/4
  - difference in ns
  - timestamped messages delay measurement
  - 2p2 and e2e

Timestamping

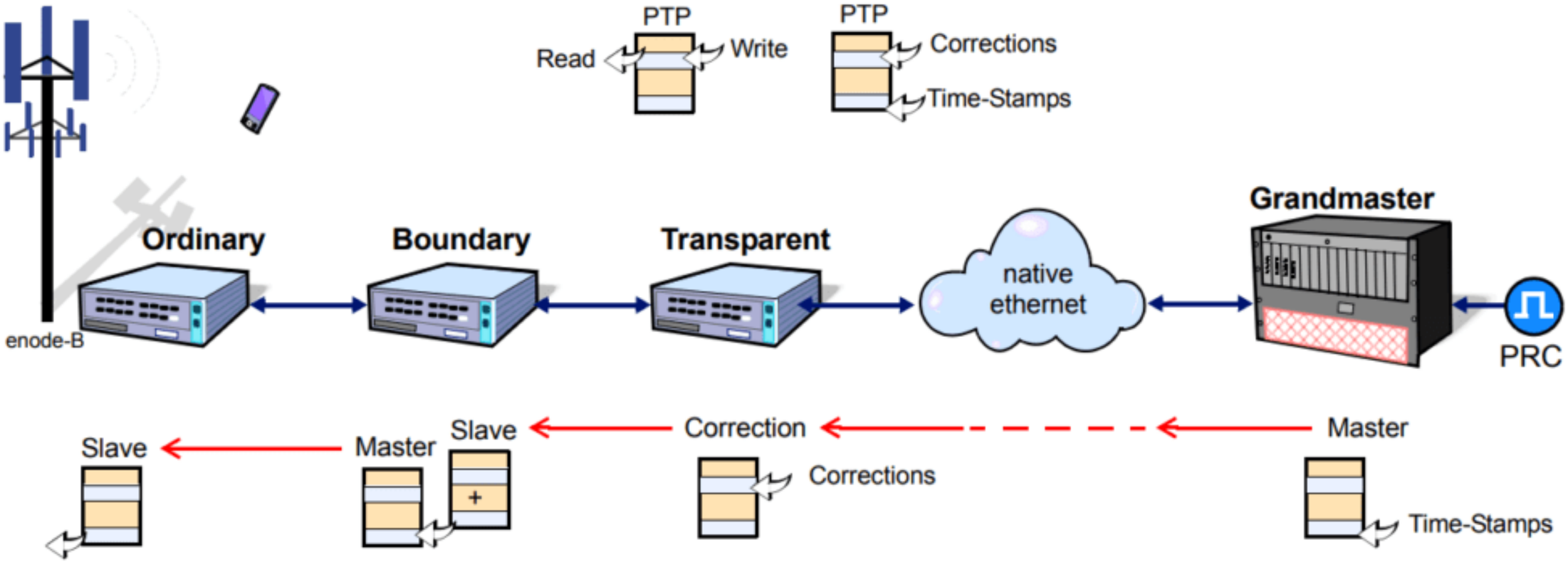
- The lower the better (accuracy)



gPtP

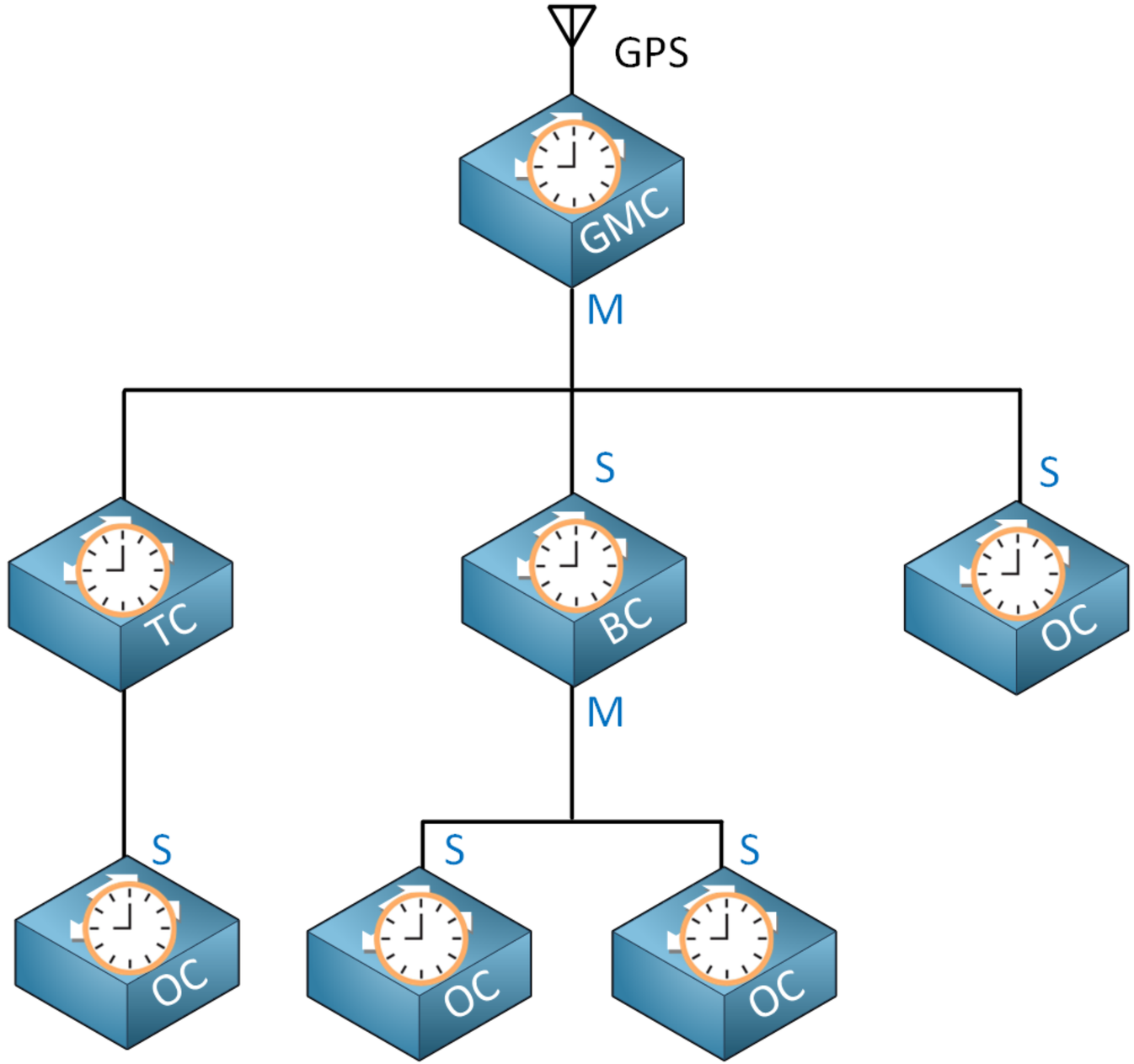
- gPtP (enhancements) https://www.youtube.com/watch?v=3vsO4Ndv-9M
- IEEE 802.1AS
  - no end-to-end mode only time-aware nodes
  - Layer 2 only
  - larger networks
  - time sensitive bridges, multiple domains
  - fault tolerant
  - sub microsecond
  - ethertype 88f7

PtP setup

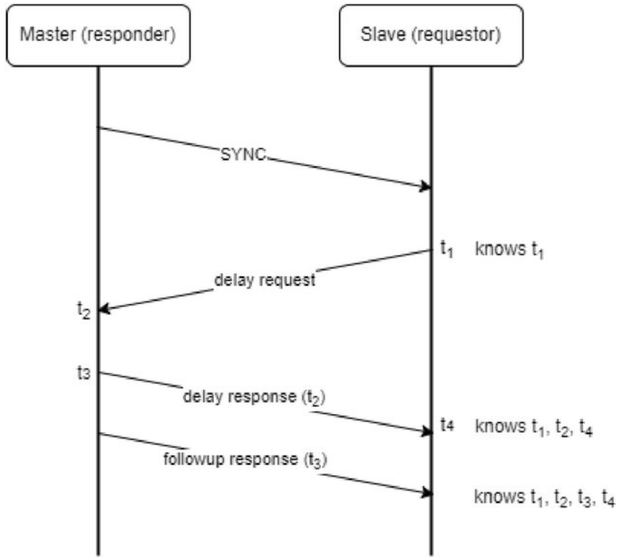

PTP
Read
Write
PTP
Corrections
Time-Stamps
Ordinary
Boundary
Transparent
native
ethernet
Grandmaster
enode-B
PRC
Slave
Master
Slave
+
Correction
Corrections
Master
Time-Stamps

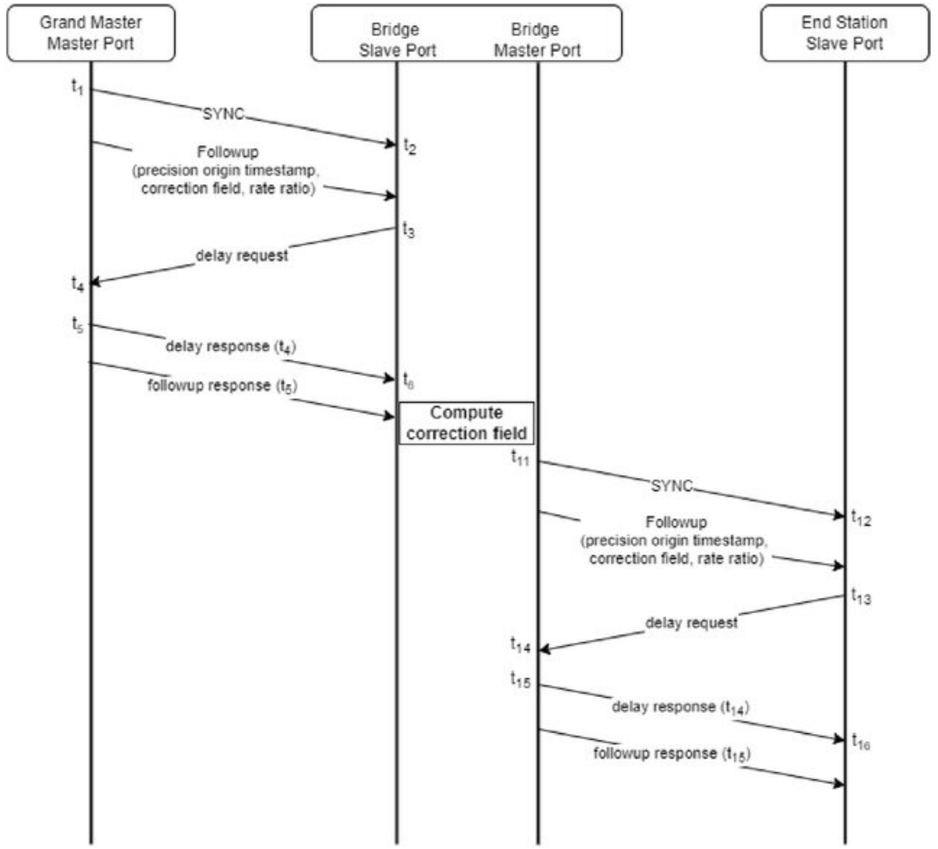
PtP

- GrandMaster
- Boundary
- Transparent
  - E2E transparent
  - P2P transparent
- Ordinary



PtP vs gPtP


Master (responder)
Slave (requestor)
SYNC
t1 knows t1
delay request
t2
t3
delay response (t2)
t4 knows t1, t2, t4
followup response (t3)
knows t1, t2, t3, t4


Grand Master
Master Port
Bridge
Slave Port
Bridge
Master Port
End Station
Slave Port
t1
SYNC
t2
Followup
(precision origin timestamp,
correction field, rate ratio)
t3
delay request
t4
t5
delay response (t4)
t6
followup response (t5)
Compute
correction field
t11
SYNC
t12
Followup
(precision origin timestamp,
correction field, rate ratio)
t13
delay request
t14
t15
delay response (t14)
t16
followup response (t15)

PtP

messages

- UDP (319,320)
- Ethernet (Frame 0x88f7)
- Event
  - SYNC
  - Follow_Up
  - Delay_Req
  - Delay Resp
- General
  - Announce
  - Management
  - Signal

Demo

Wireshark Capture

- Sorry: canned thing (no spanport and no time (pun intended))
  - PtP Layer 2
  - PtP layer 3/4
  - multicast (so: udp)
    - `01:80:C2:00:00:0E / 224.0.1.129` all except peer__delay
    - `01:1B:19:00:00:00 / 224.0.0.107` peer_delay / election
  - unicast
    - delay_req (319) delay_resp (320)

Linux PtP

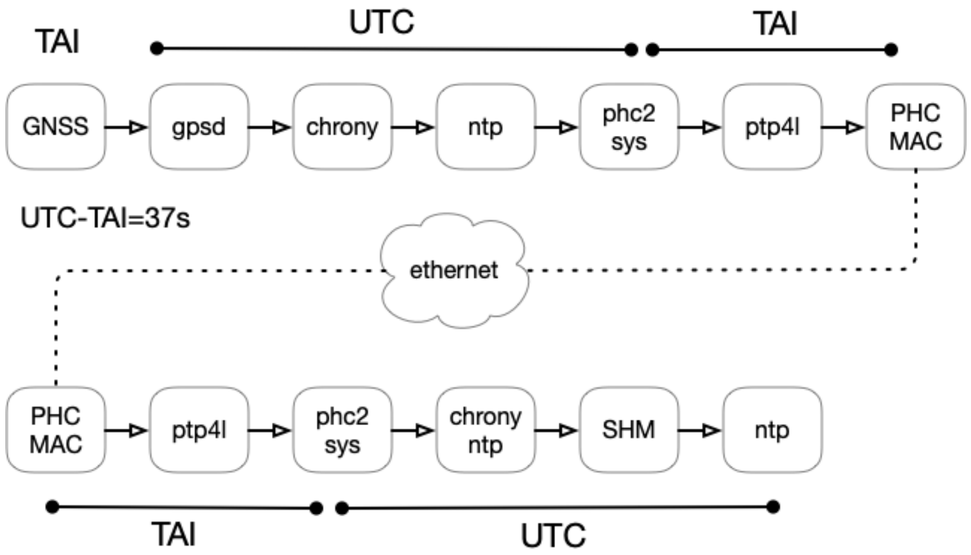
linuxptp.sourceforge.net

- Supports hardware and software time stamping via the Linux `SO_TIMESTAMPING` socket option.
- Supports the Linux PTP Hardware Clock (PHC) subsystem using the `clock_gettime` family of calls, including `clock_adjtimex` system call.
- Implements Boundary Clock (BC), Ordinary Clock (OC) and Transparent Clock (TC).
- Transport over UDP/IPv4, UDP/IPv6, and raw Ethernet (Layer 2).
- Supports IEEE 802.1AS-2011 (gPtP) in the role of end station.
- Modular design allowing painless addition of new transports and clock servos.
- Implements unicast operation.
- Supports a number of profiles, including:
  - The automotive profile
  - The default 1588 profile.
  - The enterprise profile.
  - The telecom profiles G.8265.1, G.8275.1, and G.8275.2.
- Supports bonded, IPoIB, and vlan interfaces.

Links

combining ntp - ptp

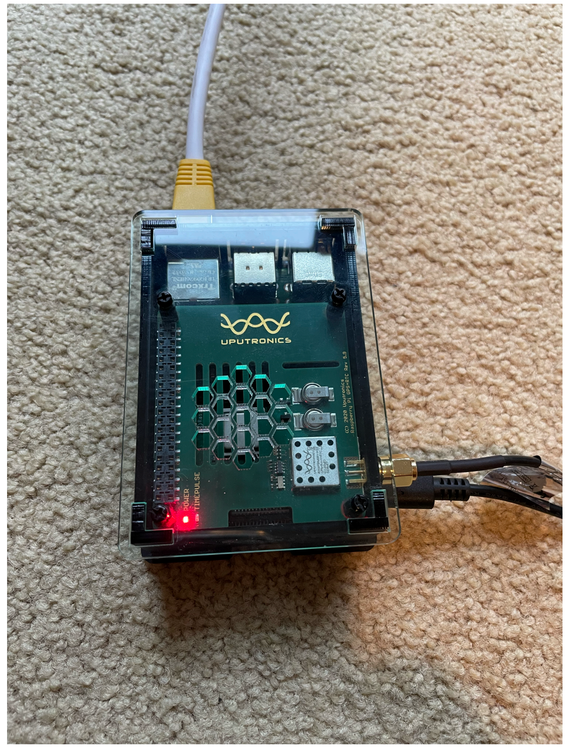
- https://forums.raspberrypi.com/viewtopic.php?t=358275
- https://github.com/by/ptp4RaspberryPi/blob/main/Setup.md
  - config ptp on pi
  - sync system through ntp to ptp to ntp



Demo time

this is remote (sorry) [my tiny time-lab @ home]

- GPS clock (NTS)
  - chrony
- PtP setup
  - A:
    - NTP chrony synced
    - Grandmaster
  - B:
    - Ordinary clock



Hardware support

PtP

```
rudi@raspberrypi-ptp-B:~$ ethtool -T eth0

Time stamping parameters for eth0:
Capabilities:
     hardware-transmit
     software-transmit
     hardware-receive
     software-receive
     software-system-clock
     hardware-raw-clock
PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes:
     off
     on
     onestep-sync
Hardware Receive Filter Modes:
     none
```

Software

ptp4l

```
usage: ptp4l [options]

 Delay Mechanism

 -A        Auto, starting with E2E
 -E        E2E, delay request-response (default)
 -P        P2P, peer delay mechanism

 Network Transport

 -2        IEEE 802.3
 -4        UDP IPV4 (default)
 -6        UDP IPV6

 Time Stamping

 -H        HARDWARE (default)
 -S        SOFTWARE
 -L        LEGACY HW
```

```
Other Options


-f [file] read configuration from 'file'
-i [dev]  interface device to use, for example 'eth0'
          (may be specified multiple times)
-p [dev]  Clock device to use, default auto
          (ignored for SOFTWARE/LEGACY HW time stamping)
-s        slave only mode (overrides configuration file)
-l [num]  set the logging level to 'num'
-m        print messages to stdout
-q        do not print messages to the syslog
-v        prints the software version and exits
-h        prints this message and exits
```

Software

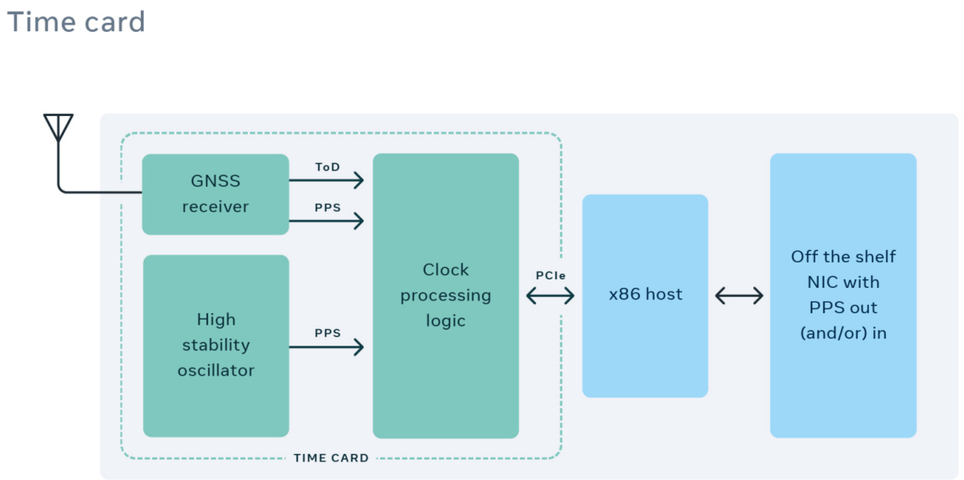
phc2sys

```
usage: phc2sys [options]



 automatic configuration:
 -a              turn on autoconfiguration
 -r              synchronize system (realtime) clock
                 repeat -r to consider it also as a time source
 manual configuration:
 -c [dev|name]   slave clock (CLOCK_REALTIME)
 -d [dev]        master PPS device
 -s [dev|name]   master clock
 -O [offset]     slave-master time offset (0)
 -w              wait for ptp4l
```

```
common options:
 -f [file]      configuration file
 -E [pi|linreg] clock servo (pi)
 -P [kp]        proportional constant (0.7)
 -I [ki]        integration constant (0.3)
 -S [step]      step threshold (disabled)
 -F [step]      step threshold only on start (0.00002)
 -R [rate]      slave clock update rate in HZ (1.0)
 -N [num]       number of master clock readings per update (5)
 -L [limit]     sanity frequency limit in ppb (200000000)
 -M [num]       NTP SHM segment number (0)
 -u [num]       number of clock updates in summary stats (0)
 -n [num]       domain number (0)
 -x             apply leap seconds by servo instead of kernel
 -z [path]      server address for UDS (/var/run/ptp4l)
 -l [num]       set the logging level to 'num' (6)
 -t [tag]       add tag to log messages
 -m             print messages to stdout
 -q             do not print messages to the syslog
 -v             prints the software version and exits
 -h             prints this message and exi
```

Next

- PtP Rpi PPS output
  - RPi5 MAC is capable but needs HW mods
  - CM4 MAC (baseboard)
- Add on Hardware MAC with PtP and PPS



- https://github.com/opencomputeproject/Time-Appliance-Project
- https://www.opencompute.org/w/index.php?title=Time_Appliances_Project

Questions ?

rudi@freedom.nl