INTRODUCTION TO TIME SYNCHRONIZATION

PT. 2 HOW DO YOU TURN THIS ON?

Maciek Machnikowski

WHAT’S IN IT FOR ME?

Synchronize timers to sub-microsecond accuracy

Administration-free operation

Synchronize events across multiple platforms

Better monitoring and telemetry

Better distributed databases

Better cryptography

Measure latency between nodes

Your hardware supports it already

AGENDA

- Previously on NetDev 0x15
- Time in Linux
- Introducing linuxptp
- Putting it all together
- Tips&Tricks

IN PREVIOUS EPISODE…

NetDev 0x15

PROTOCOL INTRODUCTION

PTP – Precision Time Protocol

IEEE 1588 standard

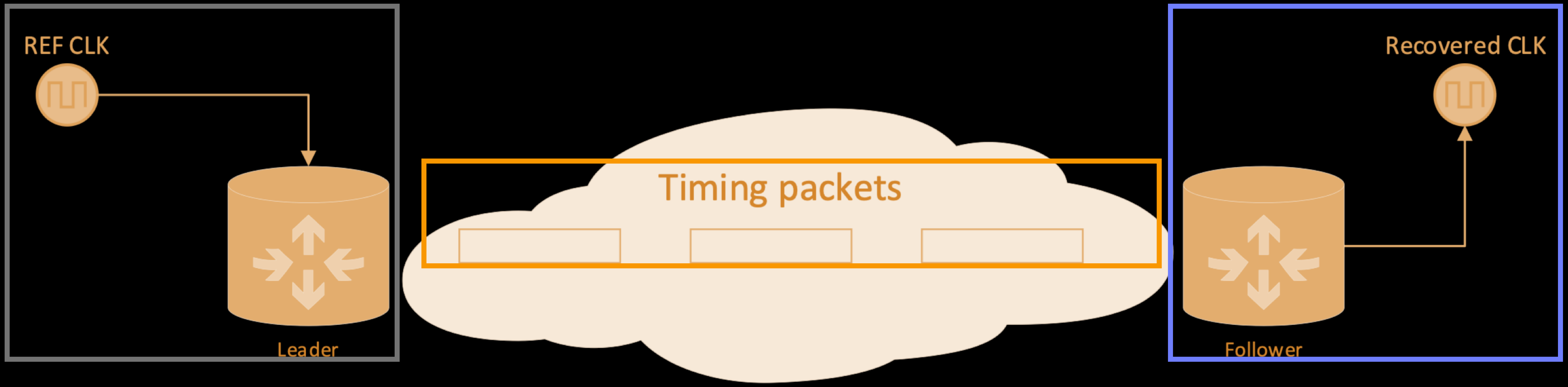
Synchronize timers to sub-microsecond accuracy

Hierarchical M-S architecture for clock distribution

Administration-free operation

For both high-end devices and low-end devices

FREQUENCY AND TIME TRANSFER

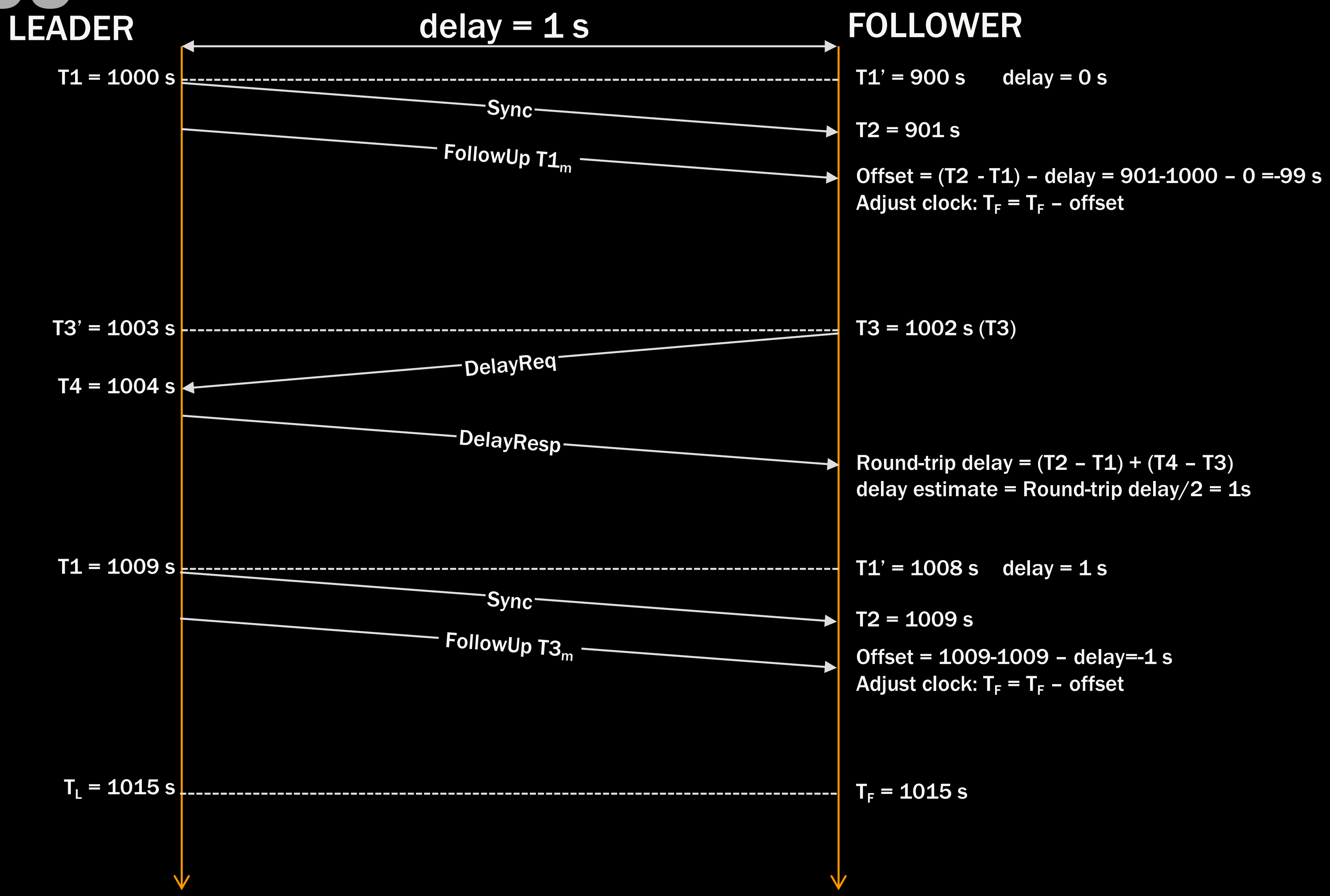


Generation: create packets from physical reference source

Transfer: packet transmission over packet network

Recovery: regenerate the physical frequency from the received packets

IEEE 1588



LEADER

delay = 1 s

FOLLOWER

T1 = 1000 s

T1’ = 900 s delay = 0 s

Sync

T2 = 901 s

FollowUp $T1_m$

Offset = (T2 - T1) – delay = 901-1000 – 0 =-99 s

Adjust clock: T_F = T_F – offset

T3’ = 1003 s

T3 = 1002 s (T3)

DelayReq

T4 = 1004 s

DelayResp

Round-trip delay = (T2 – T1) + (T4 – T3)

delay estimate = Round-trip delay/2 = 1s

T1 = 1009 s

T1’ = 1008 s delay = 1 s

Sync

T2 = 1009 s

FollowUp $T3_m$

Offset = 1009-1009 – delay=-1 s

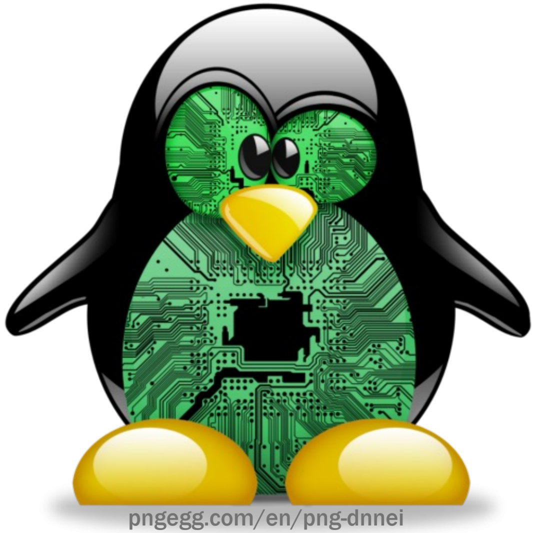
Adjust clock: T_F = T_F – offset

T_L = 1015 s

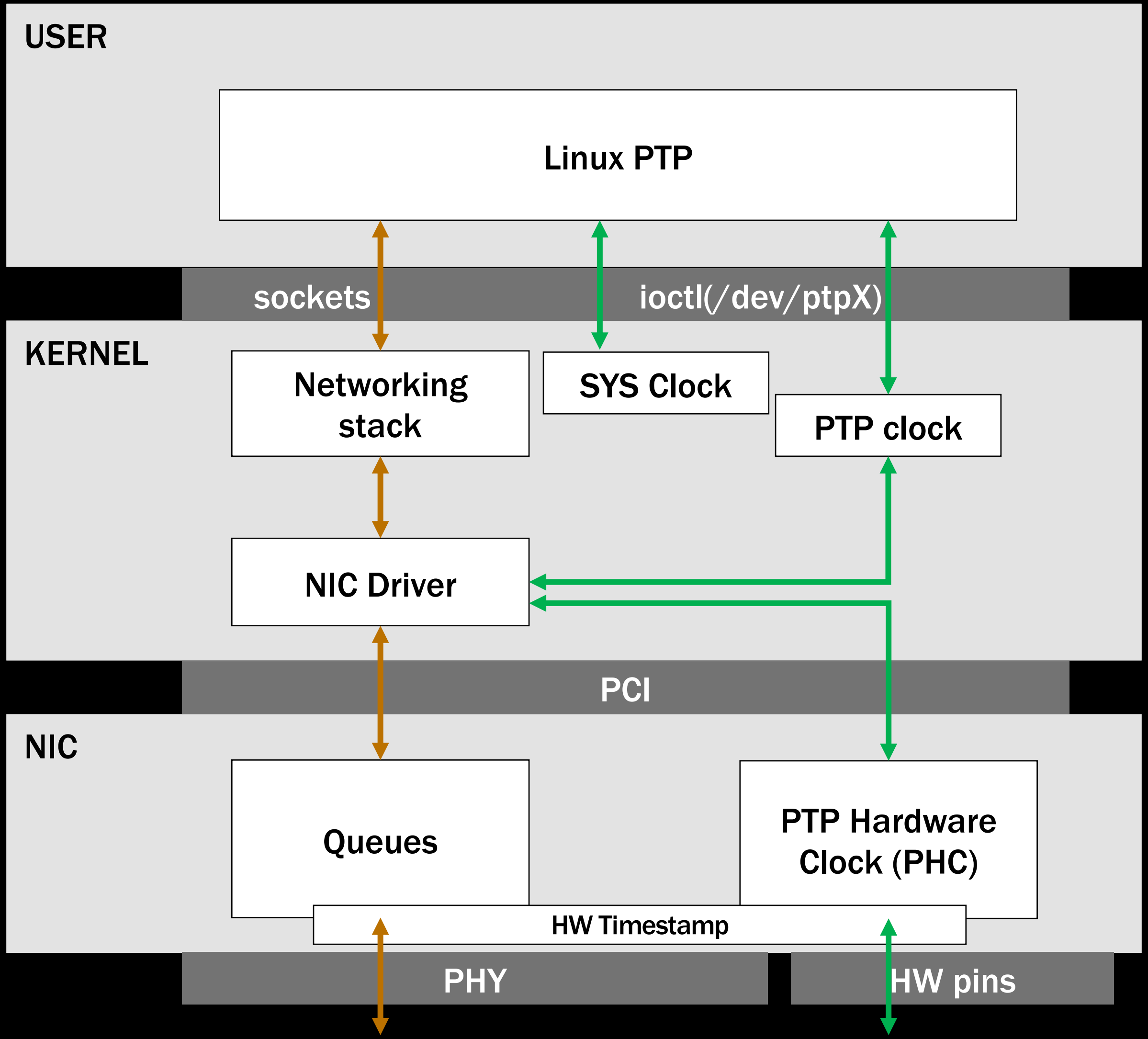
T_F = 1015 s

TIME IN LINUX

How a sausage is made



pngegg.com/en/png-dnnei


USER
Linux PTP
sockets
ioctl(/dev/ptpX)
KERNEL
Networking stack
SYS Clock
PTP clock
NIC Driver
PCI
NIC
Queues
PTP Hardware Clock (PHC)
HW Timestamp
PHY
HW pins

POSIX CLOCK API

- PHCs are represented by POSIX clocks (/dev/ptpX)
- clock_adjtime: Adjust the clock
  - adjfreq (adjfine)
  - adjtime
- clock_gettime: Read the current time
- clock_settime: Set the current time
- ioctl: Optional IOCTL methods

ETHTOOL

```
maciek@cm4-4G:~ $ ethtool -T eth0
Time stamping parameters for eth0:
Capabilities:
        hardware-transmit
        hardware-receive
        hardware-raw-clock
PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes:
        off
        on
        onestep-sync
        onestep-p2p
Hardware Receive Filter Modes:
        none
        ptpv2-event
```

PTP SYSFS INTERFACE

```
# echo <channel> <enable> > extts_enable
```

channel - channel index

enable – set to 1 to enable or 0 to disable

```
# echo <channel> <st_s> <st_ns> <per_s> <per_ns> > period
```

channel - channel index

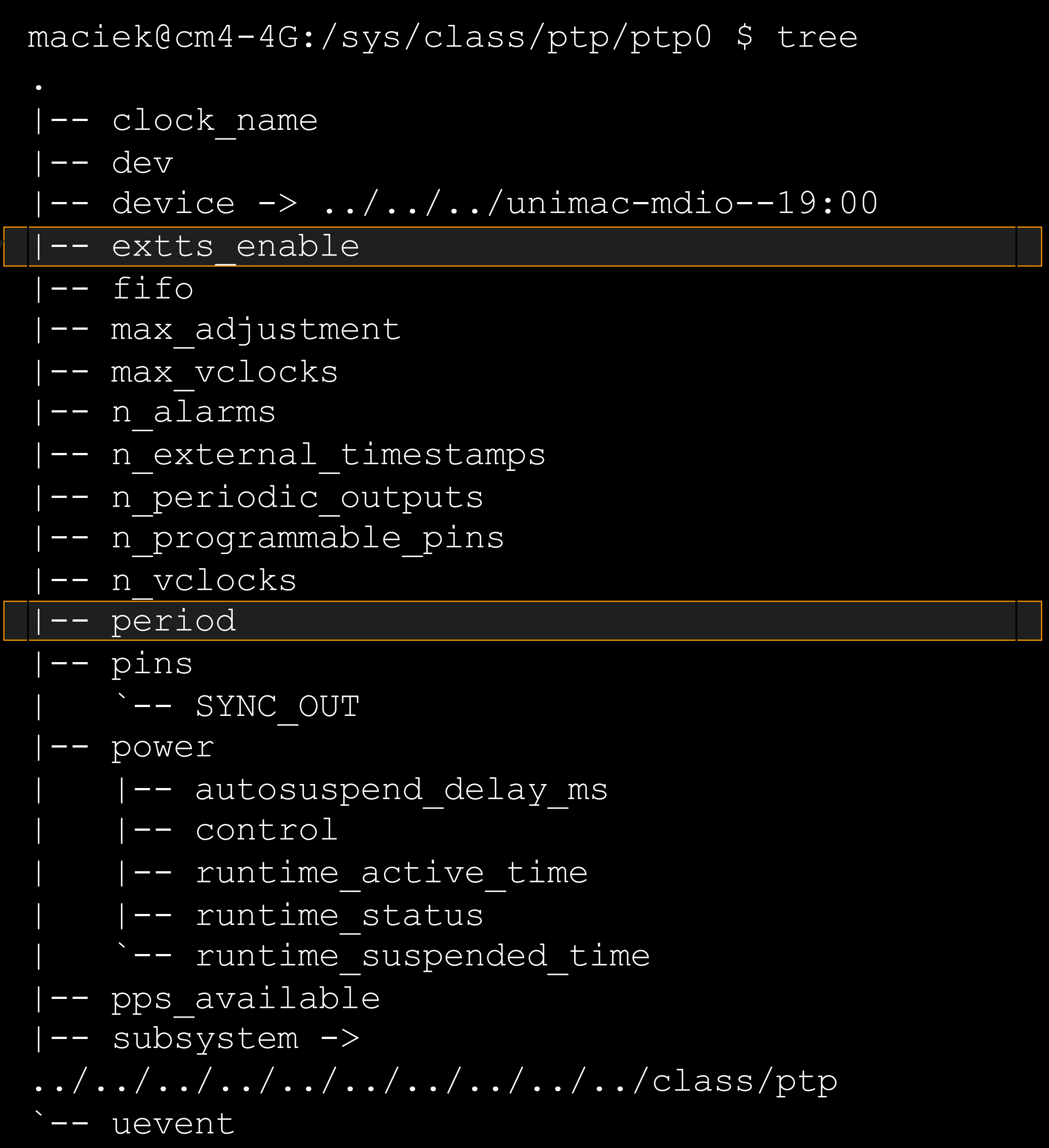
st_s - start time seconds

st_ns - start time nanoseconds

per_s - period seconds

per_ns - period nanoseconds

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-ptp



```
maciek@cm4-4G:/sys/class/ptp/ptp0 $ tree
.
|-- clock_name
|-- dev
|-- device -> ../../../unimac-mdio--19:00
|-- extts_enable
|-- fifo
|-- max_adjustment
|-- max_vclocks
|-- n_alarms
|-- n_external_timestamps
|-- n_periodic_outputs
|-- n_programmable_pins
|-- n_vclocks
|-- period
|-- pins
|   `-- SYNC_OUT
|-- power
|   |-- autosuspend_delay_ms
|   |-- control
|   |-- runtime_active_time
|   |-- runtime_status
|   `-- runtime_suspended_time
|-- pps_available
|-- subsystem ->
../../../../../../../../../../class/ptp
`-- uevent
```

PTP PINS INTERFACE

```
# echo <function> <channel> > pins/SYNC_OUT
```

function – desired pin function:

- 0 – none
- 1 – ext_ts
- 2 – perout

channel - channel index

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-ptp

```
maciek@cm4-4G:/sys/class/ptp/ptp0 $ tree
.
|-- clock_name
|-- dev
|-- device -> ../../../unimac-mdio--19:00
|-- extts_enable
|-- fifo
|-- max_adjustment
|-- max_vclocks
|-- n_alarms
|-- n_external_timestamps
|-- n_periodic_outputs
|-- n_programmable_pins
|-- n_vclocks
|-- period
|-- pins
|   `-- SYNC_OUT
|-- power
|   |-- autosuspend_delay_ms
|   |-- control
|   |-- runtime_active_time
|   |-- runtime_status
|   `-- runtime_suspended_time
|-- pps_available
|-- subsystem ->
../../../../../../../../../../class/ptp
`-- uevent
```

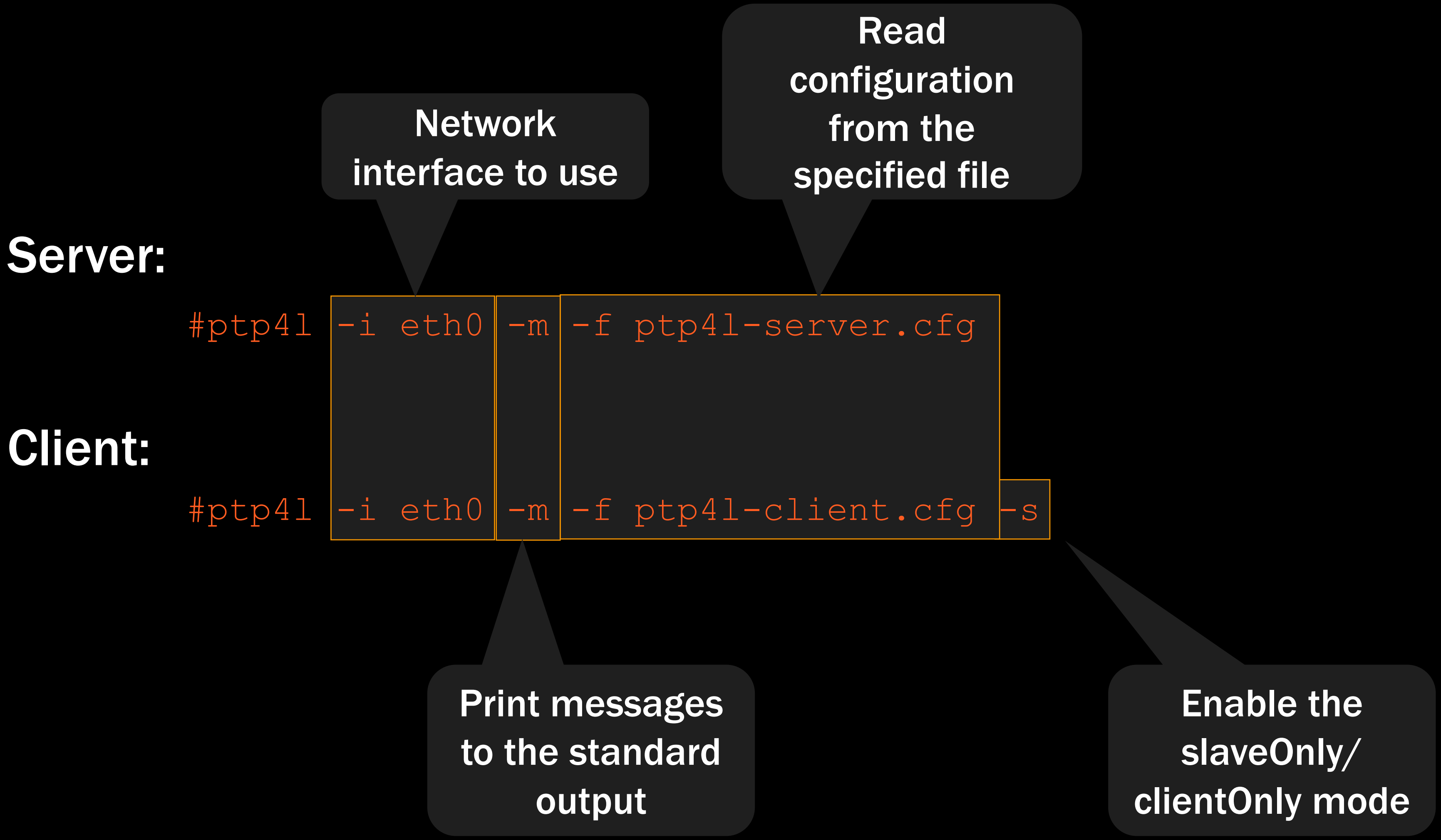
INTRODUCING LINUXPTP

1588 swiss-army knife

LINUX PTP PROJECT

- ptp4l
  - Synchronize 2 PHCs using the PTP protocol (IEEE 1588)
- ts2phc
  - Synchronize PHCs to external time stamp signal (1PPS signals).
- phc2sys
  - synchronize two (or more) POSIX clocks
  - can synchronize PHC and SYS
- pmc
  - PTP management client
  - Can obtain additional information from a running ptp4l and reconfigure it
- timemaster
  - synchronize the system clock to NTP and PTP time sources

STARTING PTP4L



Server:

```
#ptp4l -i eth0 -m -f ptp4l-server.cfg
```

Client:

```
#ptp4l -i eth0 -m -f ptp4l-client.cfg -s
```

-i eth0: Network interface to use

-f: Read configuration from the specified file

-m: Print messages to the standard output

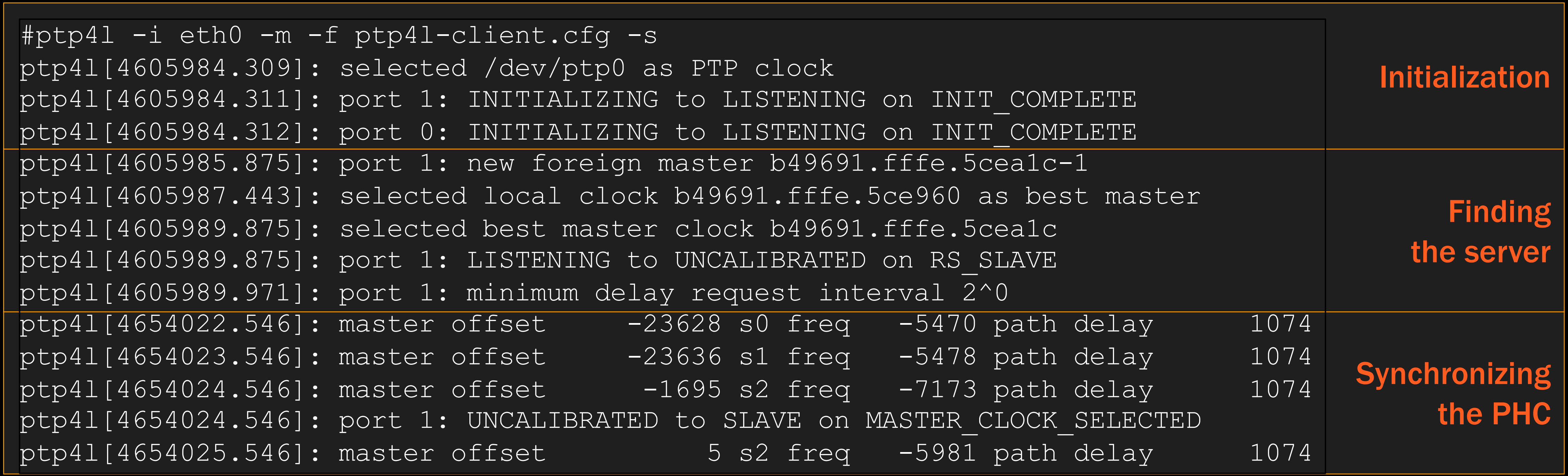
-s: Enable the slaveOnly/ clientOnly mode

PTP4L - SERVER

```
#ptp4l -i eth0 -m -f ptp4l-server.cfg
ptp4l[212068.052]: selected /dev/ptp0 as PTP clock
ptp4l[212068.053]: port 1: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[212068.054]: port 0: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[212075.790]: port 1: LISTENING to MASTER on ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES
ptp4l[212075.790]: selected local clock b49691.fffe.5cea1c as best master
ptp4l[212075.790]: assuming the grand master role
```

We're the grandmaster

PTP4L - CLIENT



```
#ptp4l -i eth0 -m -f ptp4l-client.cfg -s
ptp4l[4605984.309]: selected /dev/ptp0 as PTP clock
ptp4l[4605984.311]: port 1: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[4605984.312]: port 0: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[4605985.875]: port 1: new foreign master b49691.fffe.5cea1c-1
ptp4l[4605987.443]: selected local clock b49691.fffe.5ce960 as best master
ptp4l[4605989.875]: selected best master clock b49691.fffe.5cea1c
ptp4l[4605989.875]: port 1: LISTENING to UNCALIBRATED on RS_SLAVE
ptp4l[4605989.971]: port 1: minimum delay request interval 2^0
ptp4l[4654022.546]: master offset     -23628 s0 freq   -5470 path delay      1074
ptp4l[4654023.546]: master offset     -23636 s1 freq   -5478 path delay      1074
ptp4l[4654024.546]: master offset      -1695 s2 freq   -7173 path delay      1074
ptp4l[4654024.546]: port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
ptp4l[4654025.546]: master offset          5 s2 freq   -5981 path delay      1074
```

Initialization

Finding the server

Synchronizing the PHC

PTP4L - FOLLOWER

Synchronization state:

- s0 – servo unlocked
- s1 – clock step
- s2 – servo locked

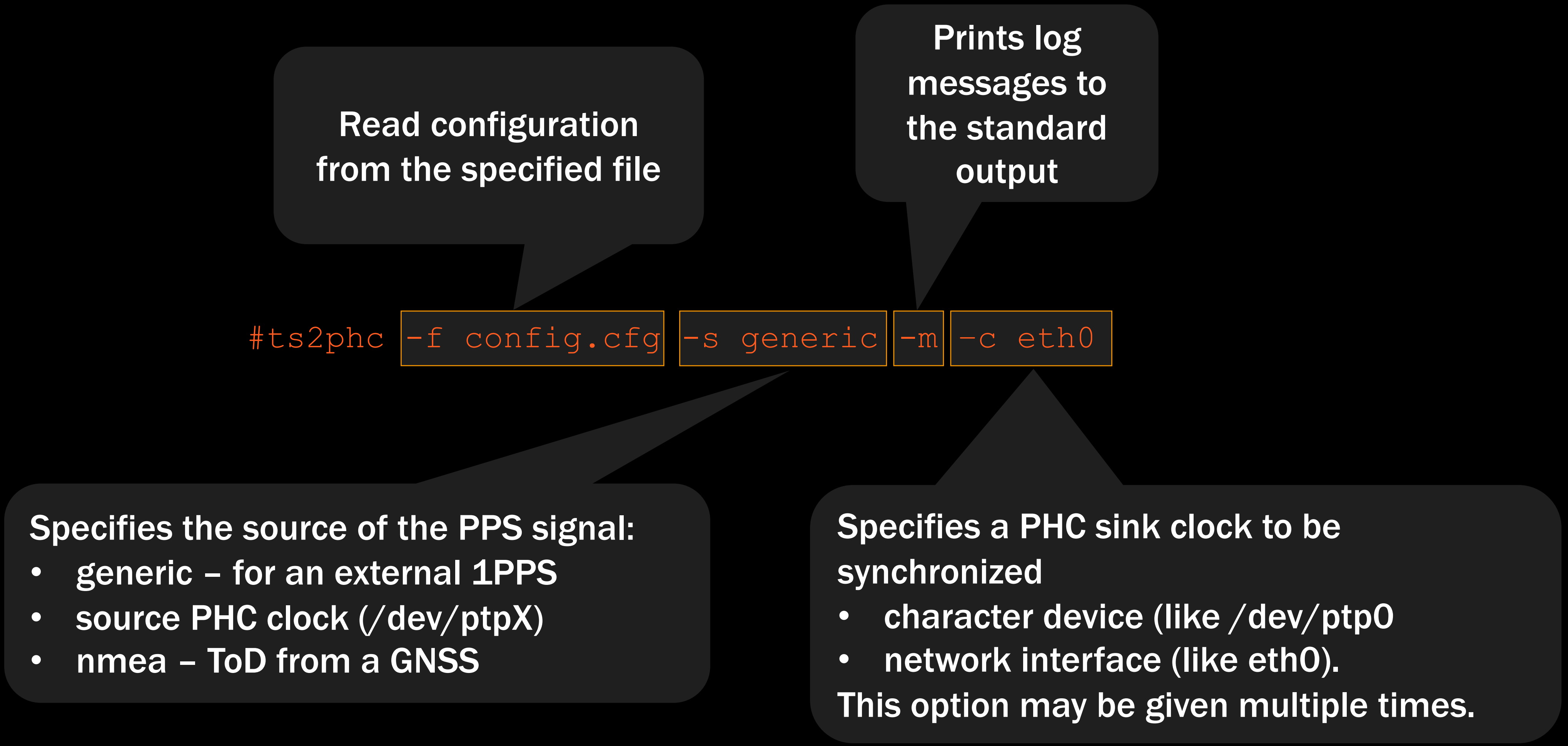
Delay of the path

```
ptp4l[466010.881]: master offset         -5 s2 freq      -5 path delay      1073
ptp4l[466011.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466012.881]: master offset          2 s2 freq      +1 path delay      1073
ptp4l[466013.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466014.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466015.881]: master offset         -1 s2 freq      -2 path delay      1073
ptp4l[466016.881]: master offset         -7 s2 freq      -8 path delay      1074
ptp4l[466017.881]: master offset          6 s2 freq      +3 path delay      1074
ptp4l[466018.881]: master offset          0 s2 freq      -1 path delay      1074
ptp4l[466019.881]: master offset          1 s2 freq      -0 path delay      1074
ptp4l[466020.881]: master offset          0 s2 freq      -1 path delay      1074
ptp4l[466021.881]: master offset          0 s2 freq      -1 path delay      1074
```

Offset to the server clock (in ns)

Frequency difference to the master clock (in ppb)

TS2PHC – COMMAND LINE



Read configuration from the specified file

Prints log messages to the standard output

```
#ts2phc -f config.cfg -s generic -m -c eth0
```

Specifies the source of the PPS signal:

- generic – for an external 1PPS
- source PHC clock (/dev/ptpX)
- nmea – ToD from a GNSS

Specifies a PHC sink clock to be synchronized

- character device (like /dev/ptp0
- network interface (like eth0).

This option may be given multiple times.

PHC2SYS – COMMAND LINE

Specify the source clock by:

- device (e.g. /dev/ptp0)
- interface (e.g. eth0)
- by name (e.g. CLOCK_REALTIME for the system clock)

Wait until ptp4l is in a synchronized state

```
#phc2sys -s eth0 -c CLOCK_REALTIME -w -m
#phc2sys -c eth0 -s CLOCK_REALTIME -O 37 -m
#phc2sys -s eth0 -O -37 -m
#phc2sys -a -r
```

Specify the offset between the source and sink

Enable auto mode and sync to the running ptp4l

Specify the sink clock by:

- device (e.g. /dev/ptp1)
- interface (e.g. eth1)
- Name

The default is CLOCK_REALTIME

Print messages to the standard output

PHC2SYS - OUTPUT

Synchronization state:

- s0 – servo unlocked
- s1 – clock step
- s2 – servo locked

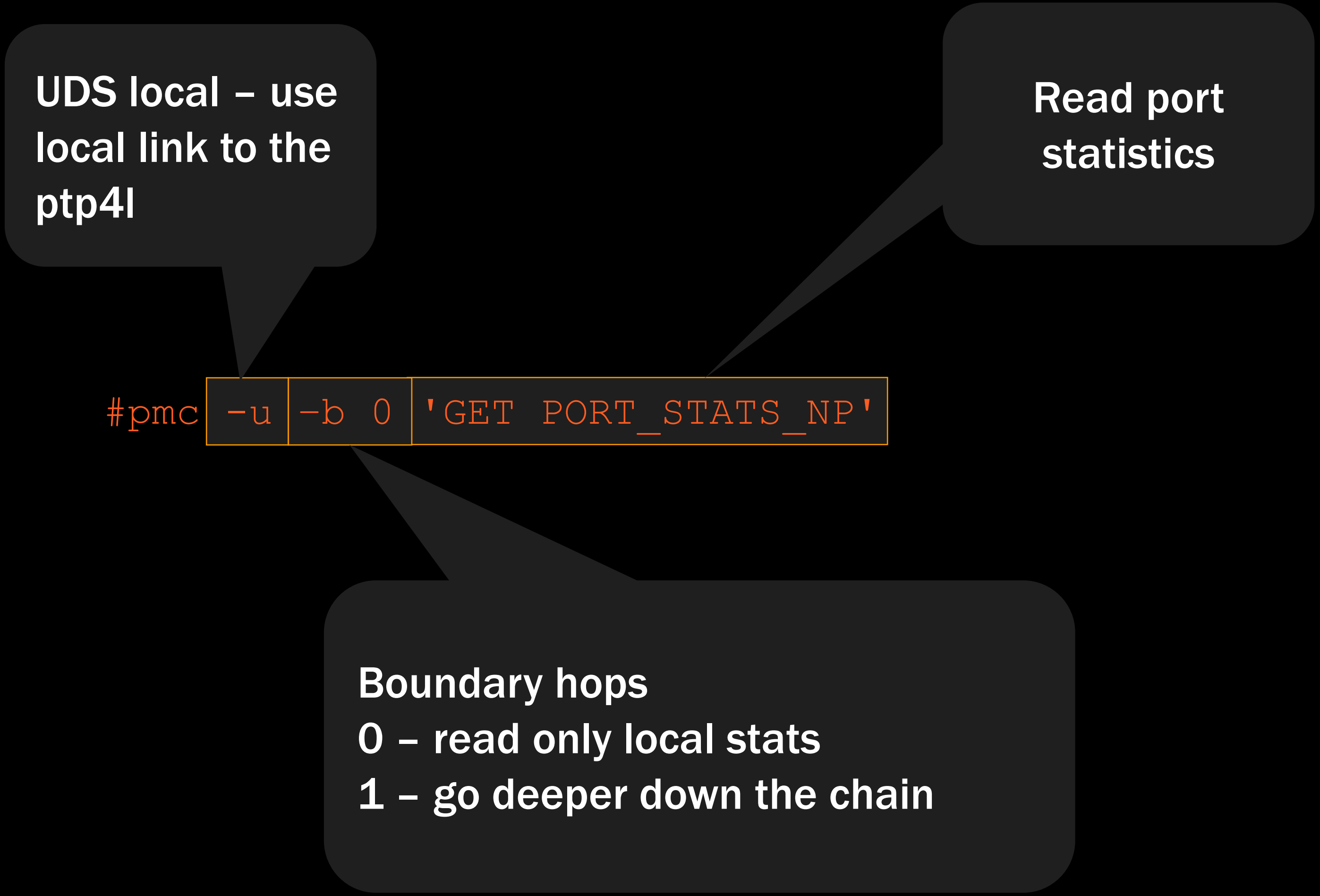
Delay of the path

```
phc2sys[4649082.619]: CLOCK_REALTIME phc offset  -4717525 s0 freq      +0 delay   1274
phc2sys[4649083.619]: CLOCK_REALTIME phc offset  -4728162 s1 freq  -10632 delay   1273
phc2sys[4649084.620]: CLOCK_REALTIME phc offset        40 s2 freq  -10592 delay   1278
phc2sys[4649085.620]: CLOCK_REALTIME phc offset         7 s2 freq  -10613 delay   1276
phc2sys[4649086.620]: CLOCK_REALTIME phc offset        -5 s2 freq  -10623 delay   1280
phc2sys[4649087.620]: CLOCK_REALTIME phc offset        13 s2 freq  -10606 delay   1277
phc2sys[4649088.621]: CLOCK_REALTIME phc offset       -13 s2 freq  -10628 delay   1278
phc2sys[4649089.621]: CLOCK_REALTIME phc offset        -5 s2 freq  -10624 delay   1290
phc2sys[4649090.621]: CLOCK_REALTIME phc offset         5 s2 freq  -10616 delay   1277
phc2sys[4649091.621]: CLOCK_REALTIME phc offset       -18 s2 freq  -10637 delay   1294
phc2sys[4649092.622]: CLOCK_REALTIME phc offset         7 s2 freq  -10618 delay   1288
phc2sys[4649093.622]: CLOCK_REALTIME phc offset        12 s2 freq  -10611 delay   1286
```

Offset to the master clock

Frequency difference to the master clock (in ppb)

PMC – COMMAND LINE



UDS local – use local link to the ptp4l

Read port statistics

```
#pmc -u -b 0 'GET PORT_STATS_NP'
```

Boundary hops
0 – read only local stats
1 – go deeper down the chain

TIMEMASTER

- uses **ptp4l** and **phc2sys**
- combines them with **chronyd** or **ntpd**
- synchronize the system clock to NTP and PTP time sources

- https://www.mankier.com/8/timemaster

PHC_CTL

- simple tool for controlling PHCs
- mostly for debug
- can set a PHC from the system time
- can compare time in a PHC to the system time

- https://www.mankier.com/8/phc_ctl

FLEXIBILITY

- Many profiles to chose from
  - P2P (802.1AS) vs E2E
- Multicast vs unicast
- L2 or L3 (IPv4 or IPv6)
- Boundary Clocks or Transparent Clocks
- PTP only vs PTP + NTP

IEEE 1588 PROFILES

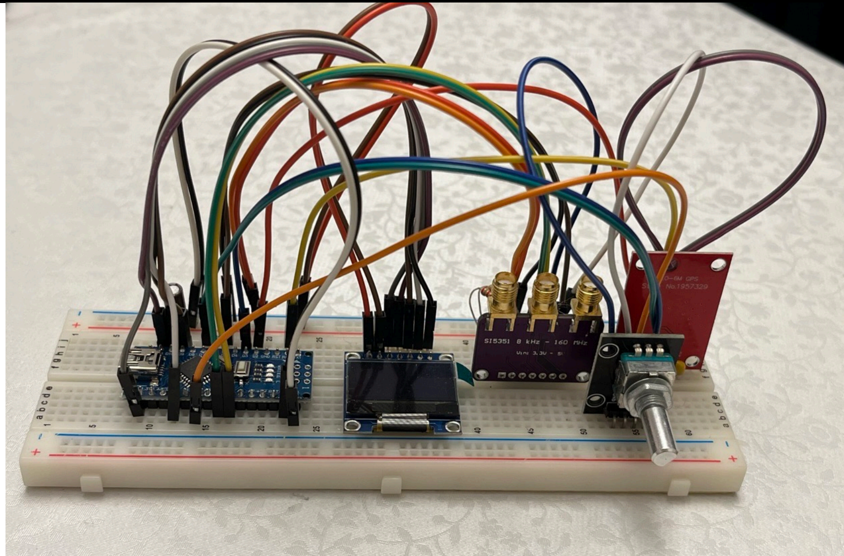
- Generic
- Telecom (G.8265.1, G.8275.1, G.8275.2)
- Industrial (PIP)
- Power (PUP)
- Audio/Video (AVB, 802.1AS-2020)
- White Rabbit
- Automotive
- Enterprise (TICTOC)

TIMESCALES

- PTP operates in PTP Timescale (which is same as TAI)
- System timer operates in UTC time
- UTC includes leap-seconds
    - Currently 37 second difference to TAI because of leap seconds
    - UTC-TAI difference encoded in leap-files
- GNSS receiver gives UTC time in NMEA messages

PUTTING IT ALL TOGETHER

How do you turn this on?



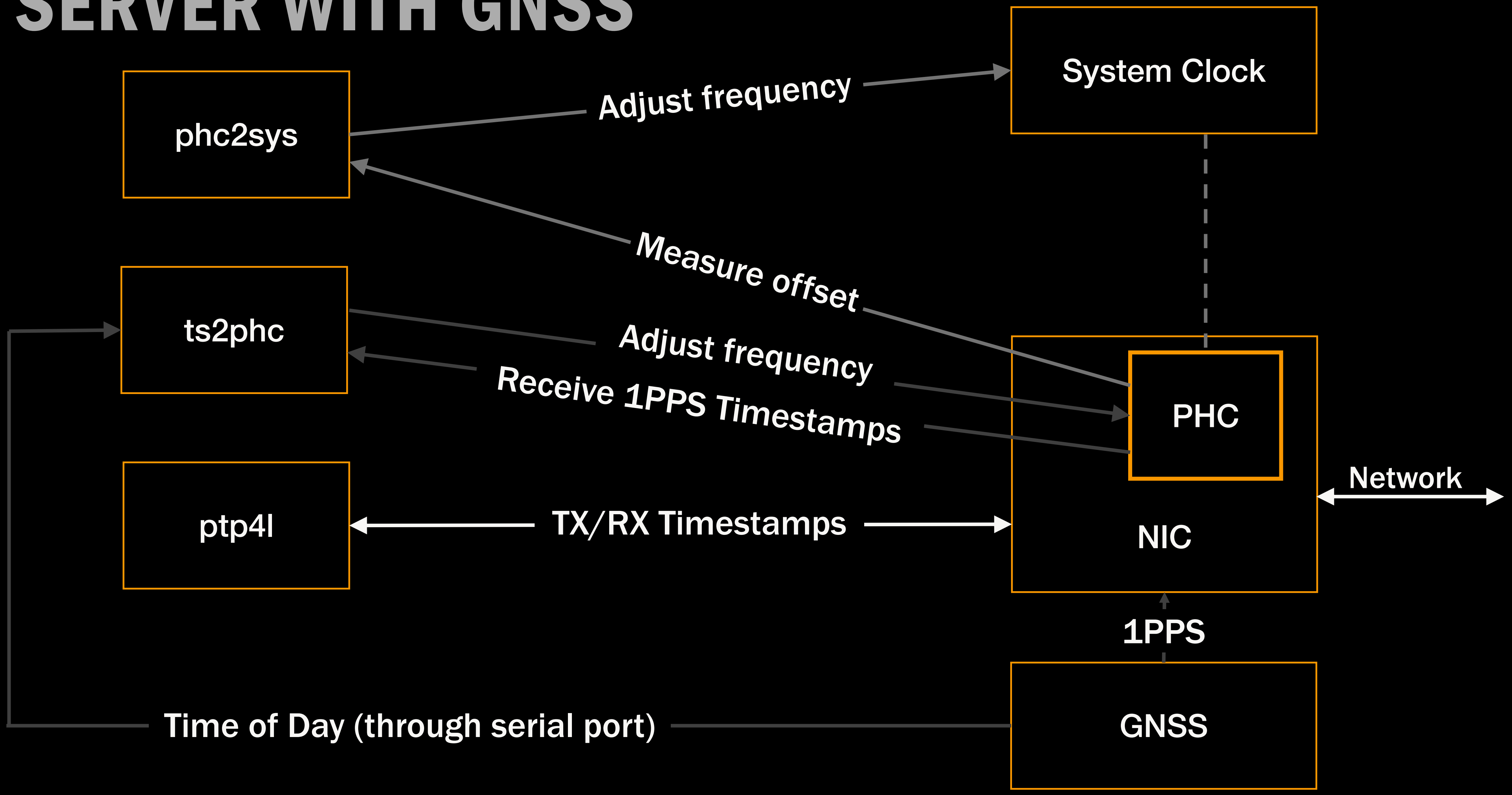
PREPARATIONS

- Locate timing pins/inputs
- Connect 1PPS from the GNSS to the 1PPS input on the NIC
- Connect UART port of the GNSS (or the USB)
- Connect GNSS antenna
- Know limitations of your HW
    - Reading time may block HW timestamps
    - Only certain way of timestamping may be available

HW REQUIREMENTS

- HW timestamps
- Exposed external PPS input
- (optionally) Exposed PPS output
- GNSS module with PPS out
- Wires to connect everything

SERVER WITH GNSS



phc2sys

Adjust frequency

System Clock

Measure offset

ts2phc

Adjust frequency

Receive 1PPS Timestamps

PHC

Network

ptp4l

TX/RX Timestamps

NIC

1PPS

Time of Day (through serial port)

GNSS

SERVER WITH GNSS

Run ts2phc to get time from the GNSS

- `ts2phc -f config.cfg -s nmea -c eth0`

-
  ```
  ts2phc -c eth0 -s nmea -m -l 7 --leapfile leap-seconds.list\
  --ts2phc.nmea_serialport /dev/ttyACM0
  ```

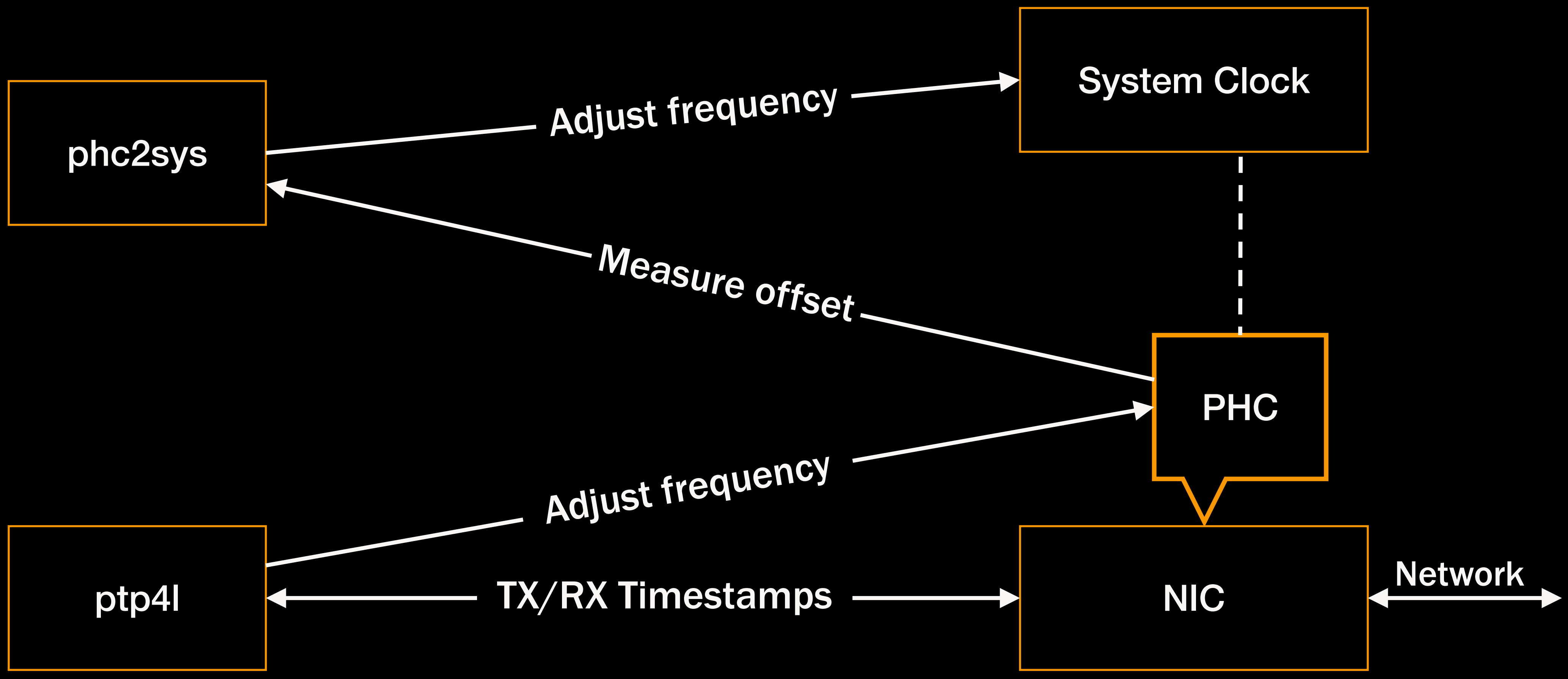
Run ptp4l

- `ptp4l -m -f config.cfg -i eth0`

Run phc2sys to synchronize system time to the PHC time

- `phc2sys -s eth0 -c CLOCK_REALTIME -w -m`

CLIENT SETUP



CLIENT SETUP

Start ptp4l to synchronize the time in PHC

- `ptp4l -m -f config.cfg –i eth0`

Run phc2sys to synchronize system time to the PHC time

- `phc2sys -s eth0 -c CLOCK_REALTIME -w -m`

TIPS&TRICKS

I have superpowers!

KNOWN ISSUES IN LINUXPTP-3.1.1

- NMEA ToD support is buggy
- Embedded leap-second list expired
- NMEA TTY speed is fixed
- Link speed mismatch will cause hidden time offset
- Ts2phc assumes UTC to TAI offset is correctly set with generic ToD source

HINTS

- Run a single ts2phc to get ToD from the GNSS to 2 adapters
  - `ts2phc -f config.cfg -s nmea -c ens2f0 –c ens1f0`
- You can use a different UDS socket when running multiple ptp4l instances `––uds_address`
- Linuxptp tools can take long parameters as arguments too
- Disable NTP when using PTP for system time
- Namespaces can be used to isolate different ptp4l instances (L3)

MONITORING

- ts2phc and ptp4l can work in `--free-running` mode for monitoring
- Correct cable lengths when using physical signals
  - Most cables: 6 ns/m
- Validate 1PPS out on the follower vs 1PPS in on the leader

ACCESSING GNSS REMOTELY

- ser2net
- muplex

- Real Time Kinematic

IN THE NEXT EPISODE…

To infinity and beyond!

https://flic.kr/p/fP1vk3

LINUXPTP 4.0

- Monitor multiple PHCs with phc2sys
- Free-running mode in phc2sys
- PTP minor version change
  - May break compatibility with old GMs
- Virtual clock support
- Dynamic clock tree reconfiguration in ts2phc
- Add read-only UDS socket for monitoring

GITHUB.COM/MACIEKMACHNI/
PTP_RECIPES