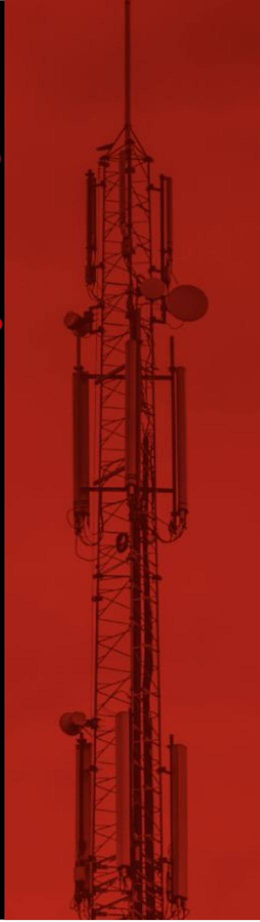
# Getting in Sync with Open Source

Workshop on Synchronization and Timing Systems (WSTS 2021)
April 1, 2021


**Timo Jokiaho**
Chief Technologist
Global Telco Technology Development

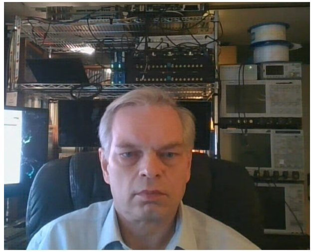
**Pasi Vaananen**
Systems Architect, Office of the CTO
Telco Enablement & Solutions

Red Hat

# Agenda

**Timo Jokiaho**
Chief Technologist
Global Telco Tech. Development
ex-Nokia/Huawei/Samsung



**Pasi Vaananen**
Systems Architect
Office of the CTO
Telco Enablement & Solutions
ex-Nokia/Motorola

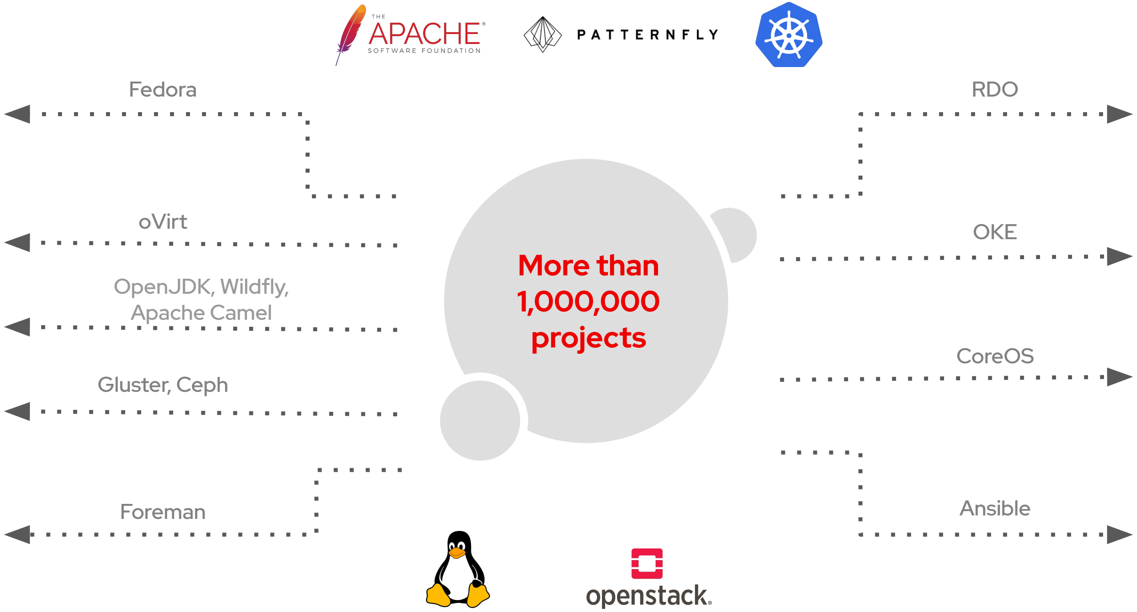
- **Brief Intro to Red Hat**
- **Key Vertical Markets with synchronization interests**
- **Synchronization in Red Hat product portfolio**
- **Synchronization architectures in 5G Cloud RAN use case**
- **5G synchronization implementations in distributed k8s clusters**
- **Red Hat Test Capabilities and some results examples**
- **Conclusions & Next Steps**

# From Communities to Enterprise

## Red Hat Development Model: Open Source - Upstream First

**Stabilize**

We commercialize these platforms together with a rich ecosystem of services and certifications.




**Stabilize**

We commercialize these platforms together with a rich ecosystem of services and certifications.

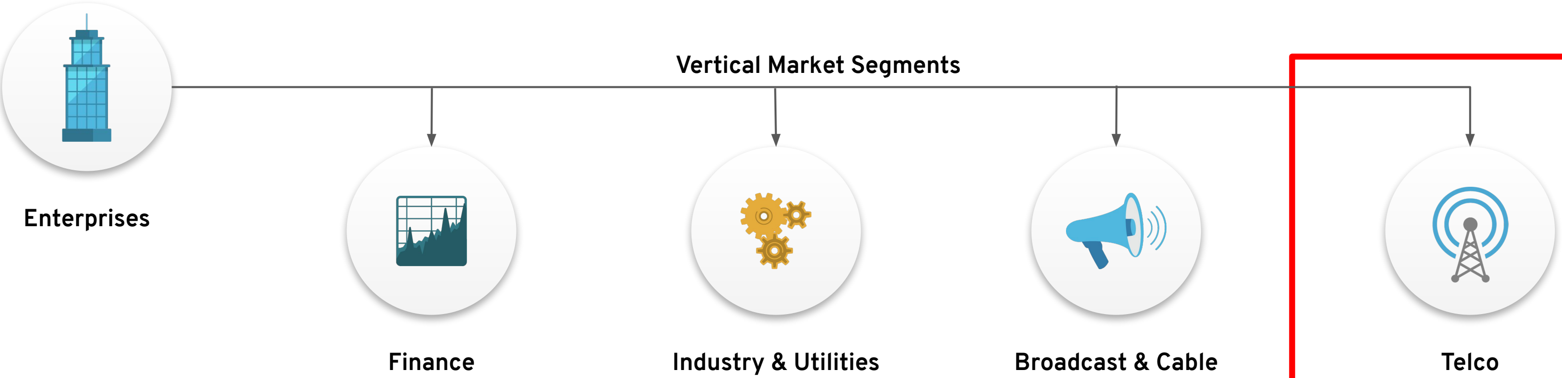
**Integrate**

We integrate upstream projects, fostering open community platforms.

**Participate**

We participate in and create community-powered upstream projects.

# Key Vertical Markets – with Synchronization Interests




| | Enterprises | Finance | Industry & Utilities | Broadcast & Cable | Telco |
|---|---|---|---|---|---|
| **Sync. Use Cases** | Distributed IT Systems, DBs, Private Clouds, B2B & B2C services | Trading systems & Exchanges; monitoring and regulatory compliance | IIoT, Process Control, Manufacturing, Logistics, Transportation, Smart Grid | Video Production & Distribution Chain; Cable MSOs | Radio Access Network (Cloud RAN), enhanced positioning, TaaS |
| **Typical Accuracy Targets** | 100's ms to 1's ms | 10's ms to 1's us | 10's us to 1 us | | 1's us (abs) to <10's ns (rel); NE level specs downto 1's ns |
| **Primary Protocols & Profiles** | NTP → PTP Enterprise Profile; TAP DC profile (WIP) | Regulatory, e.g. MiDIF 2; STAC; PTP enterprise profile | IEEE TSN; 802.1AS; IEC profiles | SMPTE Broadcast Profile; DOCSIS/DTP | ITU-T Telecom Profiles; IEEE TSN FH; MEF; O-RAN; 3GPP |

# Distributed K8s Clouds – synchronization enablement overview

| RAN | CORE | MEC/App |
|---|---|---|

**System / Network Level Mechanisms**; Operations involving multiple clusters / regions

Discovery ⁝ Policy ⁝ Compliance ⁝ Configuration ⁝ Workloads

**Multi-Cluster Management**

| Manage Workloads | Build Cloud-Native Apps | Developer Productivity |
|---|---|---|
| **Platform Services** | **Application Services** | **Developer Services** |
| Service Mesh ⁝ Serverless<br>Builds ⁝ CI/CD Pipelines<br>Full Stack Logging<br>Chargeback | Databases ⁝ Languages<br>Runtimes ⁝ Integration<br>Business Automation<br>100+ ISV Services | Developer CLI ⁝ VS Code extensions ⁝ IDE Plugins<br>Code Ready Workspaces<br>CodeReady Containers |

**Container Platform**

**Cluster Level Mechanisms**; Cluster Synchronization Management, (configuration, performance, fault); for both co-located and distributed cluster nodes and other resources

**Cluster Services**

Automated Ops ⁝ Over-The-Air Updates ⁝ Monitoring ⁝ Registry ⁝ Networking ⁝ Router ⁝ KubeVirt ⁝ OLM ⁝ Helm

**Kubernetes**

**Node Level Mechanisms**; Synchronization protocols, resiliency, monitoring, HW Enablement and "hiding"

**Linux / Real-Time Linux**

**Kubernetes Engine**

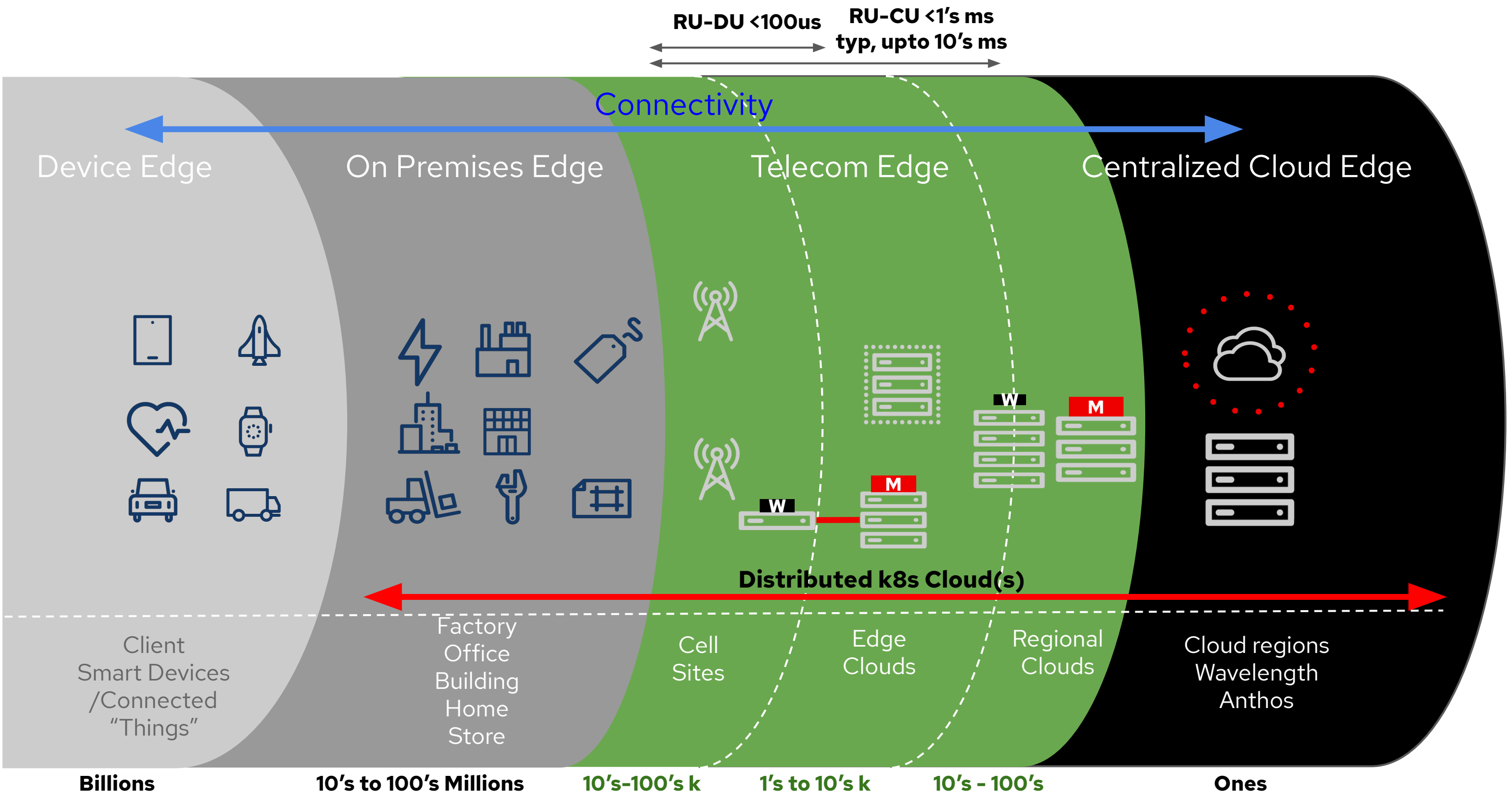
**Primary Deployment Target Environments w/ sync**

**Physical** | **Virtual** | **Private cloud** | **Public cloud** | **Managed cloud** (Azure, AWS, IBM, Red Hat)

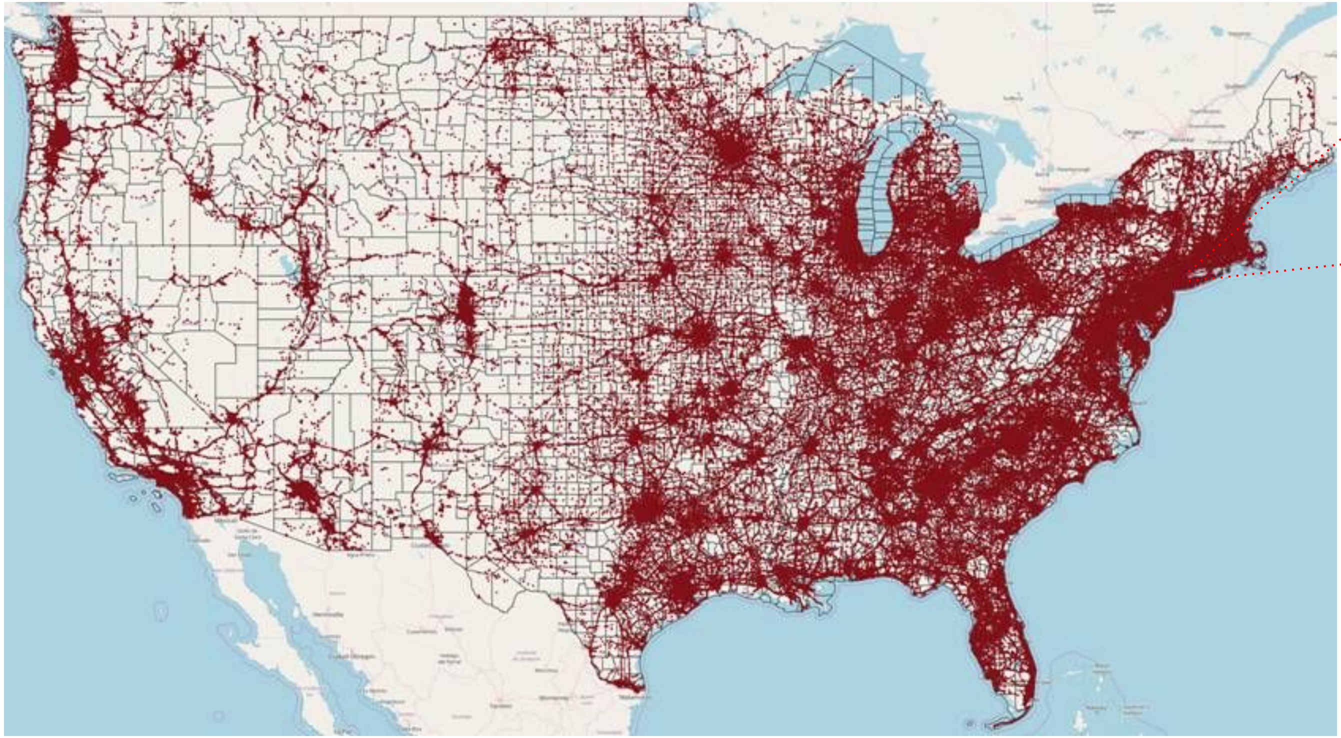
Red Hat

# Edge Classification

## Four Edge Conversations



# RAN Synchronization Challenge in one slide



**NYC: >10k+ cells, >2k+ sites**

**One provider: >500k cell sectors, >60k cell sites; radio & site numbers will increase for 5G, esp. with utilization of FR2 “mmwave” spectrum**

**Synchronize**

- 100’s of k – M’s of distributed Network Elements to 1.5us absolute |TE|
- 100’s to thousands of physical and cloud NEs within a synchronization cluster to O(10’s-100’s) ns relative |TE|,

**while**

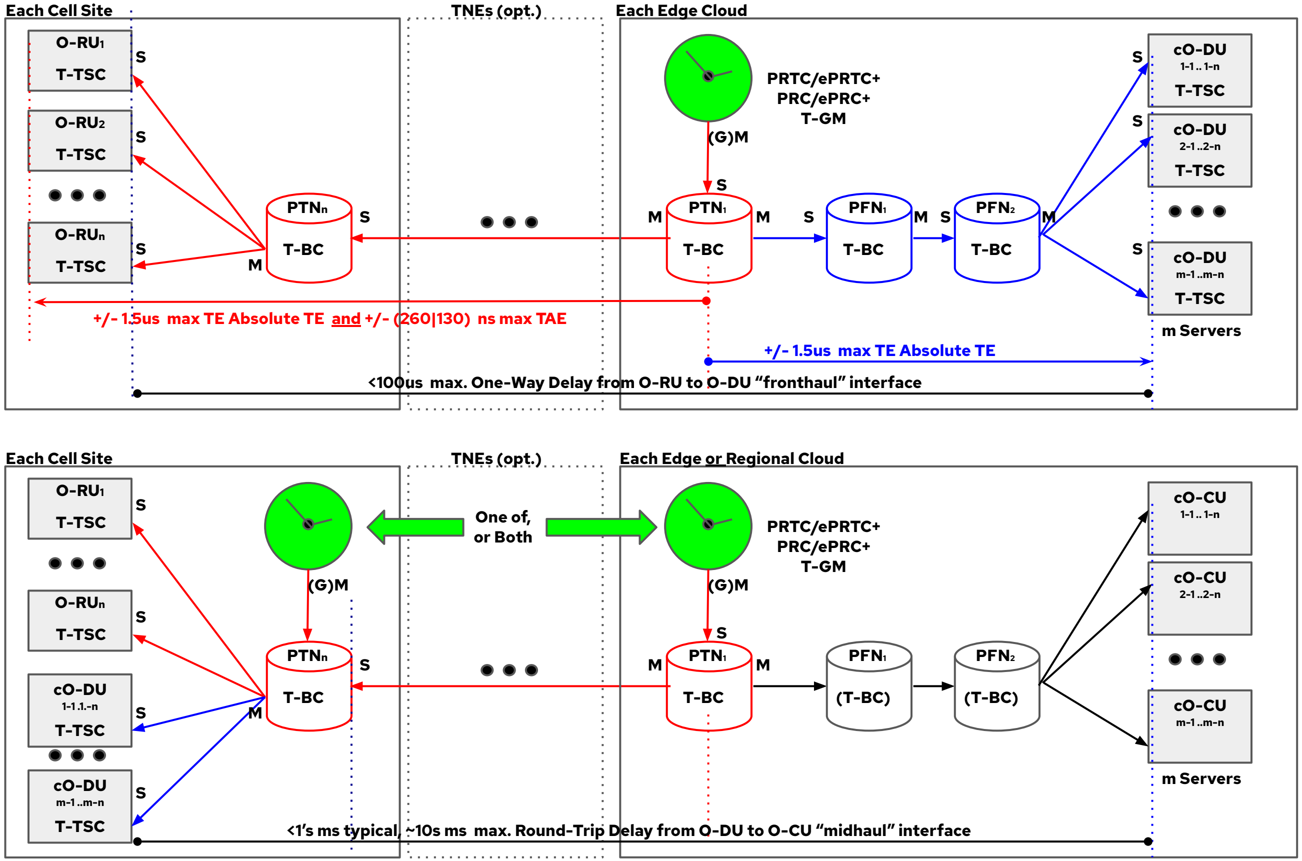
- Minimizing CapEx, Installation and OpEx costs, and
- Maximizing the synchronization performance and availability

**and also**

- Across widely diverse transports
- Interoperability in multi-vendor & multi-technology environment
- With tight freq. stability reqs.

Absolute |TE| (TE=Time Error): maximum absolute time difference against a common reference standard
Relative |TE| (rel. |TAE|=Time Alignment Error): max. relative time difference between two entities (within a sync. cluster)

# 5G (cloud) RAN – Example Deployment Synch. Architectures




## Centralized (C-)RAN:

- DU's are implemented as DU-pool,
- located at Edge Cloud Sites
- In cloud, typ. run in servers with RT kernel + HW acceleration resources
- Each Edge Cloud can serve RUs from upto 400 km$^2$ area
- Area limited by speed of light in fiber & 100us constraint
- Less with PTNs in path
- Many, 10's to 100+ servers/site

## Distributed (D-)RAN:

- DU's distributed to cell sites
- Can still be cloudified
- In cloud, typ. run in servers with RT kernel + HW acceleration resources
- DUs serve only single site cells (typically 1 to 12 sector/band combinations)
- Few, typ. 1 to 3 servers/site

Performance Critical Synch Path, abs|TE| & rel|TE| G8275.1 with SyncE; high / enhanced performance element requirements

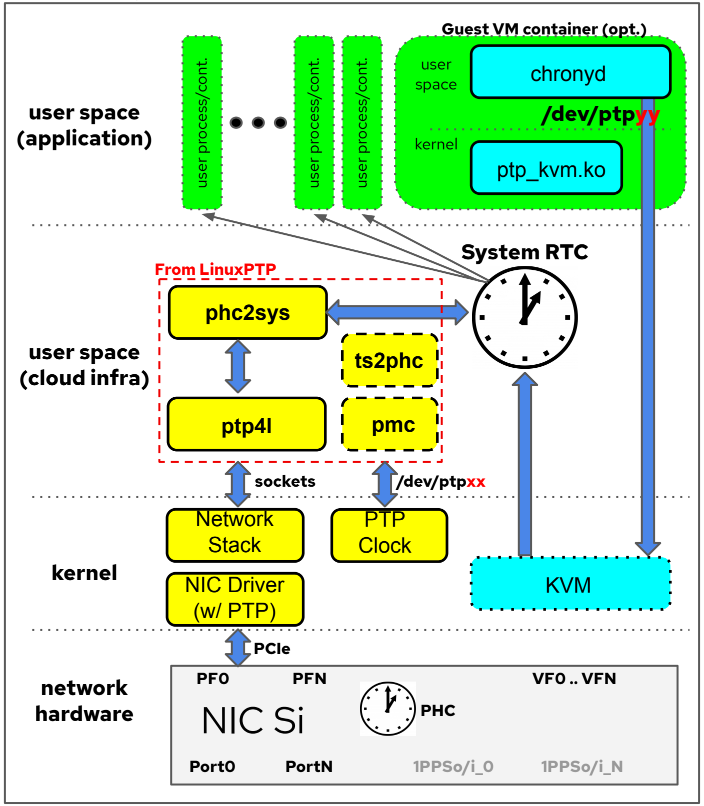
Less Performance Critical Synch Path, abs|TE| only; G8275.1, SyncE optional; Relaxed performance network element requirements

Non-Critical synchronization path, PTP use is optional (may use relaxed PTP impl. or NTP)

Red Hat

# Precision Time Protocol (PTP) in Linux / k8s nodes

## Each Physical Node




## Key Components of the node PTP implementation

- HW specific synchronization SW support features are implemented in HW device drivers (PTP driver)
- HW Clock (PHC) support in NIC Si is required for high accuracy
- **Linuxptp** is an implementation of the PTP SW stack for Linux
- **ptp4l** implements Boundary Clock (BC) and Ordinary Clock (OC), it synchronizes PTP hardware clock (PHC) to remote master clock
- ptp4l is very flexible, and can be configured to support specific profiles, assuming that HW supports the associated profile features / accuracy (e.g. L3 vs. L2 transport)
- **phc2sys** synchronizes two or more clocks in the system, typically used tp synchronize the system clock to PTP / PHC clock
- **pmc** - PMC management client; 1588 basic management access for ptp4l
- [**ts2phc** synchronizes PHC(s) using external timestamps, such as 1PPS in and ToD - used in some HW assisted BC, and GNSS GM implementations]
- [for VMs: **ptp_kvm.ko** is s kernel module with gettime method returning hosts clock, to synchronize a guest to host clock]
- In k8s clusters, synchronization processes are configured and monitored with k8s, and associated general CM, PM and FM event and metrics tools.

**And many other Si & HW Partners**



Red Hat

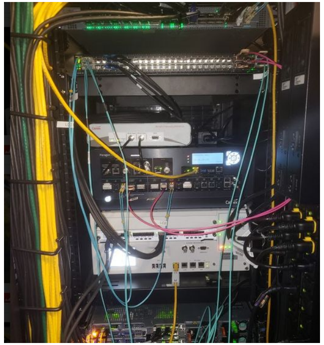
# Synchronization Validation Capabilities at Red Hat

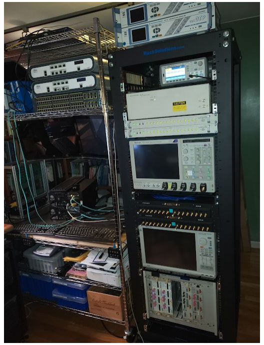
## Generic Time & Frequency test

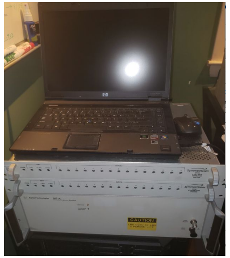
- Rb and **Cs** Primary **Reference Stds**
- **GPSDO** References & **GNSS GMs**
- **Oscilloscopes** up to 100GSPS/16Ghz with advanced jitter/time/delay statistical analysis SW options
- **Freq / time interval counter** 20ps res.
- **Frequency Stability Analyzers**
- **Network emulators** - delay/loss/jitter
- **Logic analyzer** with 50ps resolution
- **PCIe analyzer** / slot interposer
- **Spectrum analyzer** w/ Phase noise analysis & RAN analysis SW
- RF/coax & optical switches for test setup reconfigs & fault injection testing
- Freq/Phase perf. analysis SW
- PTP network simulation SW tools
- Automation and data analysis tools

## PTP validation & profile conformance test

- Calnex/Spirent **Paragon Neo** 1/10/25/100G Ethernet **PTP conformance test set**
- **Spirent TestCenter** 10/25/40/50/100G Load/delay/jitter test set
- 10/25/50/100G **packet generation and capture cards**, 1ns timestamping
- Optical splitters for monitoring and non-intrusive measurement tapping
- Calibrated / matched delay cables (opt & coax)
- PTP optimized/compliant **Switches & Routers**
- **NICs** and **FPGA cards** w/ PTP HW support









Red Hat

# Set-up for a NIC; G.8275.1 & G.8273.2 T-TSC

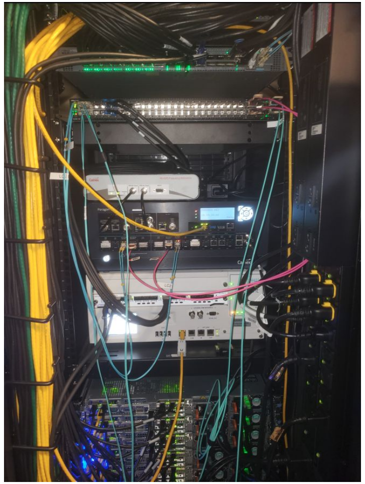
**NOTES:**

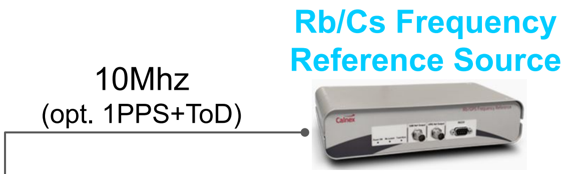
G.8275.1 Profile configuration

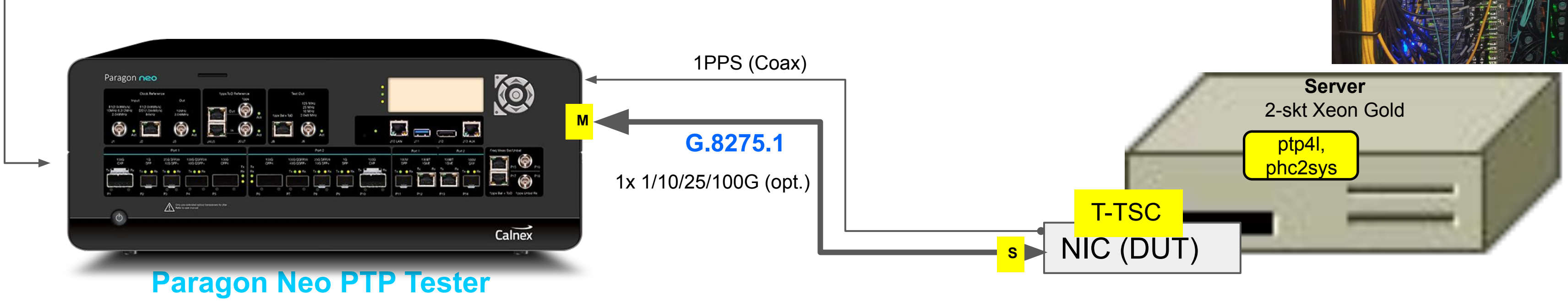
- DUT & test setup configured for T-TSC
- One-step Timestamp
- L2 Transport, no VLAN tagging
- 16 delay req/sync, 8 announces per second
- No SyncE used (SyncE is optional in these cases)
- SyncE is expected to improve accuracy and/or hold-over performance at T-TSC, if supported

G.8273.2 T-TSC compliance

- Time error, noise tolerance, noise transfer, transient response, holdover, redundancy, ...
- Assess performance of implementation to perf Classes (A to D in increasing level of accuracy)







Red Hat

# An example of T-TSC config test result subset

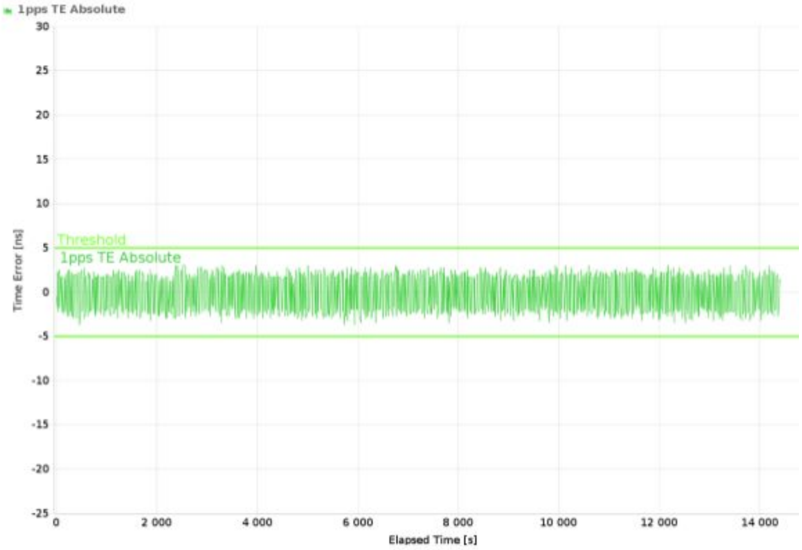
Mask results:

| | |
|---|---|
| All Mask Results | Pass |

| | |
|---|---|
| Mask Sync PDD Result | Pass |
| Mask Sync CDF Result | Pass |
| Mask Delay Req PDD Result | Pass |
| Mask Delay Req CDF Result | Pass |
| Mask 1pps TE Absolute Time Error Result | Pass |
| Mask 1pps TE Absolute Time Error (Filtered) Result | Pass |
| Mask 1pps TE Absolute Avg Time Error (cTE) Result | Pass |
| Mask 1pps TE Absolute Time Error (Transient Response) Result | Pass |

Time Error Analysis

| | |
|---|---|
| Offset Removal Applied | Off |
| Zero Offset | -0.477 ns |




| | |
|---|---|
| Mean | -0.153 ns |
| Min | -3.727 ns |
| Max | 3.023 ns |
| Max-Min | 6.75 ns |

**LinuxPTP is up to the task:**

- Even to the most demanding performance classes as defined in G.8273.2 (in this case)
- Results are highly dependent on the HW capabilities and specific configurations
- We work closely with our HW partners on enablement and performance aspects



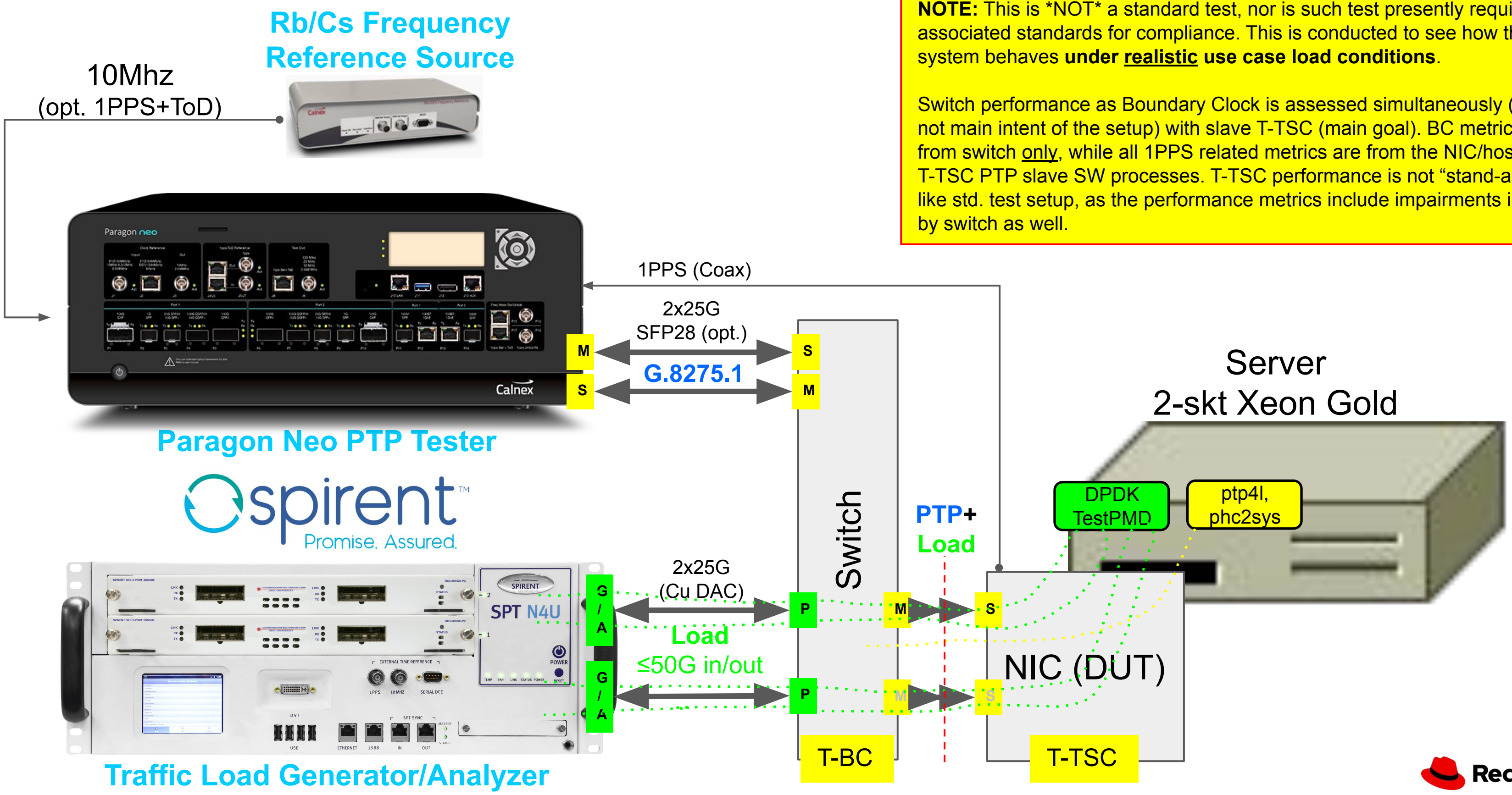
**NOTE:** each test run generates a report in tens of pages, this is limited to small set of key highlights. In addition, we capture and analyze LinuxPTP statistics and events during the run, and in some cases other relevant ancillary data, such as load/latency/jitter etc. on the under-load tests.

Red Hat

# Set-up for a NIC+switch; G.8273.2 T-BC & T-TSC

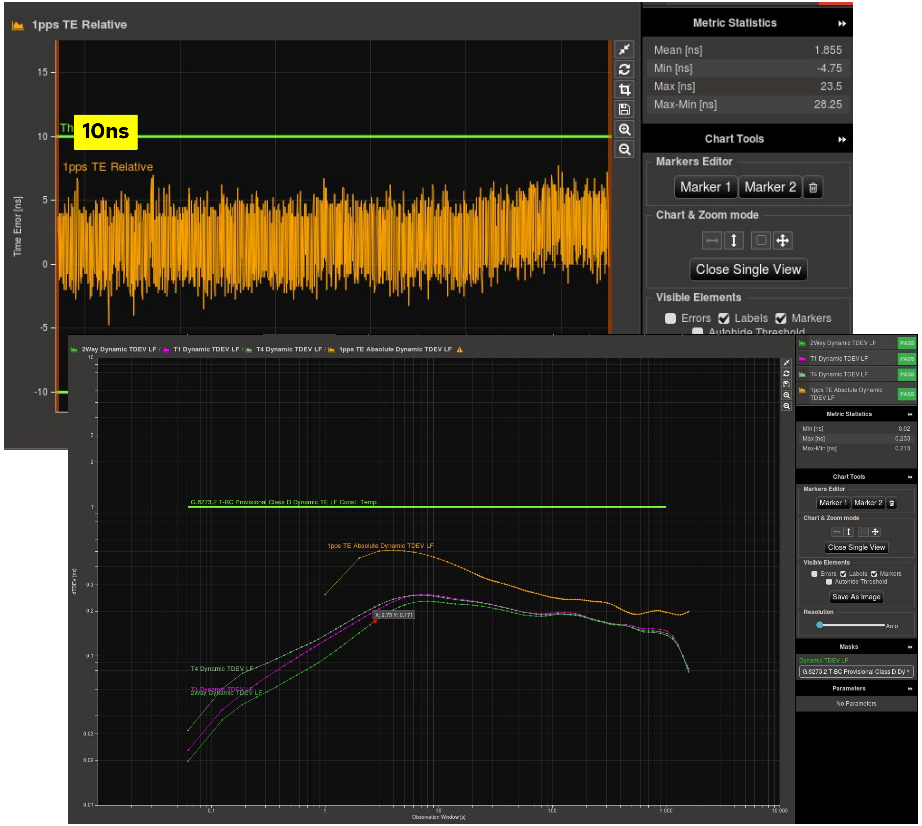
**NOTE:** This is *NOT* a standard test, nor is such test presently required by associated standards for compliance. This is conducted to see how the system behaves **under realistic use case load conditions**.

Switch performance as Boundary Clock is assessed simultaneously (this is not main intent of the setup) with slave T-TSC (main goal). BC metrics are from switch only, while all 1PPS related metrics are from the NIC/host & T-TSC PTP slave SW processes. T-TSC performance is not "stand-alone" like std. test setup, as the performance metrics include impairments induced by switch as well.

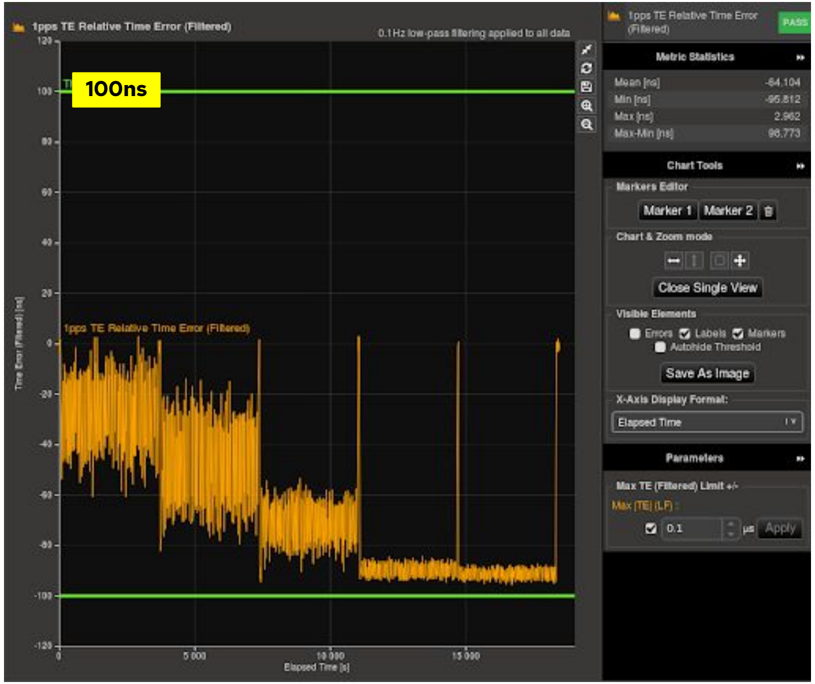


# T-TSC Performance Examples With load (1PPS i/f)

## Switch & load: a good example




## Switch & load: a less good example




Both left & right configs are loaded at 10/30/50/80/100% of interface max. rate



Red Hat

# Set-up for a NIC; G.8275.1 & G.8273.2 T-BC

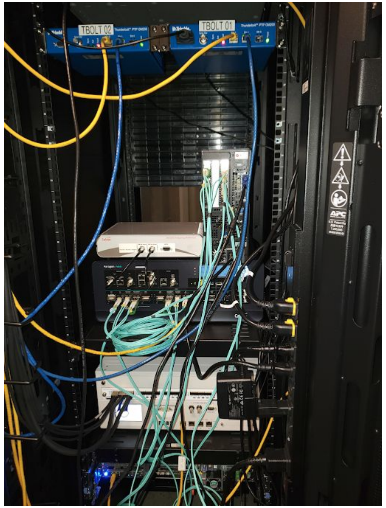
**NOTES:**

G.8275.1 Profile configuration

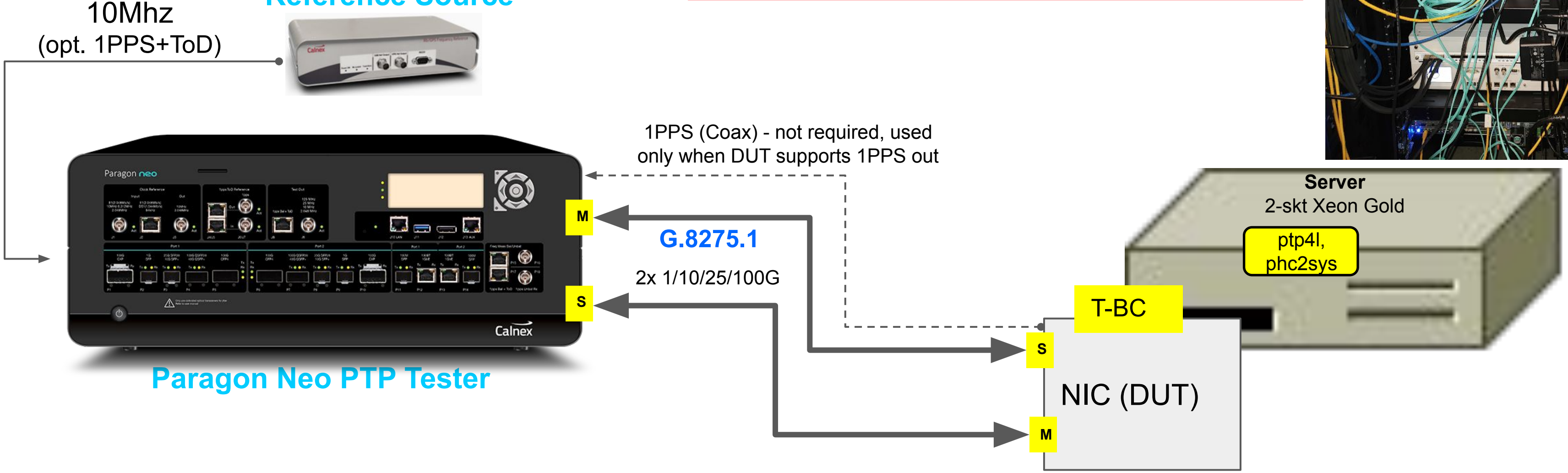
- DUT & test setup configured for T-BC
- One/two-step Timestamp
- L2 Transport, no VLAN tagging
- 16 delay req/sync, 8 announces per second
- NO SyncE used (yet, will be tested when HW available !)

G.8273.2 T-BC compliance

- Time error generation / tolerance / transfer, short-term transient response, holdover, ...
- Assess performance of implementation to perf Classes (A to D in increasing level of accuracy)



Rb/Cs Frequency Reference Source



# An example of T-BC config test result subset (25G, PTP i/f)

Mask results:

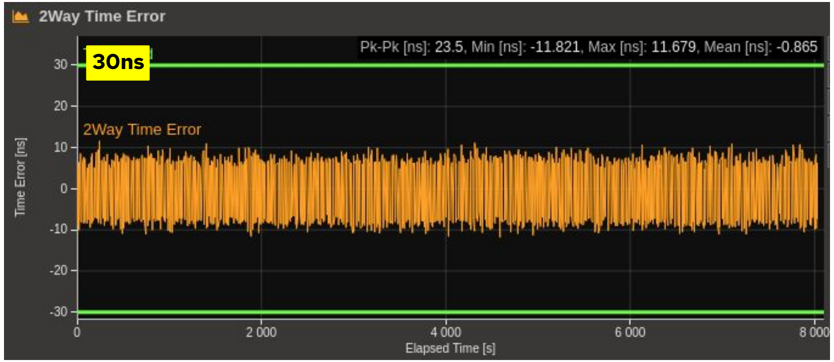
| | |
|---|---|
| All Mask Results | Pass |

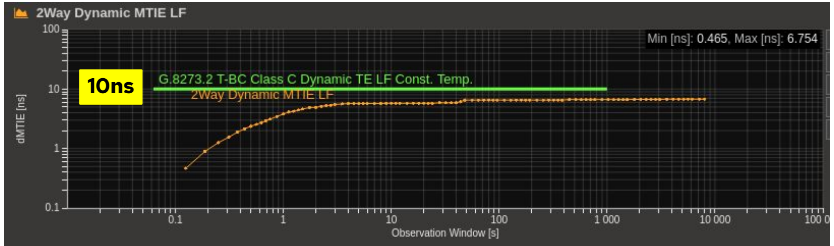
| | |
|---|---|
| Mask Sync PDD Result | Pass |
| Mask Sync CDF Result | Pass |
| Mask Sync Packet MTIE Result | Pass |
| Mask Sync Packet TDEV Result | Pass |
| Mask Sync FPC Result | Pass |
| Mask Sync FPR Result | Pass |
| Mask Delay Req PDD Result | Pass |
| Mask Delay Req CDF Result | Pass |
| Mask Delay Req Packet MTIE Result | Pass |
| Mask Delay Req Packet TDEV Result | Pass |
| Mask Delay Req FPC Result | Pass |
| Mask Delay Req FPR Result | Pass |
| Mask 2Way Time Error Result | Pass |
| Mask 2Way Time Error (Filtered) Result | Pass |
| Mask 2Way Dynamic MTIE LF Result | Pass |
| Mask 2Way Dynamic TDEV LF Result | Pass |

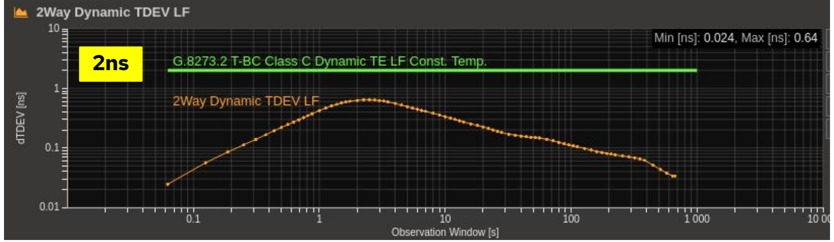
2Way Time Error PASS

| Metric Statistics | |
|---|---|
| Pk-Pk [ns] | 23.5 |
| Mean [ns] | -0.865 |
| Min [ns] | -11.821 |
| Max [ns] | 11.679 |
| Max-Min [ns] | 23.5 |

| Metric Statistics 10ns | |
|---|---|
| Constant Time Error [ns] | -0.865 |
| Min [ns] | -0.931 |
| Max [ns] | -0.726 |
| Max-Min [ns] | 0.205 |




## LinuxPTP is up to the task:

- TE Generation perf. to G.8273.2 Class C
- Results are highly dependent on the HW capabilities and specific configurations
- Will test w/ SyncE as HW becomes available



**NOTE:** each test run generates a report in tens of pages, this is limited to small set of key highlights. In addition, we capture and analyze LinuxPTP statistics and events during the run, and in some cases other relevant ancillary data, such as load/latency/jitter etc. on the under-load tests.

# Stuff we are working on / planning for

**Note:** this is not a roadmap statement, we work according to our customer needs - contact us to discuss yours !

- 1st priority was on Slave Compliance and performance
  - Slave is a foundational capability to build on for more complex use cases (e.g 1st half of BC)
  - Profile specific compliance (protocol profile and performance requirements)
  - Performance enhancements and tuning (PCIe, filtering/servo algorithms and parameters, QoS)
- 2nd priority is on Telecom Boundary Clock implementations on Linux / cluster nodes
  - Including adding **SyncE and ESMC support**
  - Requires HW support, multiple commercial cards expected to be available in 2021
  - "Smart NICs" & FPGA cards with synchronization optimizations / enhancements
- **Grandmasters** on nodes, with Integrated **GNSS receivers**
  - Requires HW support, multiple options coming with integrated receiver options
  - External GNSS receivers with physical interfaces (typ. serial management/control & ToD, 1PPS and 10Mhz) in combination with open source PTP S can be used today
- Keeping up with **evolving standards** and **new hardware**
  - Profiles are under constant upgrades, esp. In Telco due to 5G tightening of the requirements
  - PTP "base" specification IEEE1588-2008 → -2019, including White Rabbit enhancements
- **Management and monitoring** functionality and facilities
  - Performance at node, cluster, and network levels
  - Configuration-, Fault-, and Performance Management; large-scale operations automation
- Synchronization SW stack implementations for open/whitebox HW
  - Open switches/routers, cell-site routers/GWs, FHGWs, RU's, GNSS GMs, ...
  - Working on enablement with select HW and SW partners

# Get Involved & Get In Touch

**Upstream projects - SW & open HW**

- Chrony (primarily NTP)
- LinuxPTP project
- Linux kernel - sync if/s
- Linux Drivers - sync features
- OpenCompute TAP
- OpenCompute Networking
- OpenCompute Telco
- OpenCompute Telco Edge
- TIP OOPT
- TIP RAN projects
- ...

**How to contact Red Hat**


Timo Jokiaho: tjokiaho@redhat.com

Pasi Vaananen: pvaanane@redhat.com


**Special Thanks to**


- LinuxPTP community, Linux Kernel Community, our NIC Si/card & FPGA Si/card HW partners, vRAN SW partners, Calnex & Spirent, and
- Alex Hurd, Miroslav Lichvar, James Parker, Yolanda Robla Mota, Ken Young, Jianzhu Zhang

# Thank you

Red Hat is the world's leading provider of

enterprise open source software solutions.

Award-winning support, training, and consulting

services make

Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/redhatinc

twitter.com/RedHat

Red Hat