OPC UA PubSub over TSN – Reality check

Carsten Emde

Open Source Automation Development Lab

(OSADL) eG

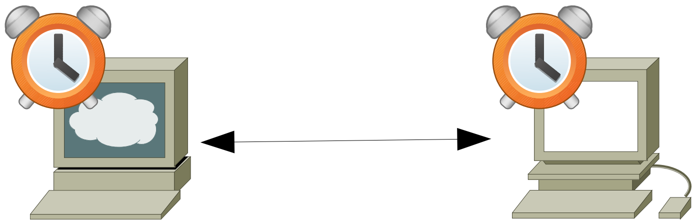
What is Real-time Ethernet?

- It is expected that the time from sending a network packet out of userspace at one peer to receiving this packet in userspace at another peer is deterministic, i.e. it never exceeds a predefined limit (the so called „worst-case latency“).
- To this end, the two communication partners – sender and receiver – must guarantee deterministic response including the operating system and network adapters.
- In addition, the low-level transmission protocol must be deterministic (unrestricted Ethernet is not).
- There are two fundamentally different approaches:
  1. Exclusive duplex connection between two real-time systems using UDP protocol (“Real-time with peer-to-peer Ethernet via UDP“)
  2. Precise time synchronization of the involved real-time systems and using time stamps in packets (“Precise time synchronization”)

Real-time with peer-to-peer Ethernet via UDP

Prerequisites (**without** concurrent non-real-time traffic)

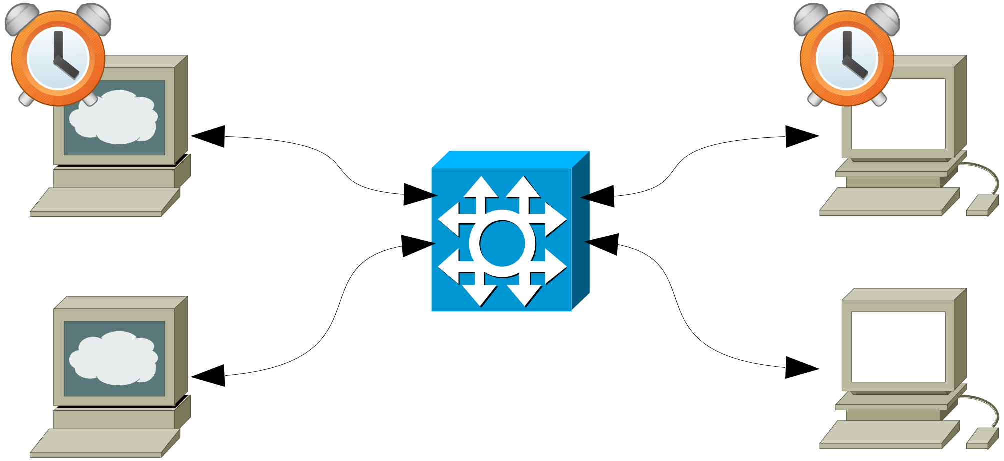
- Sender and receiver are real-time compliant.
- Sender and receiver are directly connected (no switch).
- The network adapters use duplex mode.
- Sender and receiver application are running as real-time task.
- All software interrupt handlers are running as real-time thread.



Real-time with peer-to-peer Ethernet via UDP

Prerequisites (**with** concurrent non-real-time traffic)

- Sender and receiver are real-time compliant.
- Sender and receiver are directly connected (no switch).
- The network adapters use duplex mode.
- Sender and receiver application are running as real-time task.
- All software interrupt handlers are running as real-time thread.
- All real-time networks have VLAN configured at highest priority.

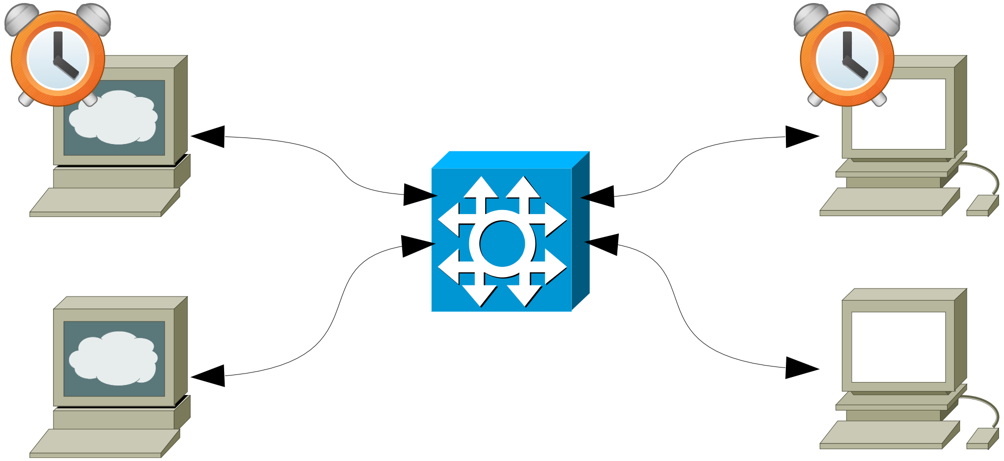


Real-time with peer-to-peer Ethernet via UDP

Prerequisites (**with** concurrent non-real-time traffic)

- Sender and receiver are real-time compliant.
- Sender and receiver are directly connected (no switch).
- The network adapters use duplex mode.
- Sender and receiver application are running as real-time task.
- All software interrupt handlers are running as real-time thread.
- All real-time networks have VLAN configured at highest priority.

Including in switch, if any



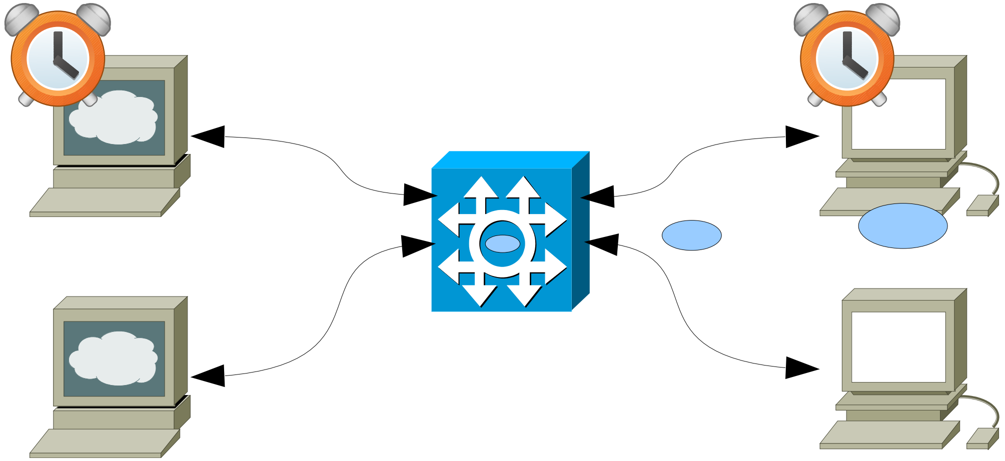
Additional current topic: OPC UA PubSub over TSN – Reality check
Technical Heidelberg OSADL Talks, September 30, 2020, Online Session 1
Open Source Automation Development Lab (OSADL), Heidelberg

Real-time with peer-to-peer Ethernet via UDP

Prerequisites (**with** concurrent non-real-time traffic)

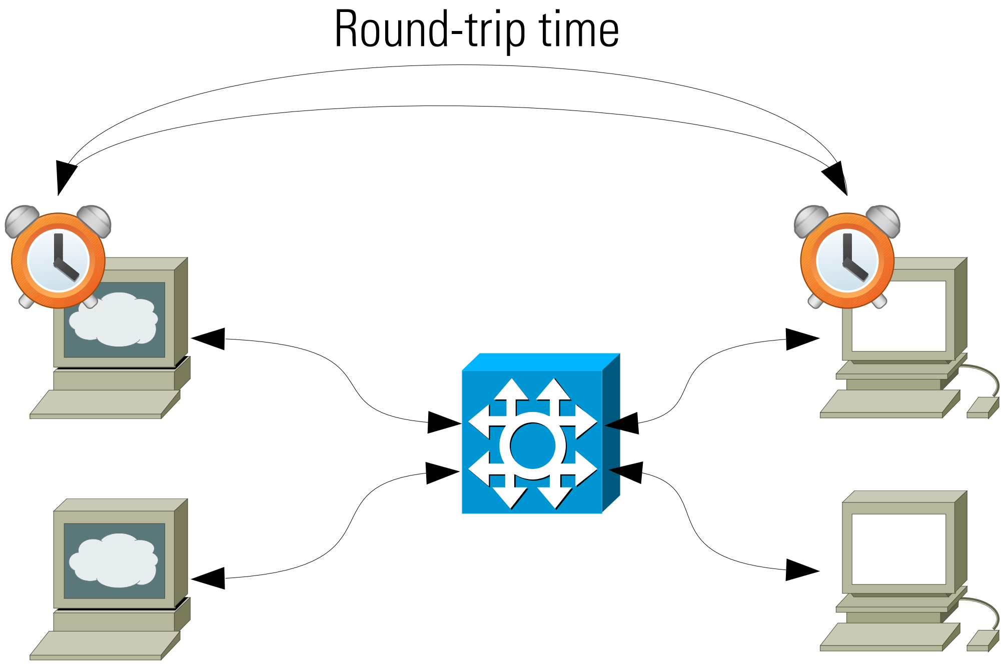
- Sender and receiver are real-time compliant.
- Sender and receiver are directly connected (no switch).
- The network adapters use duplex mode.
- Sender and receiver application are running as real-time task.
- All software interrupt handlers are running as real-time thread.
- All real-time networks have VLAN configured at highest priority.

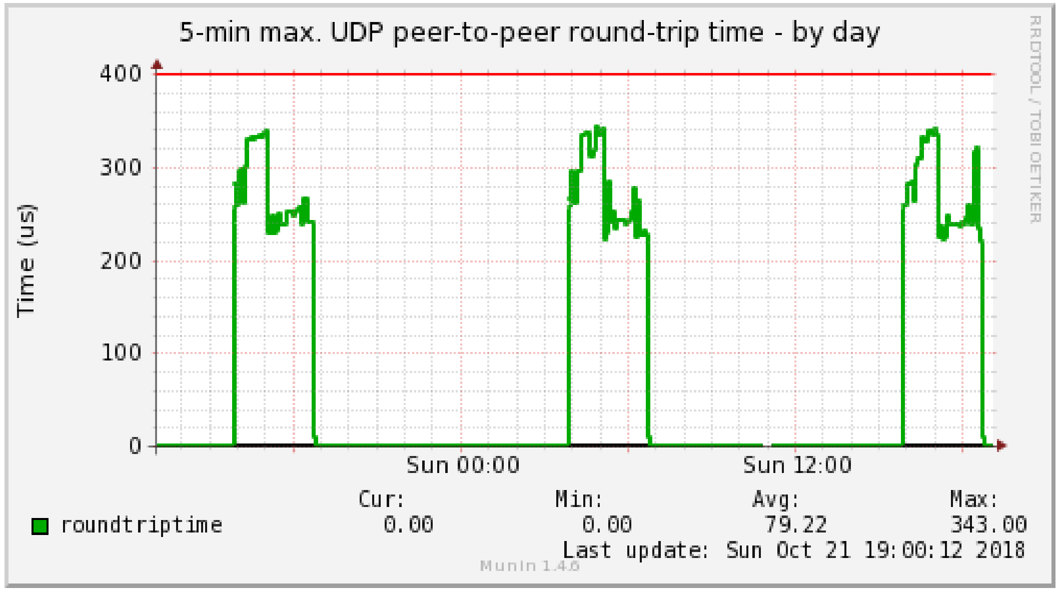
Including in switch, if any



Bandwidth limitation may be necessary

Real-time with peer-to-peer Ethernet via UDP Performance



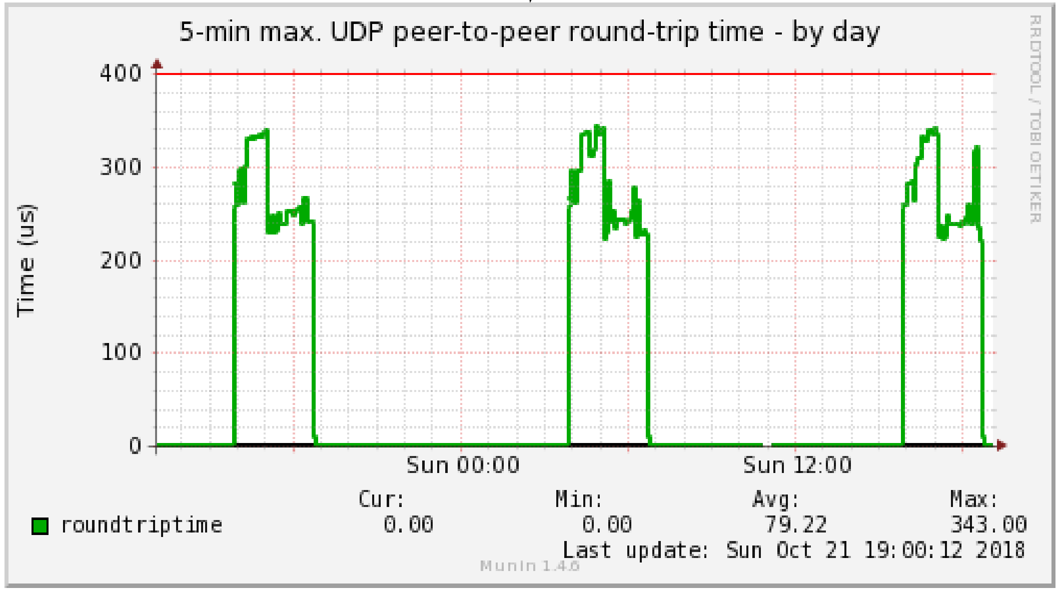


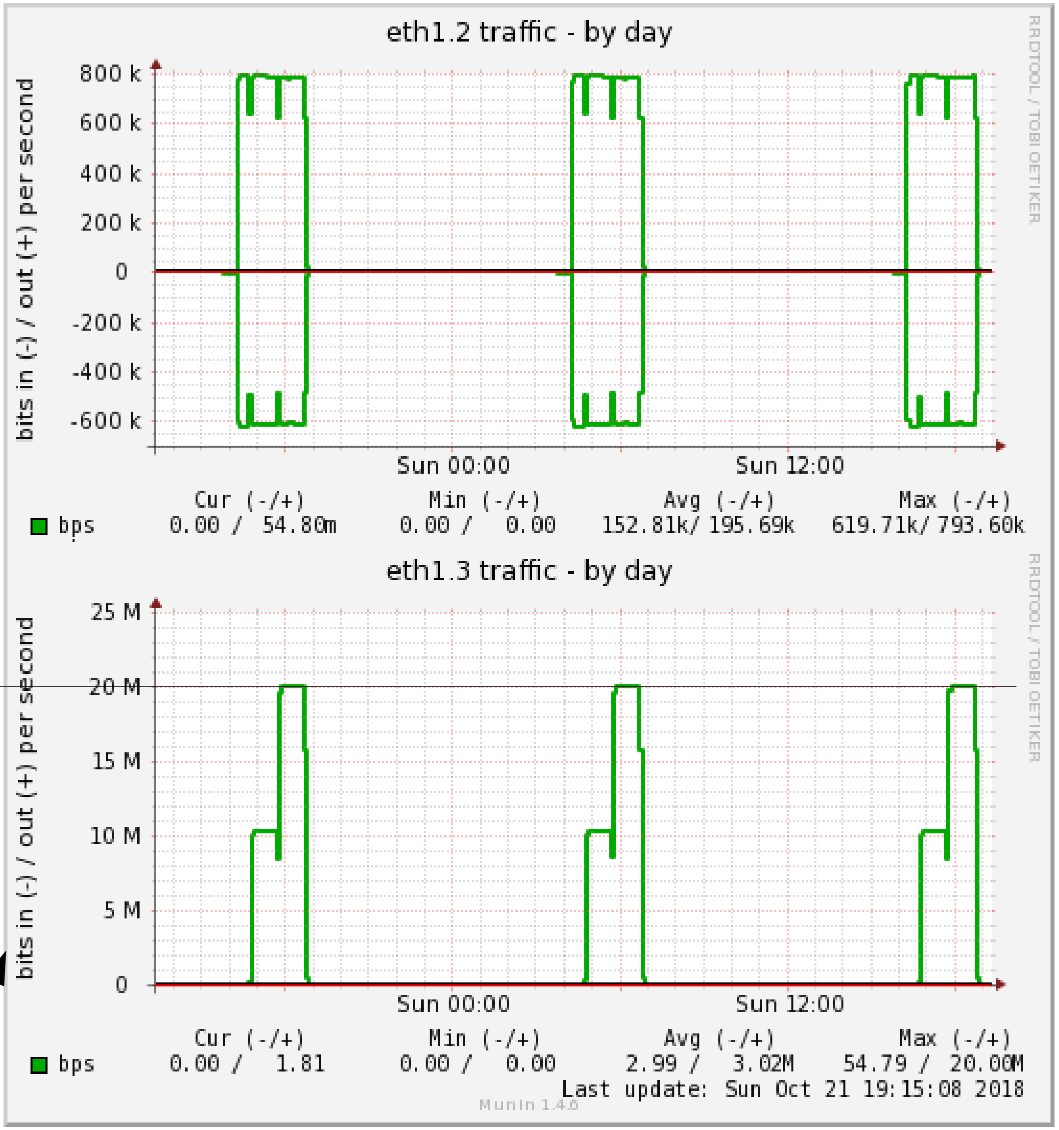
Duration of measurement: 3 h

Real-time traffic	Non-real-time traffic
Cycle interval: 500 µs	Continuous load
Bandwidth: 800 kbit/s	Bandwidth: 10 or 20 Mbit/s

Real-time with peer-to-peer Ethernet via UDP
Real-time *vs.* non-real-time traffic

Real-time traffic





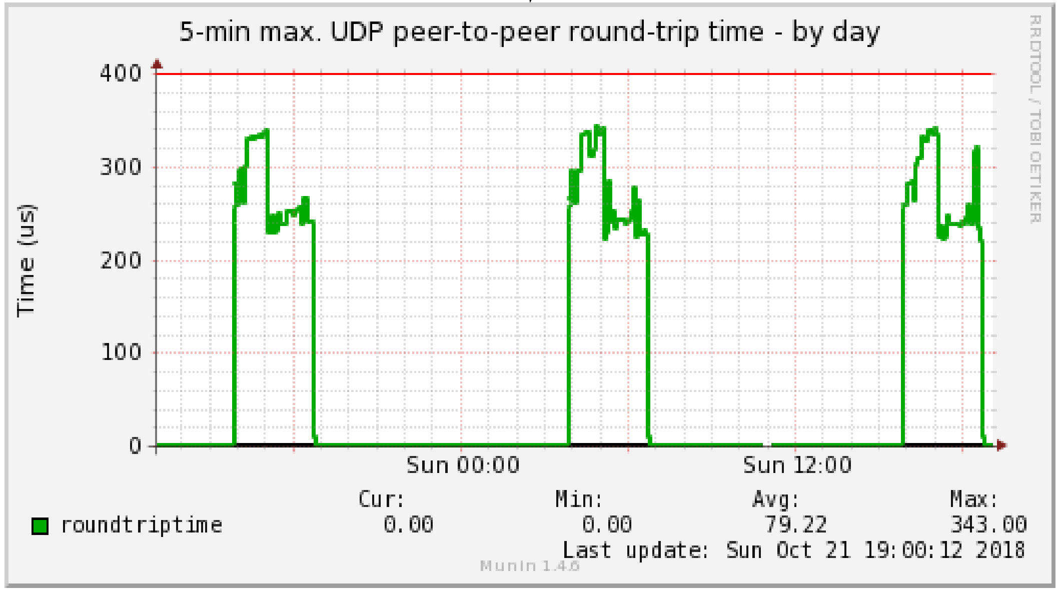
Bandwidth limitation

Non-real-time traffic

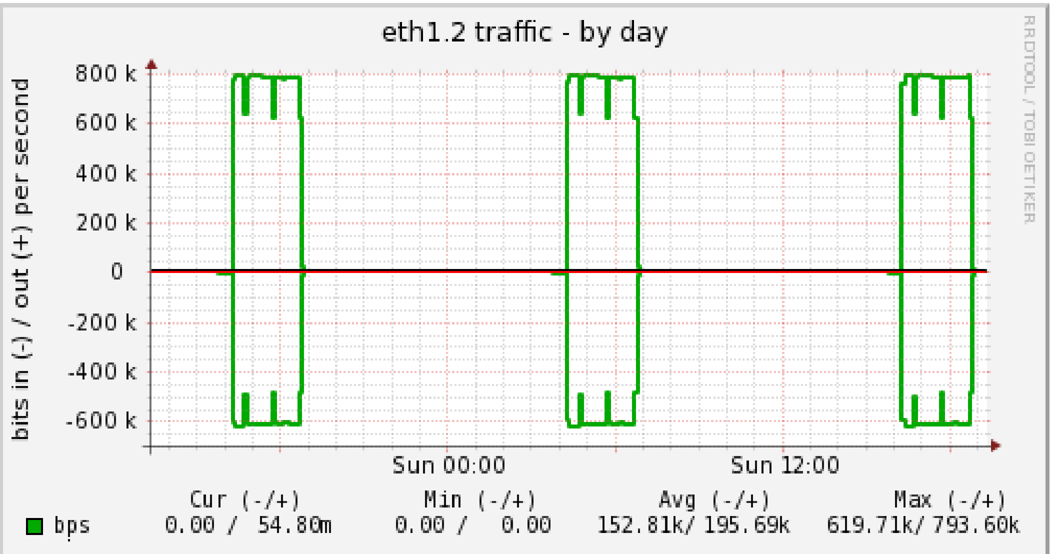
Additional current topic: OPC UA PubSub over TSN – Reality check
Technical Heidelberg OSADL Talks, September 30, 2020, Online Session 1
Open Source Automation Development Lab (OSADL), Heidelberg

Real-time with peer-to-peer Ethernet via UDP
Real-time *vs.* non-real-time traffic

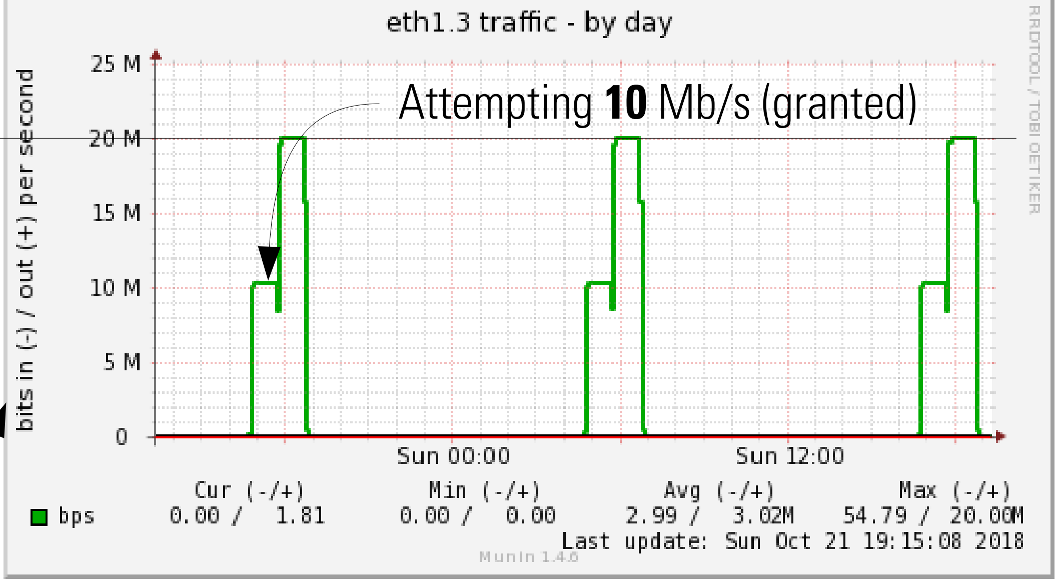
Real-time traffic


5-min max. UDP peer-to-peer round-trip time - by day

```
                 Cur:        Min:        Avg:        Max:
🟩 roundtriptime  0.00        0.00        79.22       343.00
                      Last update:  Sun Oct 21 19:00:12 2018
```


eth1.2 traffic - by day

```
        Cur (-/+)       Min (-/+)       Avg (-/+)          Max (-/+)
🟩 bps   0.00 /  54.80m  0.00 /   0.00   152.81k/ 195.69k   619.71k/ 793.60k
```


eth1.3 traffic - by day

Attempting **10** Mb/s (granted)

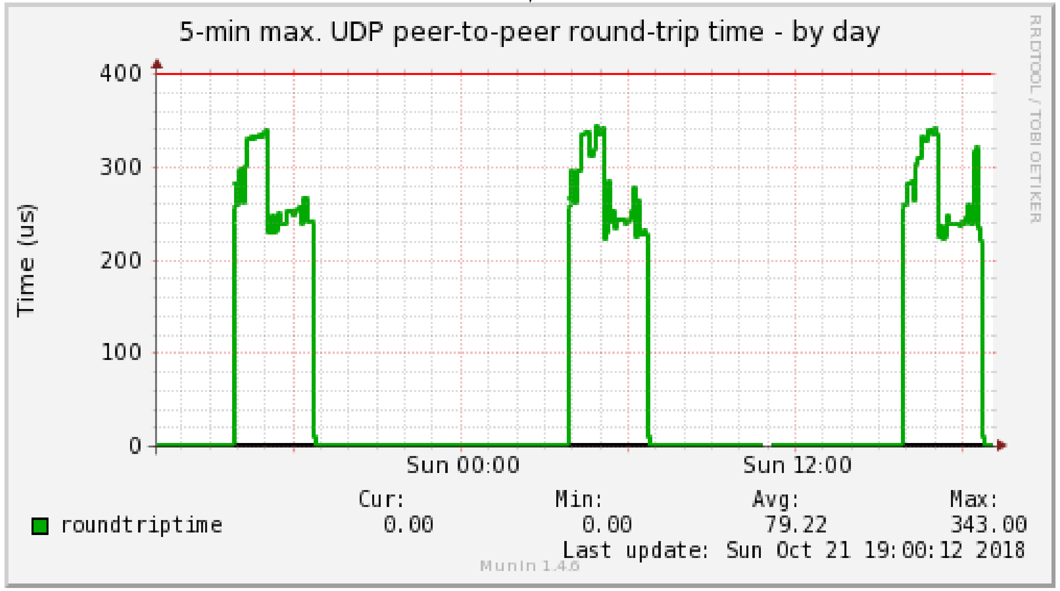
```
        Cur (-/+)       Min (-/+)       Avg (-/+)       Max (-/+)
🟩 bps   0.00 /   1.81   0.00 /   0.00   2.99 /   3.02M  54.79 /  20.00M
                            Last update:  Sun Oct 21 19:15:08 2018
```

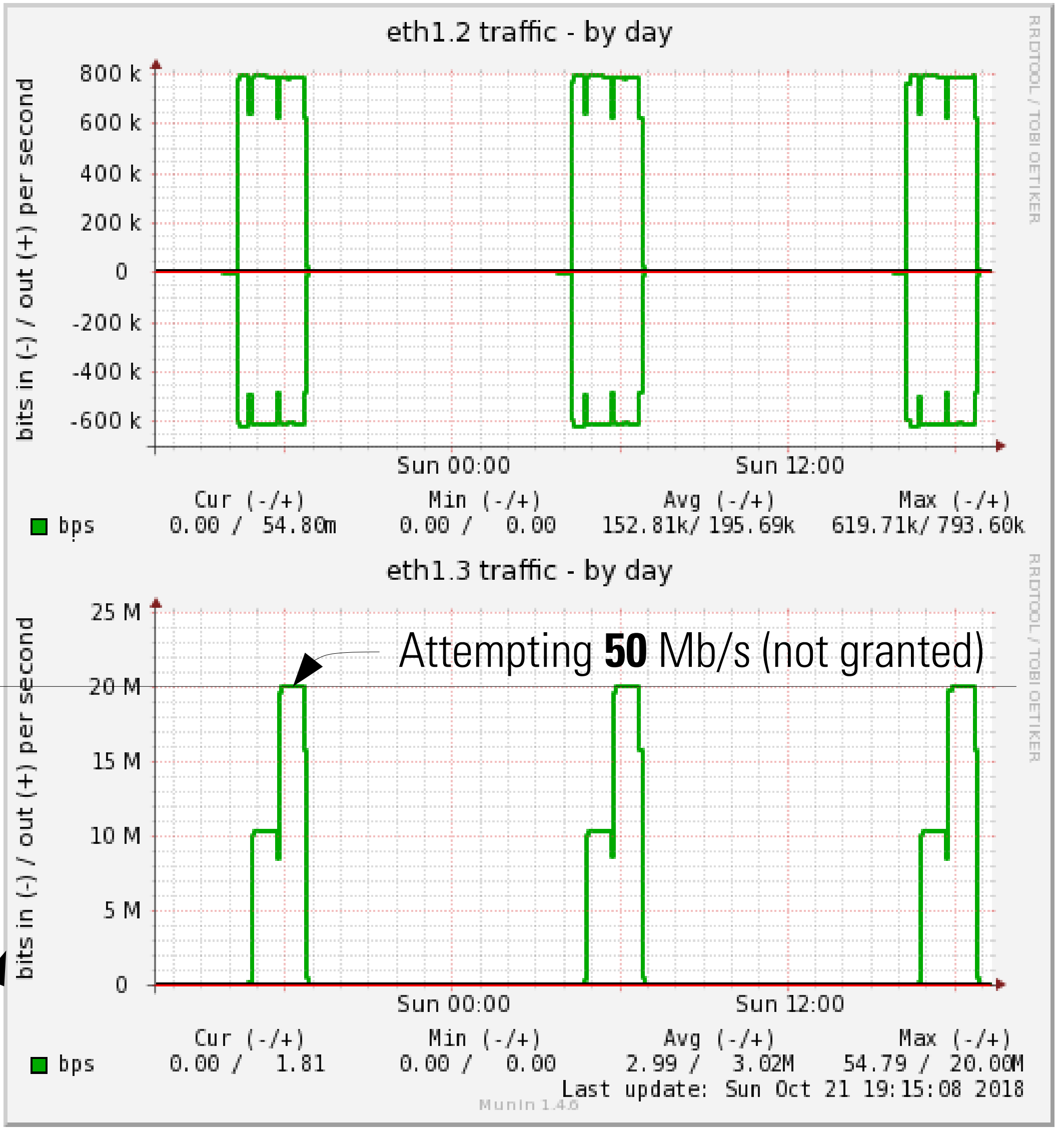
Bandwidth limitation

Non-real-time traffic

Real-time with peer-to-peer Ethernet via UDP
Real-time *vs.* non-real-time traffic

Real-time traffic





Attempting **50** Mb/s (not granted)

Bandwidth limitation

Non-real-time traffic

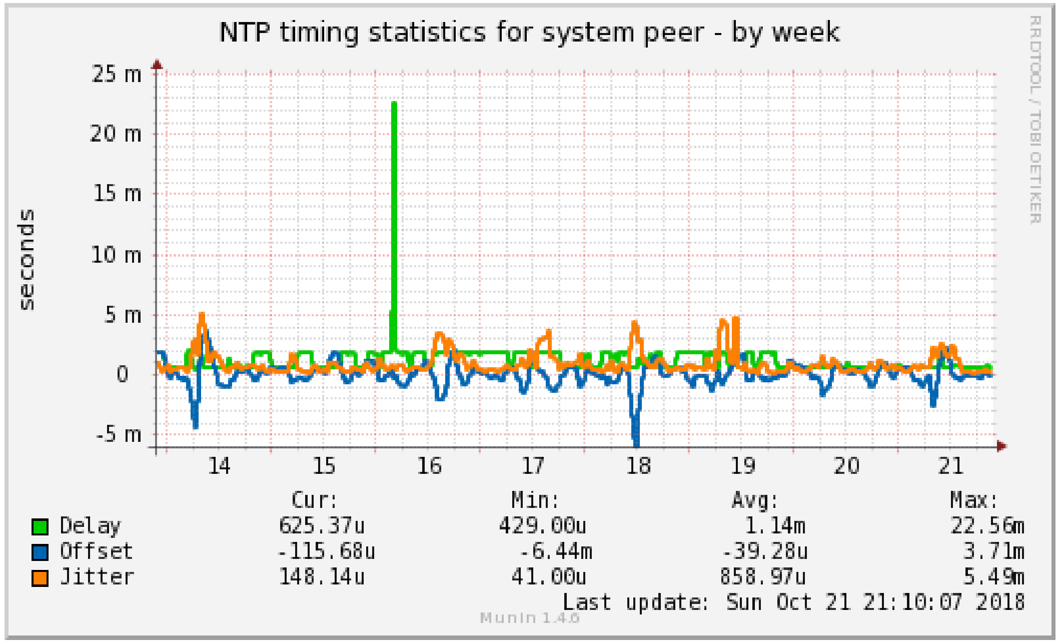
Real-time with peer-to-peer Ethernet via UDP Advantages/Disadvantages

- Advantages
  - Completely Open Source, not patented
  - Does not violate Ethernet standard
  - Readily available in Linux kernel since many years
  - No special hardware required
  - Applicable with standard switches that support VLAN
- Disadvantages
  - Restricted to peer-to-peer topology
  - Worst-case latency relatively high (about 200 µs)

Precise time synchronization

- Established method
  - Network Time Protocol (NTP)
- New method
  - Time-Sensitive Network (TSN)

Network Time Protocol (NTP)

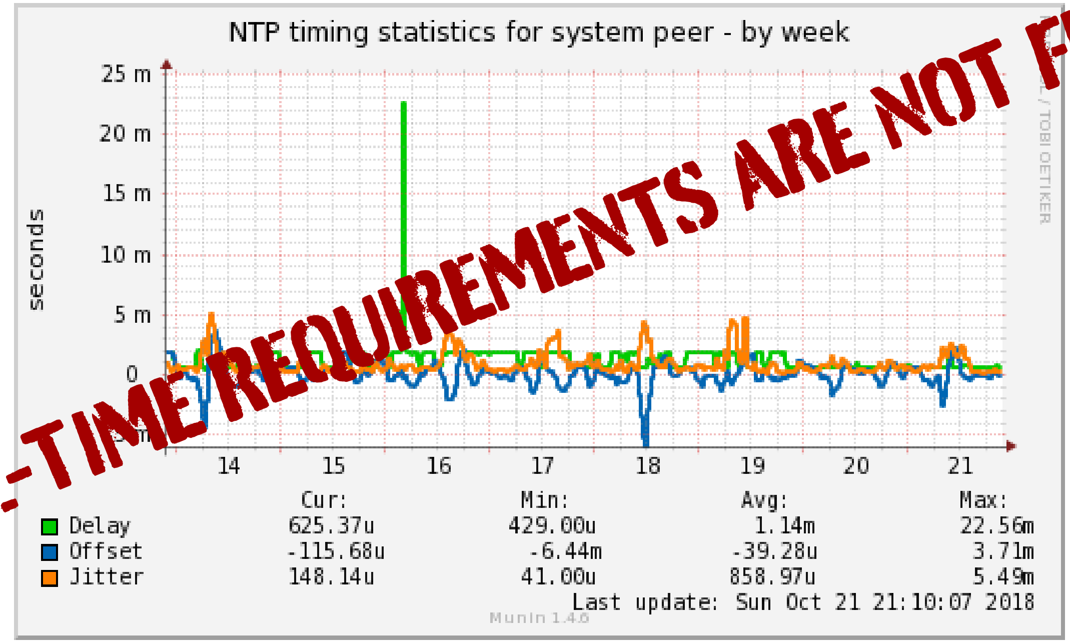


	Cur:	Min:	Avg:	Max:
🟩 Delay	625.37u	429.00u	1.14m	22.56m
🟦 Offset	-115.68u	-6.44m	-39.28u	3.71m
🟧 Jitter	148.14u	41.00u	858.97u	5.49m

Last update: Sun Oct 21 21:10:07 2018

Example of the 1-week time course of the time difference between two NTP-synchronized computers (data obtained from OSADL QA Farm) – Average ± 5 ms with an outlier of 22.5 ms.

Network Time Protocol (NTP)



	Cur:	Min:	Avg:	Max:
🟩 Delay	625.37u	429.00u	1.14m	22.56m
🟦 Offset	-115.68u	-6.44m	-39.28u	3.71m
🟧 Jitter	148.14u	41.00u	858.97u	5.49m

Last update: Sun Oct 21 21:10:07 2018

USUAL REAL-TIME REQUIREMENTS ARE NOT FULFILLED

Example of the 1-week time course of the time difference between two NTP-synchronized computers (data obtained from OSADL QA Farm) – Average ± 5 ms with an outlier of 22.5 ms.

Time-Sensitive Network (TSN)

- Standardization

IEEE Standard	Description
802.1AS-rev	Time synchronization, dedicated PTP profile (IEEE 1588)
802.1Qav	Forwarding and Queuing
802.1Qbv	Traffic scheduling
802.1Qcc	Configuration

Currently implemented

Time-Sensitive Network (TSN)

- The components of TSN

OSI layer	Description
	User program
7	Standard programs: **Traffic control, OPC UA modeler, TSN configuration libraries**
6	Server/clients: **ptp4l, phc2sys, OPC UA PubSub**
5	Sockets, TCP
4	UDP, TCP, **TBS, ETF, SO_TXTIME socket option**
3	IP
2	LLDP, Ethernet driver **with TSN support, VLAN**
1	RJ45, 100baseTX, 1000baseTX, **802.1AS-rev, 802.1Qbv**

Available under an Open Source License

User provided

Linux distribution

Linux kernel

Network adapter

Time-Sensitive Network (TSN)

- The components of TSN

OSI layer	Description
	User program
7	Standard programs: **Traffic control, OPC UA modeler, TSN configuration libraries**
6	Server/clients: **ptp4l, phc2sys, OPC UA PubSub**
5	Sockets, TCP
4	UDP, TCP, **TBS, ETF, SO_TXTIME socket option**
3	IP
2	LLDP, Ethernet driver **with TSN support, VLAN**
1	RJ45, 100baseTX, 1000baseTX, **802.1AS-rev, 802.1Qbv**

Available under an Open Source License

User provided

Linux distribution

Linux kernel

Network adapter

Here

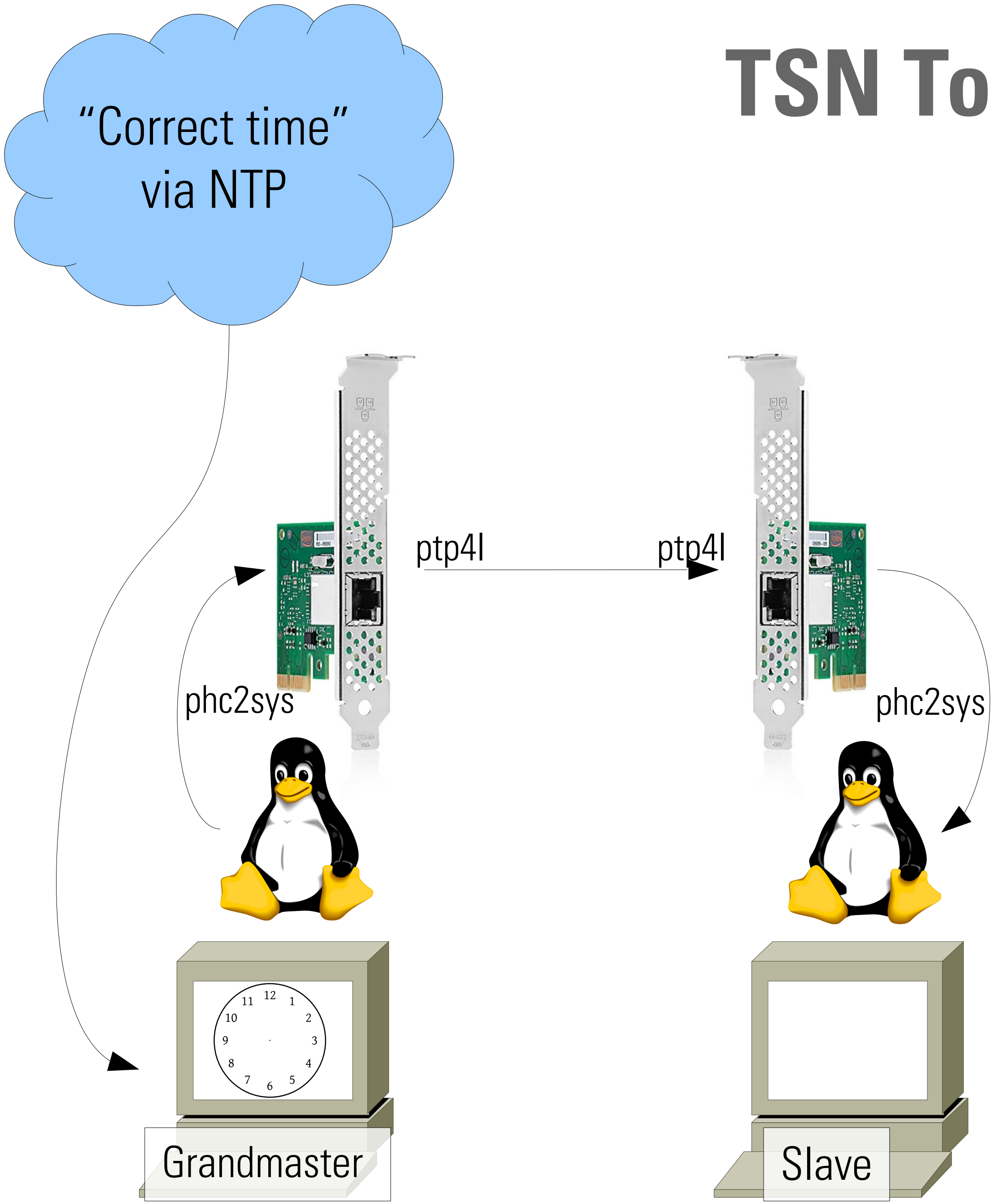
open62541.org

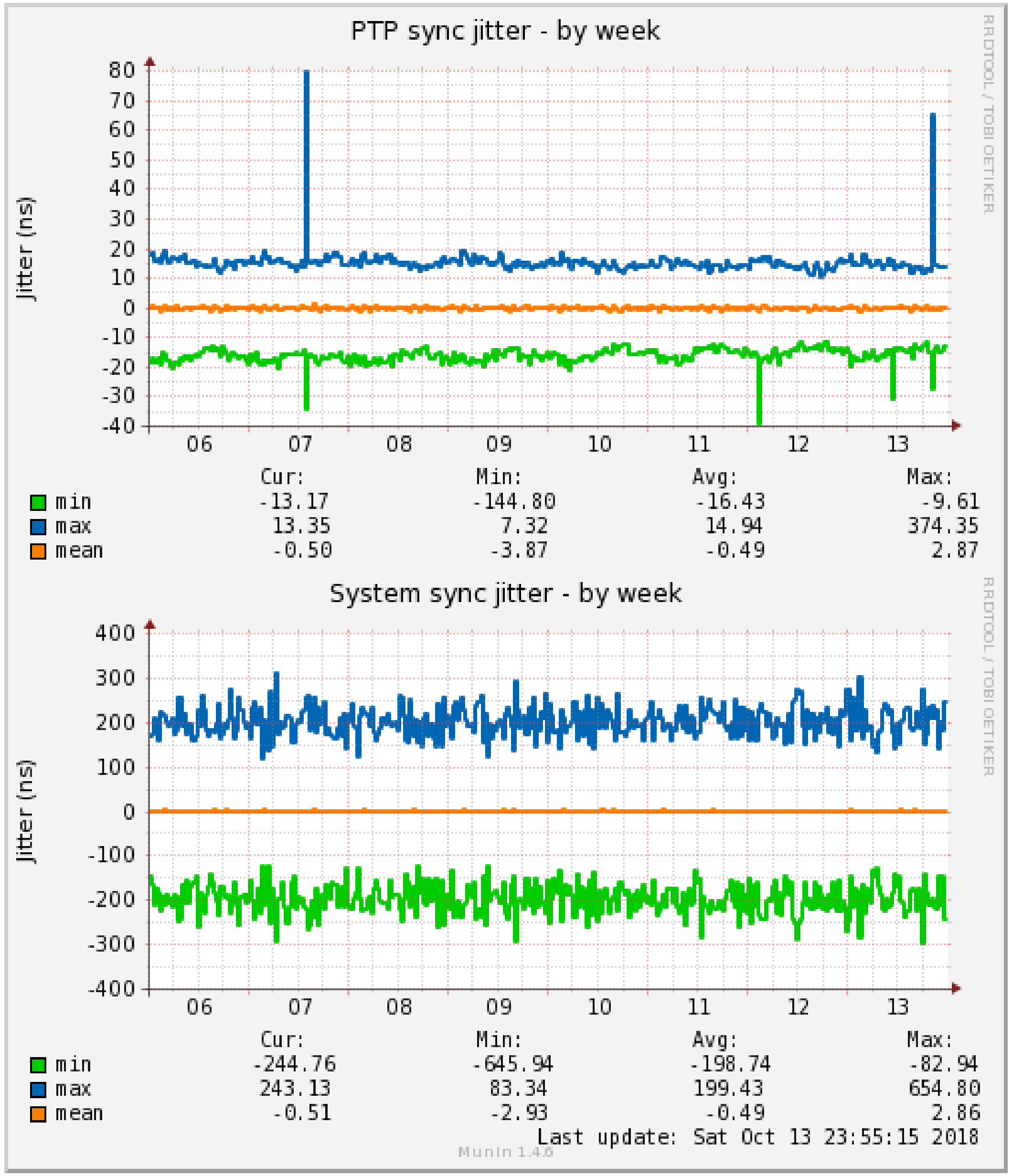
ptp4linux

ETF in Linux 4.19.X-rtY

Intel I210-T

TSN Topology





PTP sync jitter - by week

	Cur:	Min:	Avg:	Max:
🟩 min	-13.17	-144.80	-16.43	-9.61
🟦 max	13.35	7.32	14.94	374.35
🟧 mean	-0.50	-3.87	-0.49	2.87

System sync jitter - by week

	Cur:	Min:	Avg:	Max:
🟩 min	-244.76	-645.94	-198.74	-82.94
🟦 max	243.13	83.34	199.43	654.80
🟧 mean	-0.51	-2.93	-0.49	2.86

Last update: Sat Oct 13 23:55:15 2018

TSN for real-time communication

In order to use TSN not only for time synchronization, but also to send and receive network data packets deterministically, additional mechanisms are required:

- Virtual partitioning and priorization using VLAN and mapping socket priority to VLAN priority
  - This allows to send network packets with different priority over the same network connection.
  - If switches are in use, they must be configured accordingly.
- Packet transmission at a predefined point in time immediately by the network adapter hardware
  - This eliminates latency due to passing the network stack.

Configuration of a virtual LAN

Shell script

```
#!/bin/bash

for i in `seq 1 8`
do
  vconfig add $1 $i
  ifconfig $1.$i 192.168.$i.$2 up
  p=`expr $i - 1`
  for j in `seq 0 7`
  do
    vconfig set_egress_map $1.$i $j $p
  done
done

ifconfig $1 192.168.10.$2 up
```

Calling example computer no. 1

```
mkvlan.sh enp1s0 1
```

Calling example computer no. 2

```
mkvlan.sh enp1s0 2
```

Result

```
# ifconfig | grep -e enp6s0 -e "inet "
enp6s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.10.1  netmask 255.255.255.0  broadcast 192.168.10.255
enp6s0.1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
enp6s0.2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.2.1  netmask 255.255.255.0  broadcast 192.168.2.255
enp6s0.3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.3.1  netmask 255.255.255.0  broadcast 192.168.3.255
enp6s0.4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.4.1  netmask 255.255.255.0  broadcast 192.168.4.255
enp6s0.5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.5.1  netmask 255.255.255.0  broadcast 192.168.5.255
enp6s0.6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.6.1  netmask 255.255.255.0  broadcast 192.168.6.255
enp6s0.7: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.7.1  netmask 255.255.255.0  broadcast 192.168.7.255
enp6s0.8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
       inet 192.168.8.1  netmask 255.255.255.0  broadcast 192.168.8.255

# grep EGRESS /proc/net/vlan/enp6s0.1
 EGRESS priority mappings: 0:0 1:0 2:0 3:0 4:0 5:0 6:0 7:0
# grep EGRESS /proc/net/vlan/enp6s0.8
 EGRESS priority mappings: 0:7 1:7 2:7 3:7 4:7 5:7 6:7 7:7
```

Packet transmission at a predefined point in time immediately from the network adapter hardware

- In earlier kernels available as Time Based Scheduler (TBS) aka Time Based Packet Transmission (Intel provided kernel patch)
- Since Kernel 4.19 available as Earliest TxTime First (ETF) queue discipline (qdisc)
  - per-queue TxTime based scheduling

Also since Kernel 4.19 Time-Aware PRIority (TAPRIO) available

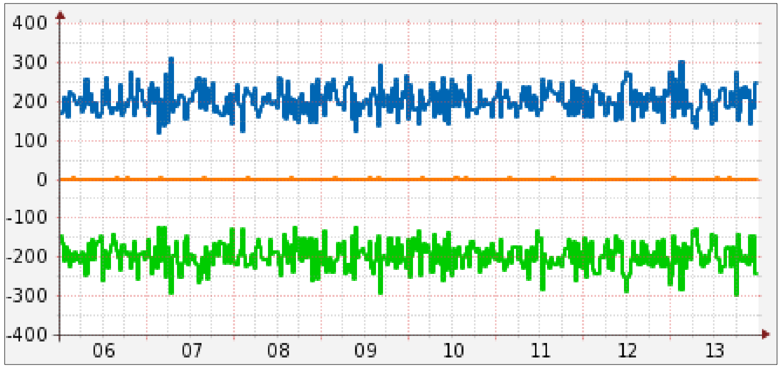
- per-port time-aware scheduling

Packet transmission at a predefined point in time immediately from the network adapter hardware (2)

- Configuration is done using the program *tc* (traffic control) via queue disciplines, for example using device $1

```
tc qdisc add dev $1 handle 8001: parent root mqprio num_tc 3 map 2 2 1 0 2 2 2
  2 2 2 2 2 2 2 2 2 queues 1@0 1@1 2@2 hw 0
tc qdisc replace dev $1 parent 8001:1 etf clockid CLOCK_TAI delta 150000
  offload
```

- Prerequisite: The network clock must be synchronized with the system clock, since packet transmission of the network adapter occurs relative to the network time, while the packet is configured from userspace based on system time.
- The achievable latency is equivalent to the jitter of the network-to-system synchronization.



Socket option SO_TXTIME, transmission time in CMSG-Block

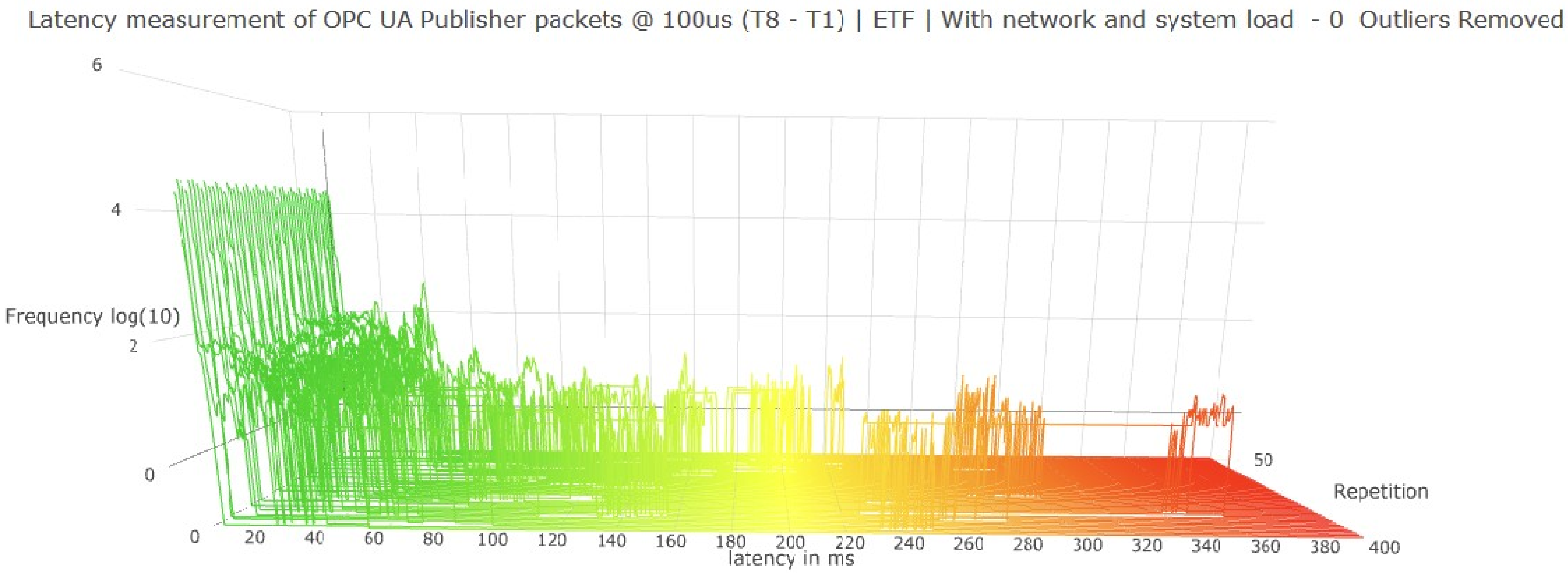
- Setting the socket option SO_TXTIME

```
setsockopt(fd, SOL_SOCKET, SO_TXTIME, &sk_txtime, sizeof(sk_txtime));
```

- Configure transmission time (tx_time) in CMSG block

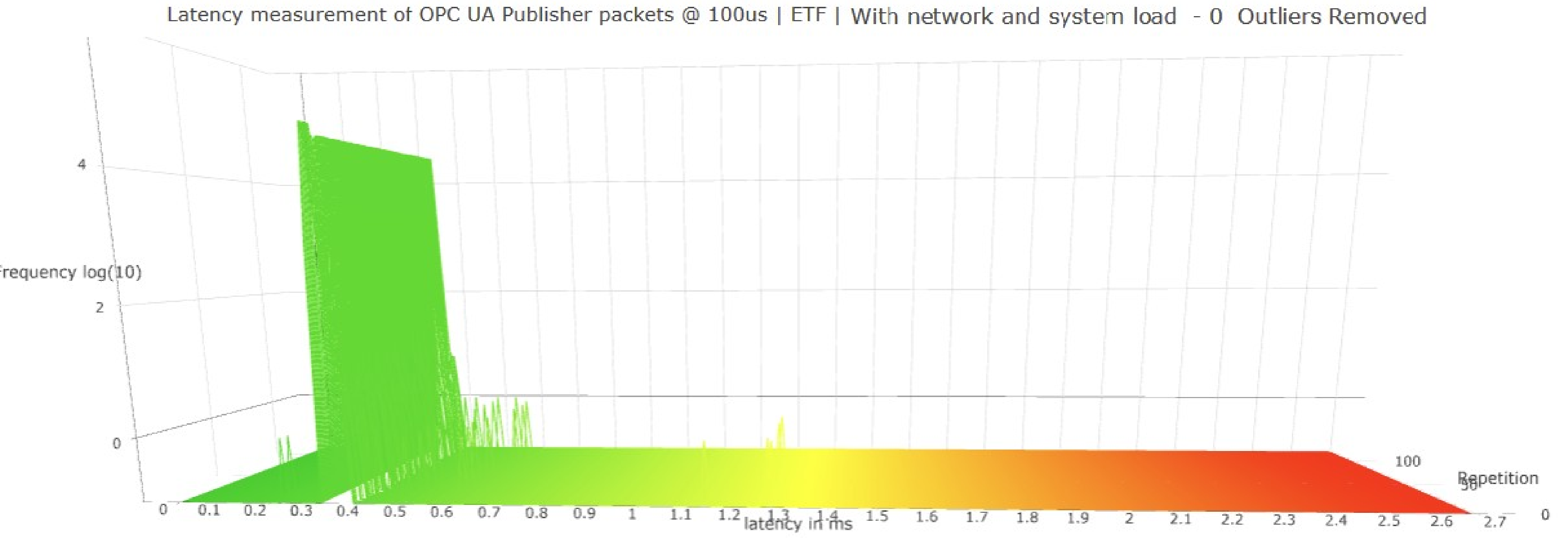
```
cmsg = CMSG_FIRSTHDR(&msg);
cmsg->cmsg_level = SOL_SOCKET;
cmsg->cmsg_type = SCM_TXTIME;
cmsg->cmsg_len = CMSG_LEN(sizeof(__u64));
*((__u64 *) CMSG_DATA(cmsg)) = tx_time;
```

TSN as transport layer for OPC UA PubSub – non-RT



Round-trip time, system under load, **no** real-time configuration

TSN as transport layer for OPC UA PubSub – RT



Round-trip time, system under load, **real-time** Configuration

Conclusion

- Currently available TSN-capable network adapters such as the Intel I210 along with suitable userspace and system software allow to create real-time Ethernet connectivity with excellent data that reach or even outperform current proprietary methods.
- Whoever will use real-time Ethernet in the future is probably well advised to setup test systems and become acquainted with this new technology.
- However, most components are still in an experimental state.
- Several important prerequisites must be fulfilled, before TSN can be used in production:
  - More components must be standardized.
  - More TSN network hardware from different manufacturers must be provided.
  - Interface between TSN network adapters and the Linux kernel must be stabilized.
  - A suitable configuration mechanism (using *snmp* or *netconf*) must be implemented and standardized.