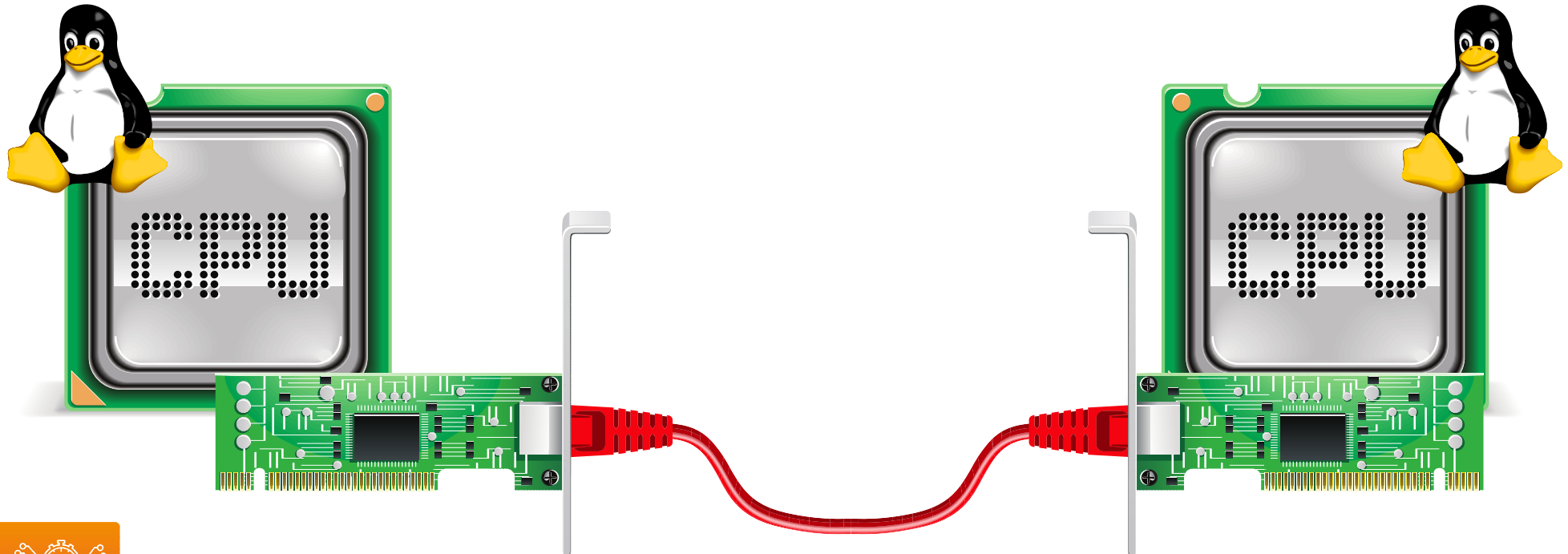
Setting up Real-time Ethernet and TSN with Linux: A 10-step guide

Jan Altenberg
Open Source Automation Development Lab (OSADL) eG

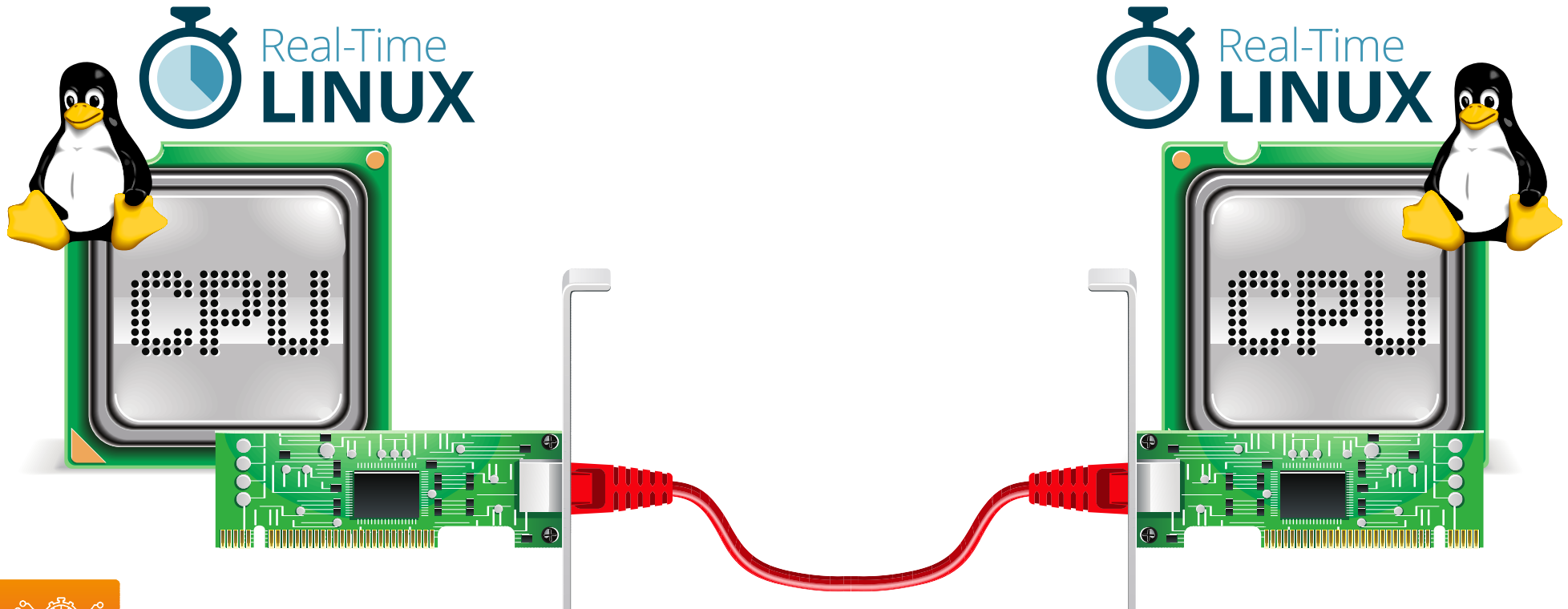
Setup



OPC UA TSN

OSADL

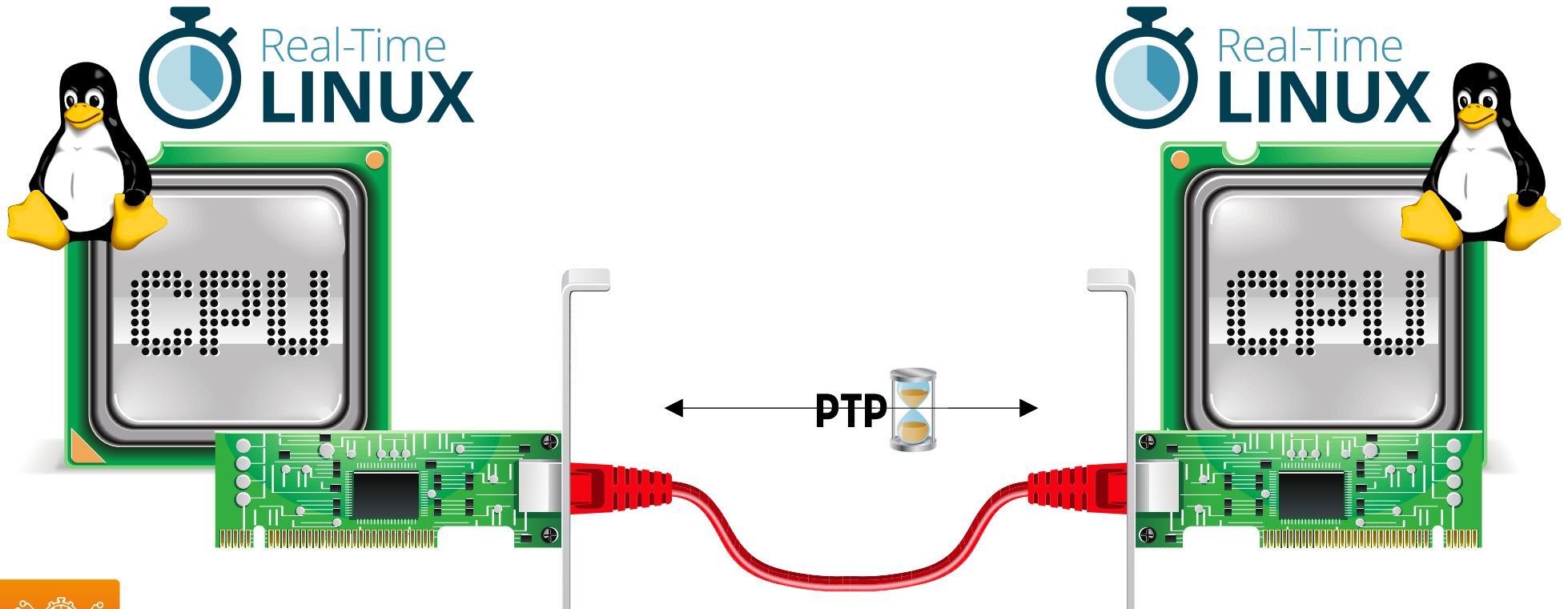
Setup



OPC UA
TSN

OSADL

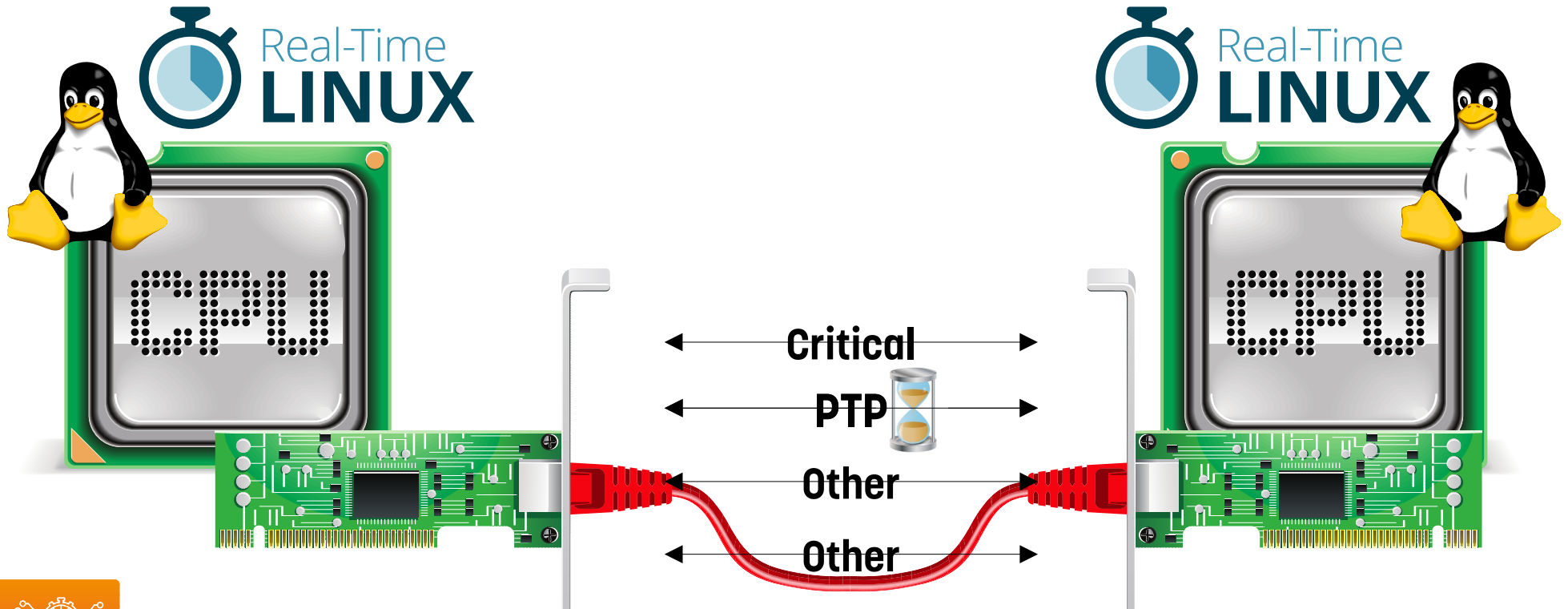
Setup



OPC UA
TSN

OSADL

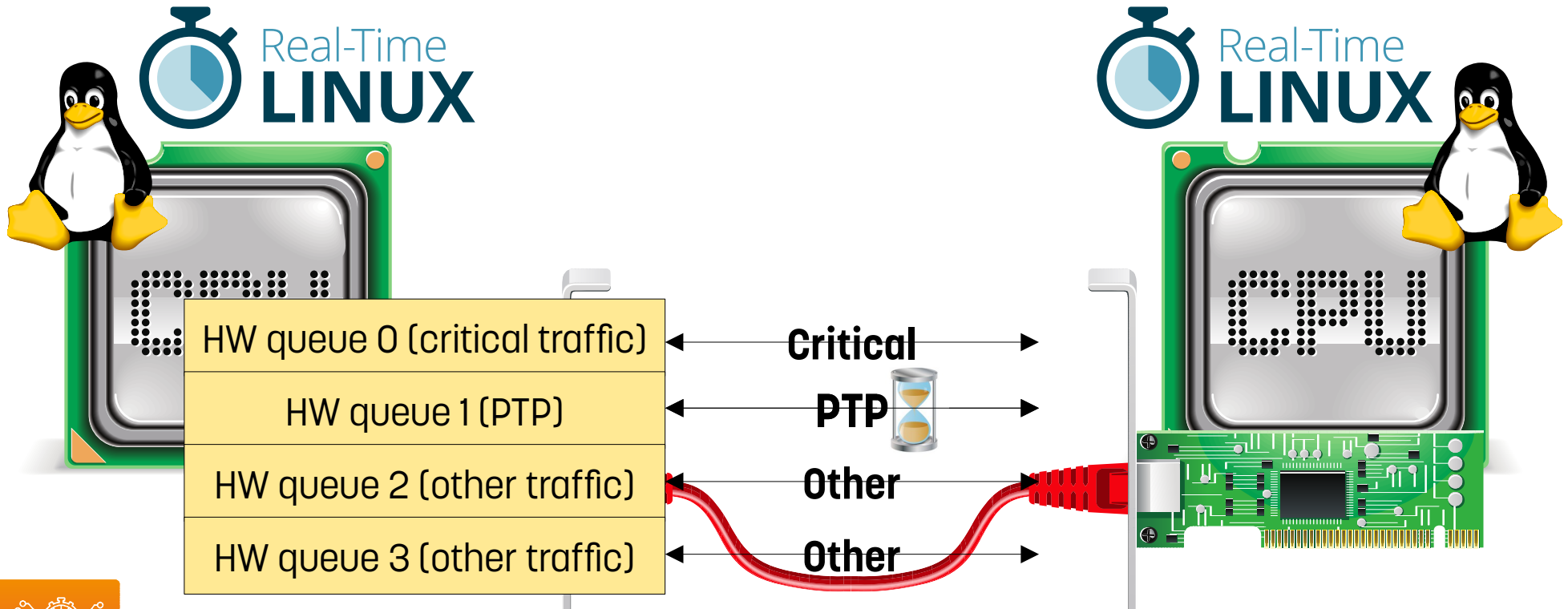
Setup



OPC UA
TSN

OSADL

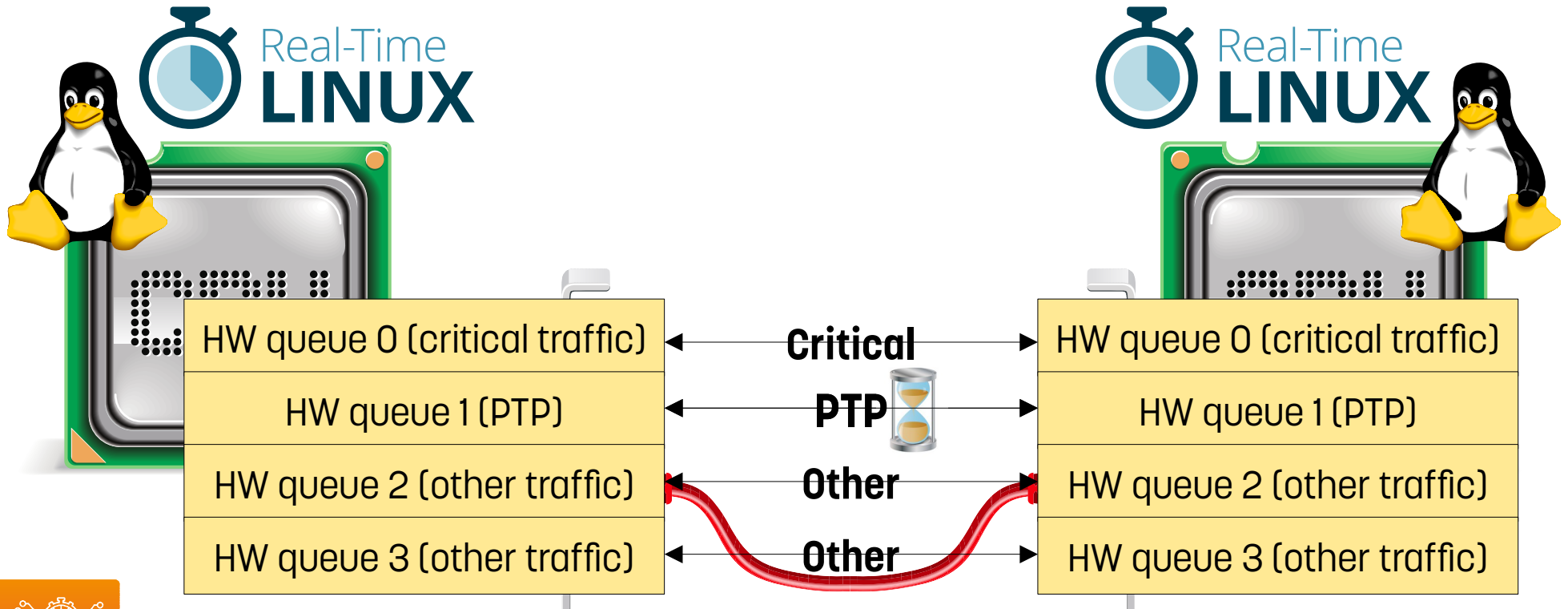
Setup



OPC UA TSN

OSADL

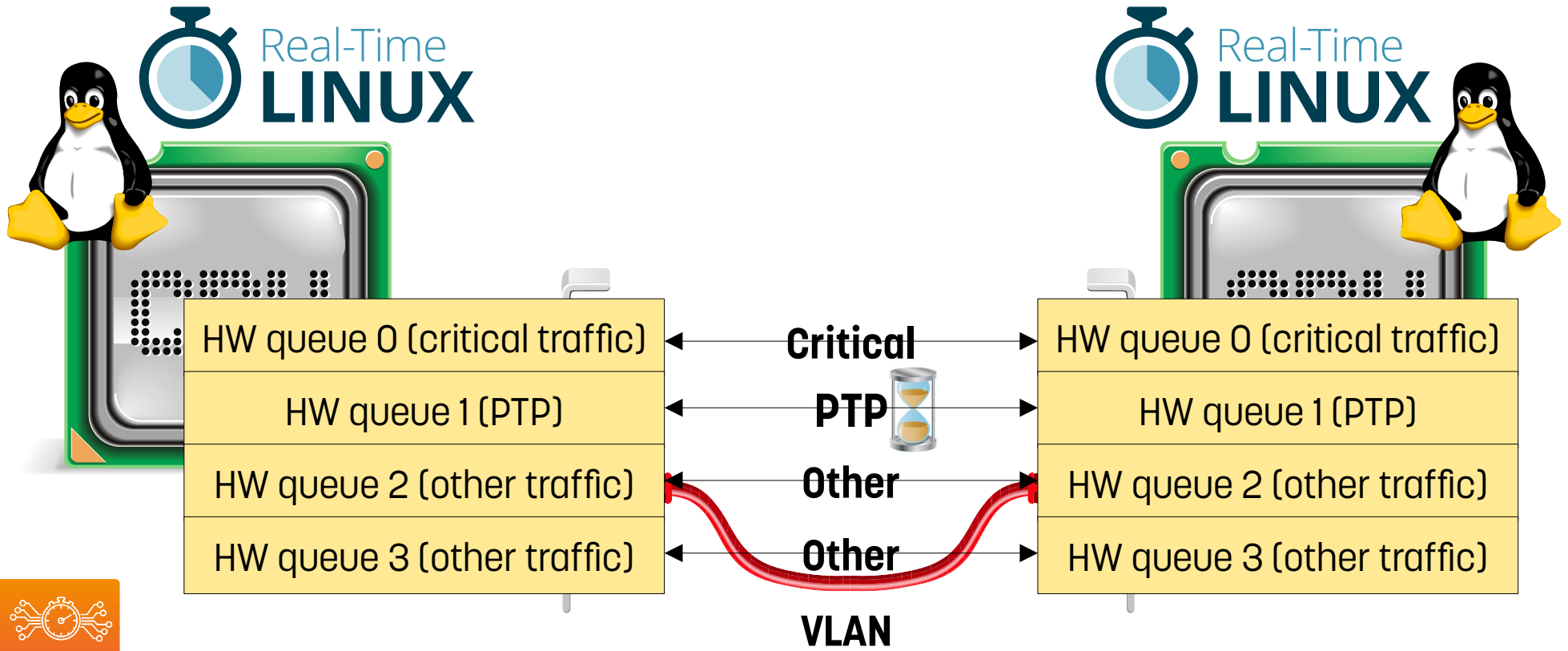
Setup



OPC UA TSN

OSADL

Setup



OPC UA TSN

OSADL

Step 1

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Kernel commandline

```
isolcpus=2,3 rcu_nocbs=2,3 nohz_full=2,3
irqaffinity=0
noefi
processor.max_cstate=0 intel_idle.max_cstate=0
i915.enable_psr=0 i915.enable_rc6=0
i915.enable_dc=0 i915.disable_power_well=0
i915.enable_guc=2
```

Kernel commandline

```
isolcpus=2,3 rcu_nocbs=2,3 nohz_full=2,3
irqaffinity=0
noefi
processor.max_cstate=0
i915.enable_rc6=0
i915.enable_dc=0 i915.disable_power_well=0
i915.enable_guc=2
```

Isolate CPU 2 and 3 for real-time applications
Route interrupts to CPU 0

Kernel commandline

```
isolcpus=2,3 rcu_nocbs=2,3 nohz_full=2,3
irqaffinity=0
noefi
processor.max_cstate=0
i915.enable_rc6=0
i915.enable_dc=0 i915.disable_power_well=0
i915.enable_guc=2
```

Don’t allow access to EFI variable from userspace
(this is a well known cause for high latencies)

Kernel commandline

```
isolcpus=2,3 rcu_nocbs=2,3 nohz_full=2,3
irqaffinity=0
noefi
processor.max_cstate=0 intel_idle.max_cstate=0
i915.enable_psr=0 i915.enable_rc6=0
i915.enable_dc=0 i915.disable_guc=2
```

Kernel commandline

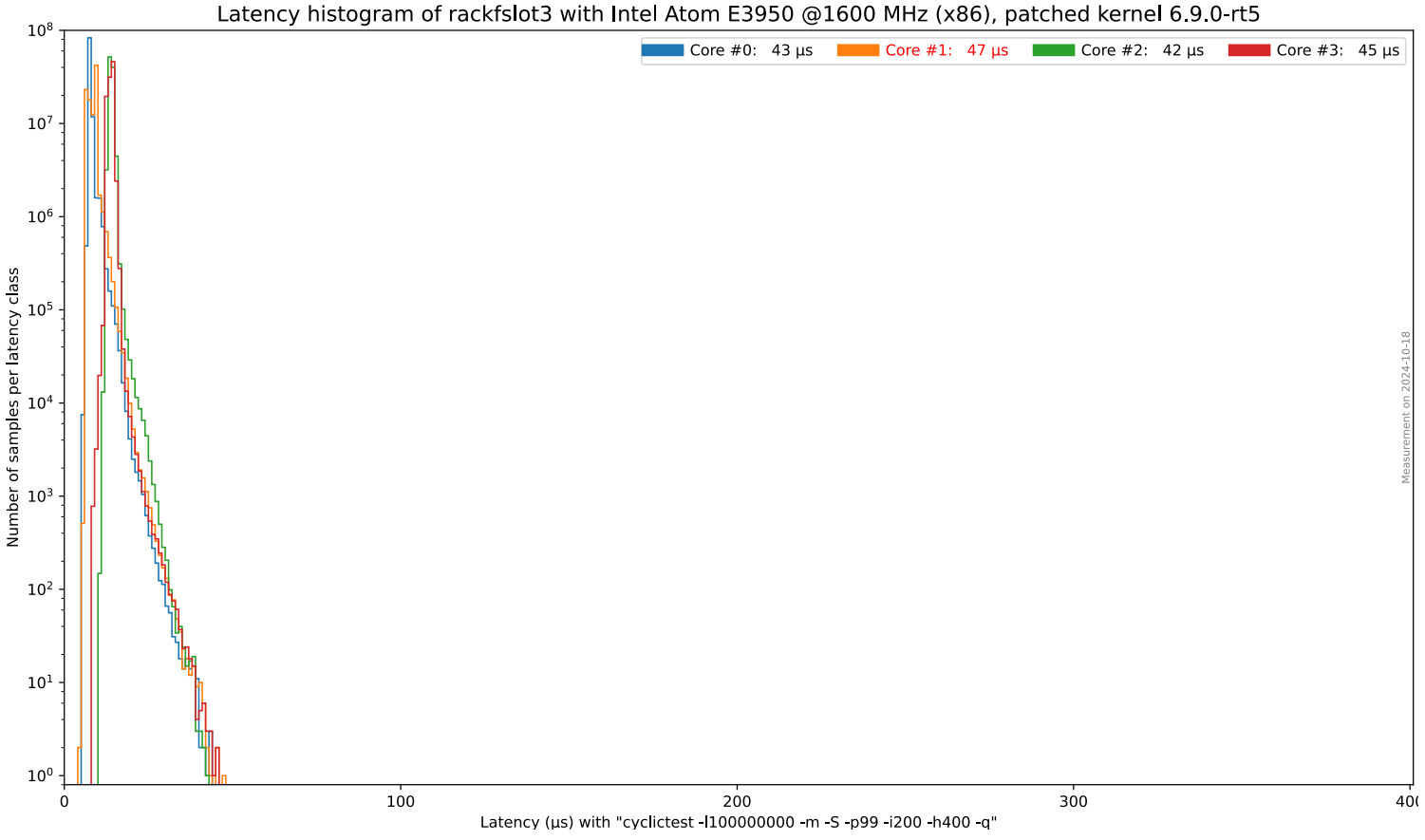
```
isolcpus=2,3 rcu_nocbs=2
irqaffinity=0
noefi
processor.max_cstate=0 intel_idle.max_cstate=0
i915.enable_psr=0 i915.enable_rc6=0
i915.enable_dc=0 i915.disable_power_well=0
i915.enable_guc=2
```

Configure the i915 graphics driver for optimal real-time behavior

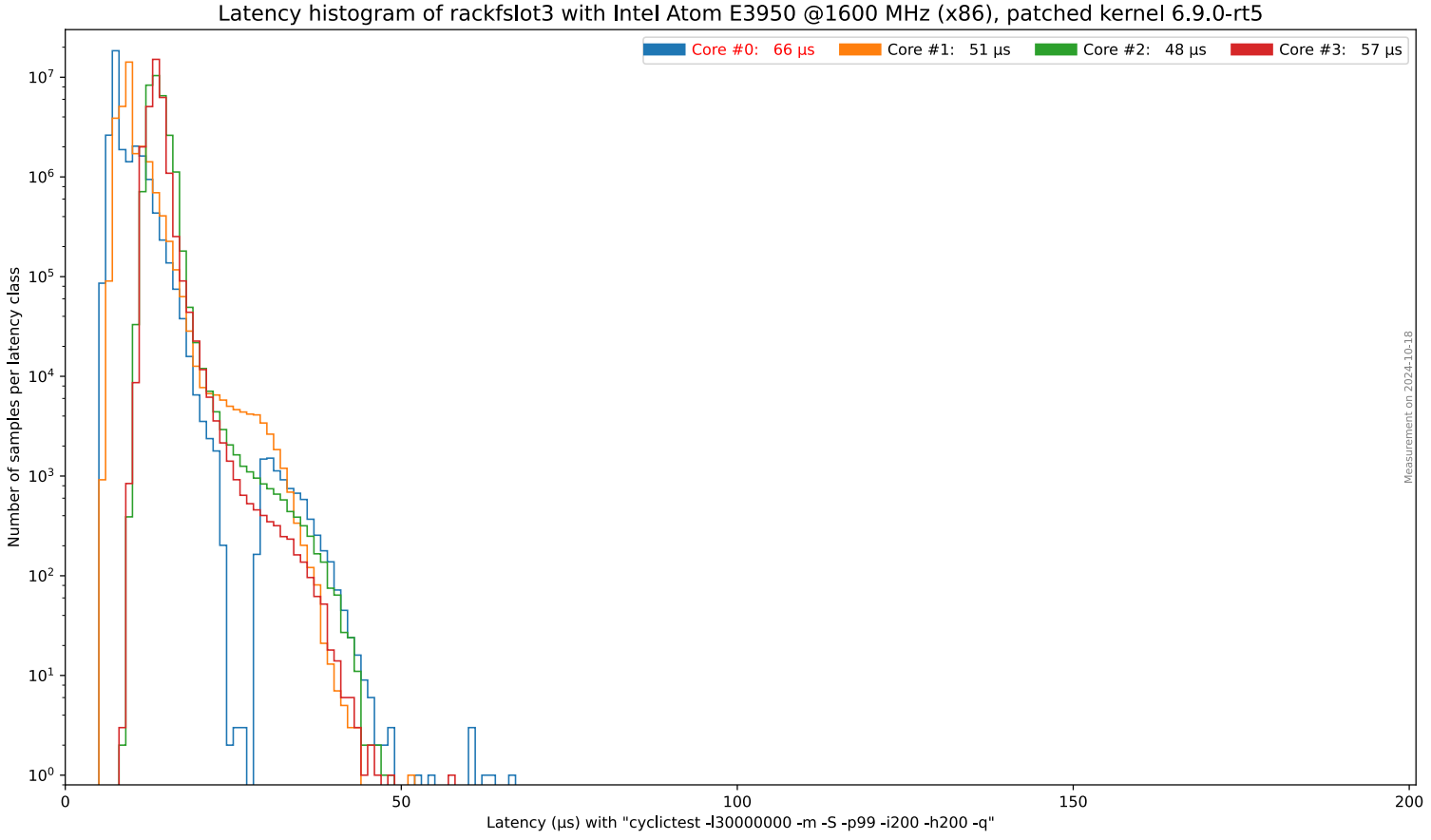
CPU frequency scaling

```
for CPU in $(seq 0 3)
do
  echo performance > \
     /sys/devices/system/cpu/cpufreq/policy${CPU}/scaling_governor
done
```

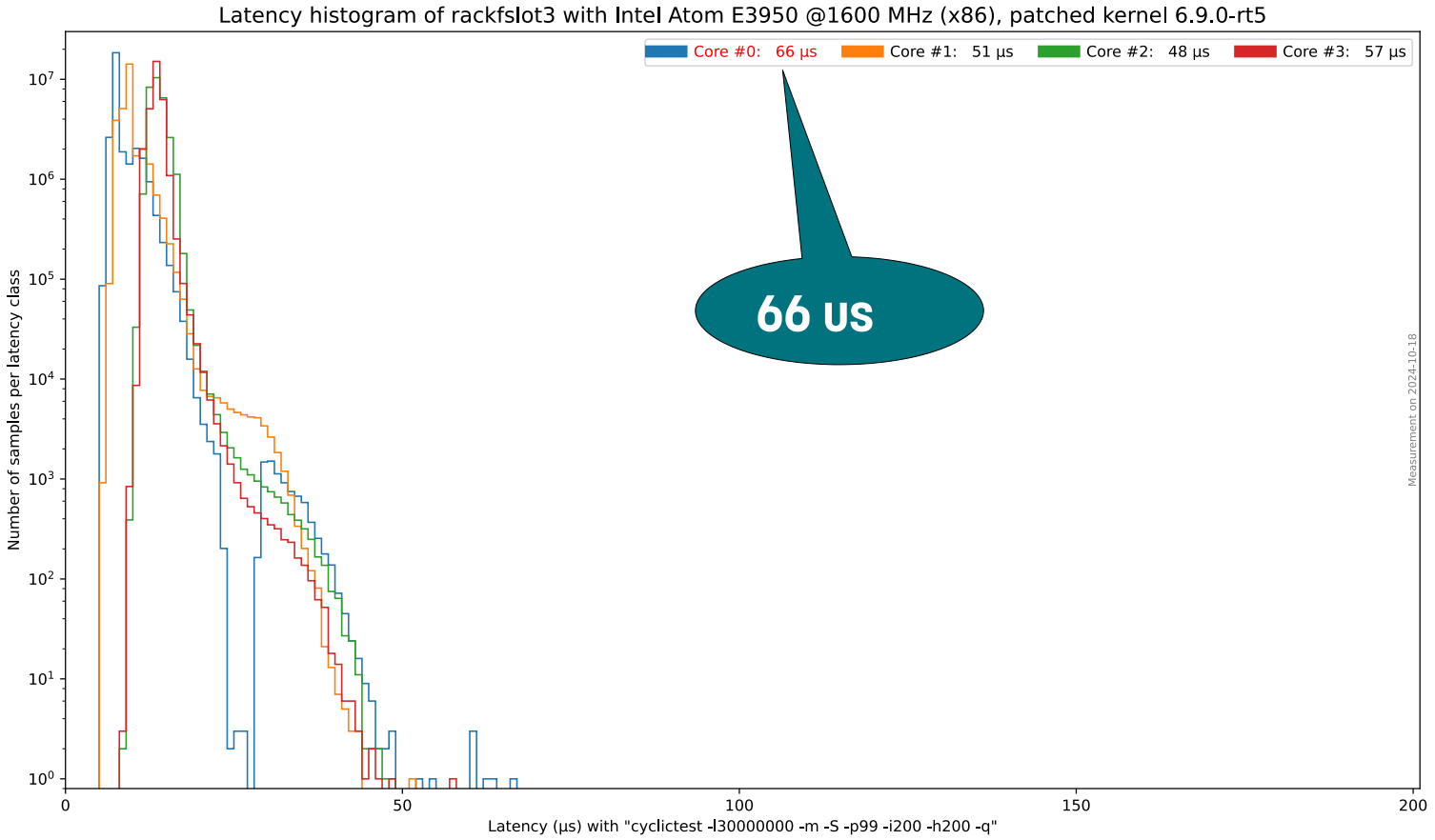
Evaluate the real-time behavior (idle)



Evaluate the real-time behavior (load)



Evaluate the real-time behavior (load)

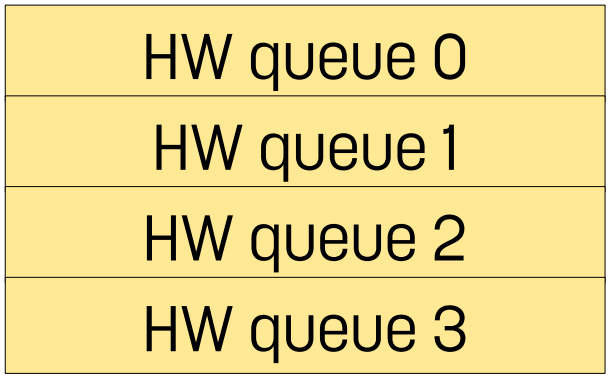


Step 2

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Set up 4 combined HW queues

```
$ ethtool -L ${INTERFACE} combined 4
$ ethtool -G ${INTERFACE} rx 4096 tx 4096
```



Disable energy efficiency

```
$ ethtool --set-eee ${INTERFACE} eee off
```

Step 3

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Queueing disciplines: mqprio

- Define the number of traffic classes
- Map socket priorities to traffic classes
- Assign the traffic classes to specific hardware queues

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

The number of traffic classes to be used

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Mapping the socket priorities to a traffic class

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **0**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **1**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **2**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **15**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **0** maps to traffic class **2**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **2** maps to traffic class **1**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Priority **3** maps to traffic class **0**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Map the traffic classes into hardware queues (4 queues are being used here)

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Traffic class **0**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

Traffic class **1**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

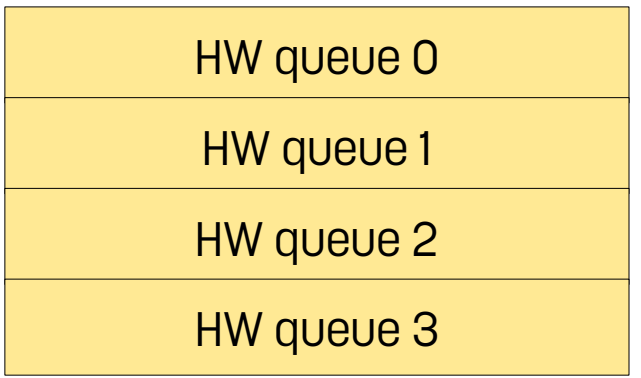
```
tc qdisc add dev eth1 handle 100: parent root mqprio \
num_tc 3 \
map 2 2 1 0 2 2 2 2 2 2 2 2 2 2 2 2 \
queues 1@0 1@1 2@2 \
hw 0
```

Traffic class **2**

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

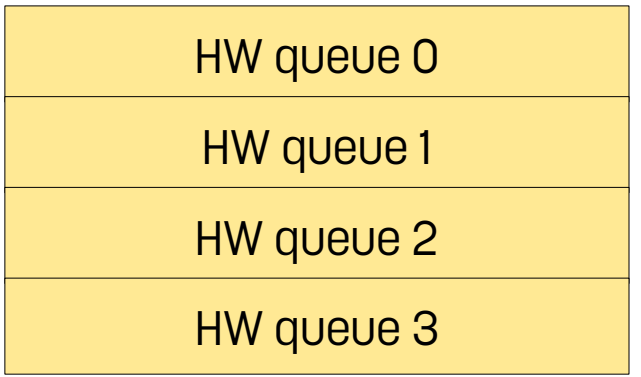


Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

TC0 goes into **1** queue

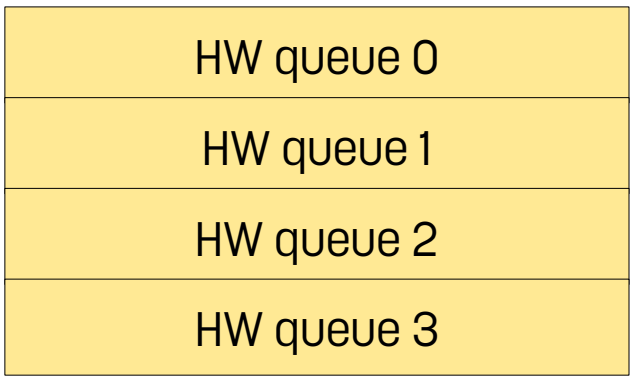


Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

TC0 goes into 1 queue starting from queue **0**



Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

TC0 goes into **1** queue starting from queue **0**

HW queue 0
HW queue 1
HW queue 2
HW queue 3

OPC UA TSN

OSADL

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

TC1 goes into **1** queue starting from queue **1**

HW queue 0
HW queue 1
HW queue 2
HW queue 3

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

TC2 goes into **2** queues starting from queue **2**

HW queue 0
HW queue 1
HW queue 2
HW queue 3

Queueing disciplines: mqprio

Manage qdiscs using the application **tc**:

```
tc qdisc add dev eth1 handle 100: 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
```

0: No hardware offloading
1: Hardware offloading

```
# Check for hardware support
ethtool -k eth1 | grep hw-tc-offload
hw-tc-offload: on
```

Step 4

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Threaded NAPI

```
$ echo 1 > /sys/class/net/${INTERFACE}/threaded
```

IRQ routing

```
HWQUEUE=0
for IRQ_NUM in $(cat /proc/interrupts | grep ${INTERFACE} | cut -f1 -d:)
do
    if [ ${HWQUEUE} -lt 3 ]
    then
        echo ${ETH_IRQTHREAD_MASK} > /proc/irq/${IRQ_NUM}/smp_affinity
    fi
    let HWQUEUE=${HWQUEUE}+1
done
```

IRQ-thread priorities

```
HWQUEUE=0
for PID in $(ps aux | grep ${INTERFACE} | grep irq | tr -s ' ' | cut -f2 -d' ')
do
        if [ ${HWQUEUE} -ge 1 -a ${HWQUEUE} -lt 3 ]
        then
                chrt -f -p ${ETH_IRQTHREAD_PRIO} ${PID}
        else
                chrt -f -p 70 ${PID}
        fi
        let HWQUEUE=${HWQUEUE}+1
done
```

NAPI-thread priorities

```
HWQUEUE=4
NAPITHREADS=$(ps aux | grep napi | grep ${INTERFACE} | awk '{ print $2; }')
for THREAD in ${NAPITHREADS}
do
        if [ ${HWQUEUE} -le 2 ]
        then
                chrt -p -f ${NAPI_THREAD_PRIO} $THREAD
                taskset -pc ${NAPI_CPU_AFFINITY} $THREAD
        else
                chrt -p -f 70 $THREAD
                taskset -pc 0 $THREAD
        fi
        let HWQUEUE=${HWQUEUE}-1
done
```

CPU pinning and priorities

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND	
[...]												
13874	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/3:1H-kblo	
2055005	root	-76	0	238944	46456	2304	S	50.0	1.2	0:56.79	pubsub_TSN_publ	
2055008	root	-76	0	238944	46456	2304	S	0.0	1.2	0:11.79	pubsub_TSN_publ	
[...]												
666	**root**	**-71**	**0**	**0**	**0**	**0**	**S**	**0.0**	**0.0**	**0:07.24**	**irq/132-enp4s0**	
667	**root**	**-99**	**0**	**0**	**0**	**0**	**S**	**0.0**	**0.0**	**25:06.80**	**irq/133-enp4s0-TxRx-0**	**2**
668	**root**	**-99**	**0**	**0**	**0**	**0**	**S**	**0.0**	**0.0**	**0:31.43**	**irq/134-enp4s0-TxRx-1**	**2**
793	**root**	**-99**	**0**	**0**	**0**	**0**	**S**	**0.0**	**0.0**	**0:30.76**	**napi/enp4s0-8198**	**2**
794	**root**	**-99**	**0**	**0**	**0**	**0**	**S**	**50.0**	**0.0**	**27:21.56**	**napi/enp4s0-8197**	**2**
1975955	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/2:1H	2
2055006	root	-79	0	238944	46456	2304	S	0.0	1.2	0:23.29	pubsub_TSN_publ	2
2055007	root	-82	0	238944	46456	2304	S	0.0	1.2	0:21.59	pubsub_TSN_publ	2

HW queue 0 (critical traffic)

HW queue 1 (PTP)

HW queue 2 (other traffic)

HW queue 3 (other traffic)

OPC UA TSN

OSADL

Step 5

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \

clockid CLOCK_TAI \

delta 66000 \

offload
```

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \
clockid CLOCK_TAI \
delta 66000 \
offload
```

handle of the parent qdisc (defined before)

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \
clockid CLOCK_TAI \
delta 66000 \
offload
```

Traffic class:
1: TC0 2: TC1 3: TC2

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \
clockid CLOCK_TAI \
delta 66000 \
offload
```

Traffic class:
1: TC0 2: TC1 3: TC2

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \
clockid CLOCK_TAI \
delta 66000 \
offload
```

The clock to be used for scheduling events and measuring the time

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \
clockid CLOCK_TAI \
delta 66000 \
offload
```

Can be used to take the scheduler latency into account

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \

clockid CLOCK_TAI \

delta 66000 \

offload
```

Can be used to take the scheduler latency into account
Next wake up = next tx time - **delta**

ETF

Manage qdiscs using the application **to**:

```
tc qdisc
clockid CLOCK_TA
delta 66000 \
offload
```

Use cyclictest to determine the latency

Can be used to take the scheduler latency into account
Next wake up = next tx time - **delta**

ETF

Manage qdiscs using the application **tc**:

```
tc qdisc replace dev eth1 parent 100:1 etf \

clockid CLOCK_TAI \

delta 66000 \
```

`offload` Use the “Launch Time” feature of the NIC (disabled by default)

Step 6

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

VLAN

```
auto enp4s0.8
iface enp4s0.8 inet static
address 192.168.4.1
netmask 255.255.255.0
```

VLAN

```
auto enp4s0.8
iface enp4s0.8 inet static
address 192.168.4.2
netmask 255.255.255.0
```

Step 7

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Setup virtual networks (VLAN)

- Using qdiscs traffic classes are mapped to hardware queues and outgoing traffic can be scheduled.
- In a next step outgoing packets have to be “marked”, so the different traffic classes can be differentiated on the network.
- This is done using the Port Control Protocol (PCP) field.
- Socket priorities can be mapped to the PCP field using VLANs.

Setup virtual networks (VLAN)

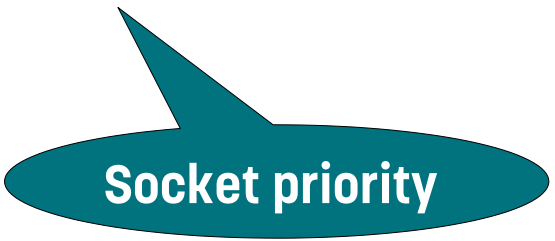
Map the traffic classes to the PCP field:

```
# ip link set enp4s0.8 type vlan \
egress 2:2 3:3
```

Setup virtual networks (VLAN)

Map the traffic classes to the PCP field:

```
# ip link set enp4s0.8 type vlan \
egress 2:2 3:3
```



Setup virtual networks (VLAN)

Map the traffic classes to the PCP field:

```
# ip link set enp4s0.8 type vlan \
egress 2:2 3:3
```

Step 8

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

Filtering incoming traffic

```
ethtool -K ${INTERFACE} ntuple on

# PCP 3 -> HW queue 0
ethtool -N ${INTERFACE} flow-type ether vlan 0x6000 m 0x1fff action 0
# PCP 2 -> HW queue 1
ethtool -N ${INTERFACE} flow-type ether vlan 0x4000 m 0x1fff action 1

# The rest goes to HW queue 2 and 3
ethtool -N ${INTERFACE} flow-type ether vlan 0xc000 m 0x1fff action 2
[...]
ethtool -N ${INTERFACE} flow-type ether vlan 0xe000 m 0x1fff action 3
```

Filtering incoming traffic

```
ethtool -K ${INTERFACE} ntuple on

# PCP 3 -> HW queue 0
ethtool -N ${INTERFACE} flow-type ether vlan 0x6000 m 0x1fff action 0
# PCP 2 -> HW queue 1
ethtool -N ${INTERFACE} flow-type ether vlan 0x4000 m 0x1fff action 1

# The rest goes to HW queue 2 and 3
ethtool -N ${INTERFACE} flow-type ether vlan 0xc000 m 0x1fff action 2
[...]
ethtool -N ${INTERFACE} flow-type ether vlan 0xe000 m 0x1fff action 3
```

PCP field 3 goes to HW queue 0

Filtering incoming traffic

```
ethtool -K ${INTERFACE} ntuple on

# PCP 3 -> HW queue 0
ethtool -N ${INTERFACE} flow-type ether vlan 0x6000 m 0x1fff action 0
# PCP 2 -> HW queue 1
ethtool -N ${INTERFACE} flow-type ether vlan 0x4000 m 0x1fff action 1

# The rest goes to HW queue 2
ethtool -N ${INTERFACE} flow-type ether vlan 0x2000 m 0x1fff action 2
[...]
ethtool -N ${INTERFACE} flow-type ether vlan 0xe000 m 0x1fff action 3
```

PCP field 2 goes to HW queue 1

Filtering incoming traffic

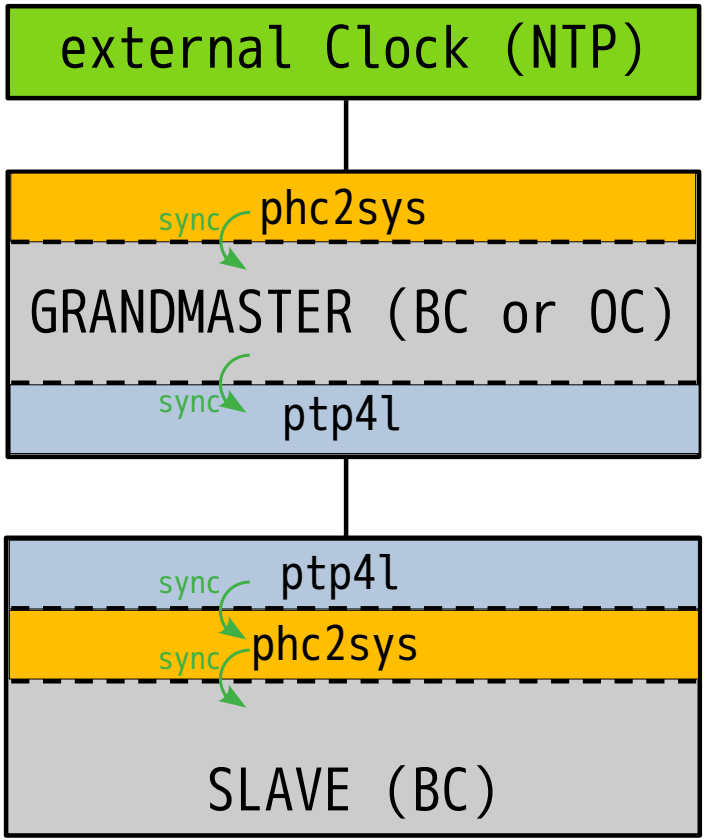
```
ethtool -K ${INTERFACE} ntuple on

# PCP 3 -> HW queue 0
ethtool -N ${INTERFACE} flow-type ether vlan 0x6000 m 0x1fff action 0
# PCP 2 -> HW queue 1
ethtool -N ${INTERFACE} flow-type ether vlan 0x4000 m 0x1fff action 1

# The rest goes to HW queue 2 and 3
ethtool -N ${INTERFACE} flow-type ether vlan 0xc000 m 0x1fff action 2
[...]
ethtool -N ${INTERFACE} f                    0x1fff action 3
```

Other traffic goes to HW queues 2 and 3

Step 9

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

System setup



- Exactly one time provider is needed per network (the “grandmaster”). This grandmaster knows the time, since it
  - has a built-in highly accurate clock,
  - obtains the correct time from an external trusted source, or
  - does both.
- As many time consumers (“slaves”) as required may be installed.
- Slaves may be connected peer-to-peer or via switches.

OPC UA TSN

OSADL

PTP synchronization on the Grandmaster

Send network time information via PTP

```
$ ptp4l -A -i enp1s0  --socket_priority=2 -f /etc/linuxptp/ptp4l.conf \
  -2 -mq >/var/log/ptp4l.out 2>/var/log/ptp4l.err
```

Synchronize system time

```
$ phc2sys -c enp1s0 -s CLOCK_REALTIME \
-w -mq >/var/log/phc2sys.out 2>/var/log/phc2sys.err
```

PTP synchronization on the Grandmaster

Send network time information via PTP

```
$ ptp4l -A -i enp1s0  --socket_priority=2 -f /etc/linuxptp/ptp4l.conf \
  -2 -mq >/var/log/ptp4l.out 2>/var/log/ptp4l.err
```

Synchronize system time

```
$ phc2sys -c enp1s0 -s CLOCK_REALTIME \
-w -mq >/var/log/phc2sys.out 2>/var/log/phc2sys.err
```

HW queue 0 (critical traffic)
HW queue 1 (PTP)
HW queue 2 (other traffic)
HW queue 3 (other traffic)

OPC UA TSN

OSADL

PTP synchronization on the Grandmaster

Send network time information via PTP

```
$ ptp4l -A -i enp1s0  --socket_priority=2 -f /etc/linuxptp/ptp4l.conf \
  -2 -mq >/var/log/ptp4l.out 2>/var/log/ptp4l.err
```

Select delay mechanism automatically

Synchronize system time

```
$ phc2sys -c enp1s0 -s CLOCK_REALTIME \
-w -mq >/var/log/phc2sys.out 2>/var/log/phc2sys.err
```

OPC UA TSN

OSADL

PTP time synchronization on the slave

Receive network time information via PTP

```
ptp4l -A -i enp1s0 --socket_priority=2 -2 -mq -s \
-f /etc/linuxptp/ptp4l.conf >/var/log/ptp4l.out 2>/var/log/ptp4l.err &
```

Synchronize system time

```
phc2sys -s enp1s0 -w -mq >/var/log/phc2sys.out 2>/var/log/phc2sys.err &
```

PTP time synchronization on the slave

Receive network time information via PTP

```
ptp4l -A -i enp1s0 --socket_priority=2 -2 -mq -s \
-f /etc/linuxptp/ptp4l.conf >/var/log/ptp4l.out 2>/var/log/ptp4l.err &
```

run as slave

Synchronize system time

```
phc2sys -s enp1s0 -w -mq >/var/log/phc2sys.out 2>/var/log/phc2sys.err &
```

OPC UA TSN

OSADL

Step 10

- Real-time operating system setup (PREEMPT_RT)
- Hardware configuration
- Traffic classes
- System level configuration
- Queuing disciplines (QDISCs)
- Networking setup
- VLAN mapping
- Routing incoming traffic
- Time synchronization
- Evaluation

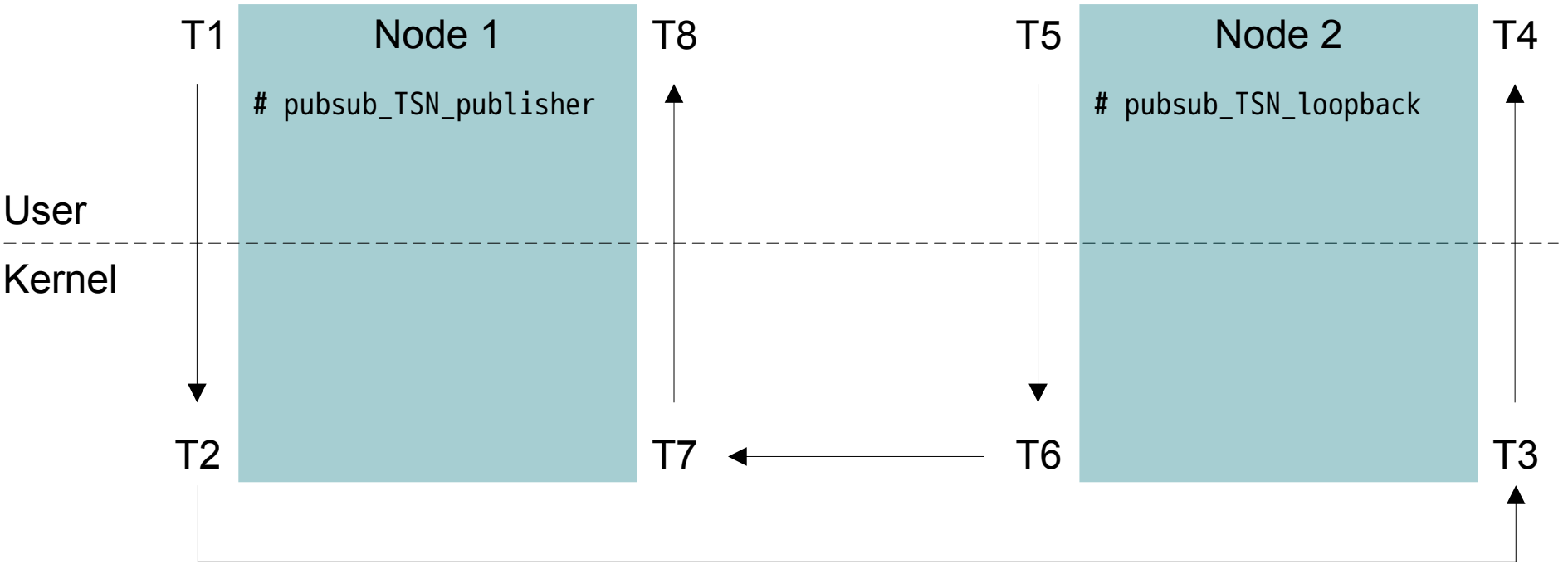
Open62541: PubSub real-time example, build configuration

```
# cmake -DUA_ENABLE_PUBSUB=ON \
-DUA_ENABLE_PUBSUB_ETH_UADP=ON \
-DUA_BUILD_EXAMPLES=ON \
-DUA_ENABLE_MALLOC_SINGLETON=ON \
-DUA_ENABLE_PUBSUB_BUFMALLOC=ON \
-DUA_ENABLE_IMMUTABLE_NODES=ON
# make
```

Open62541: PubSub real-time example, build configuration

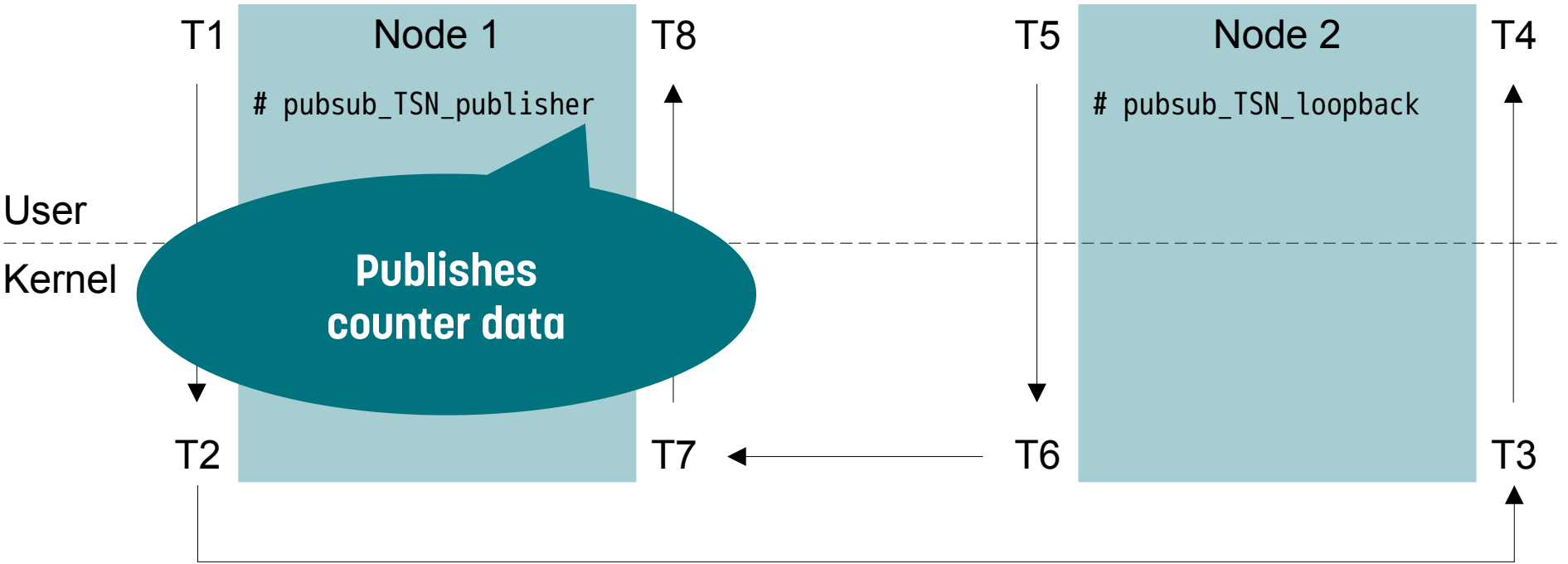
```
# ls -1 bin/examples/pubsub_*
[...]
bin/examples/pubsub_TSN_loopback
bin/examples/pubsub_TSN_loopback_single_thread
bin/examples/pubsub_TSN_publisher
[...]
```

Open62541: PubSub real-time example



T8 – T1 = Round trip time

Open62541: PubSub real-time example

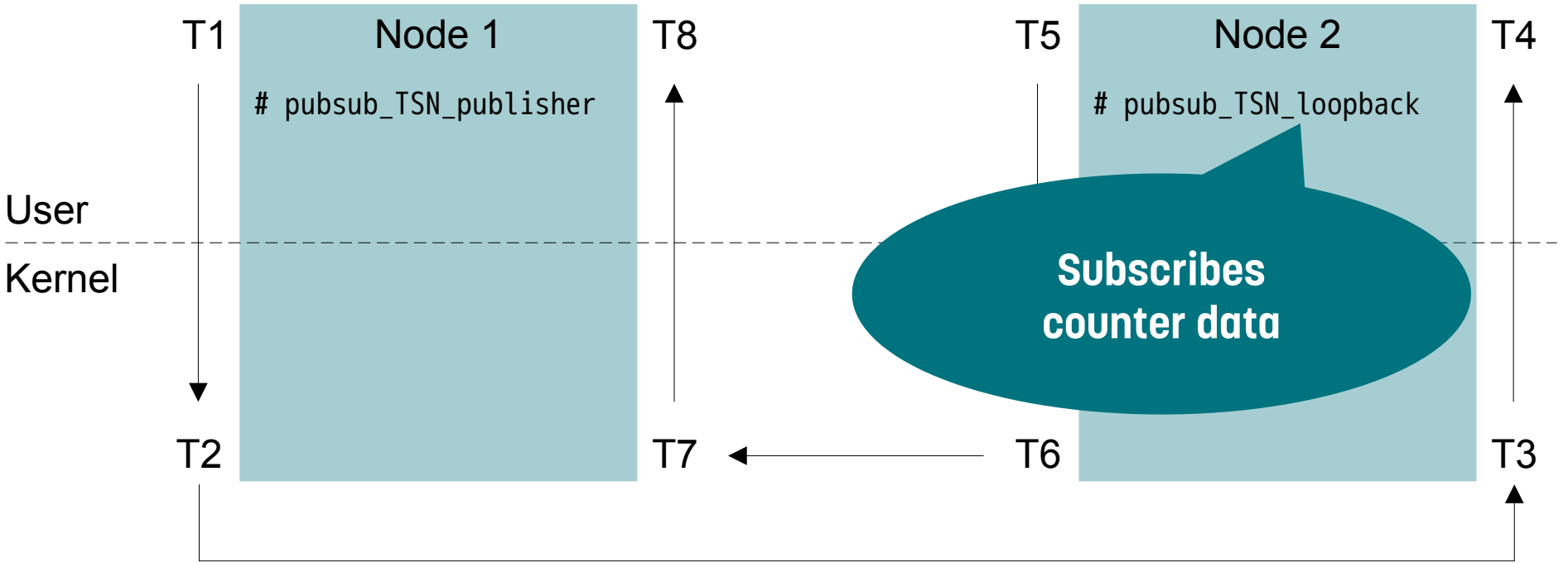


T8 – T1 = Round trip time

OPC UA TSN

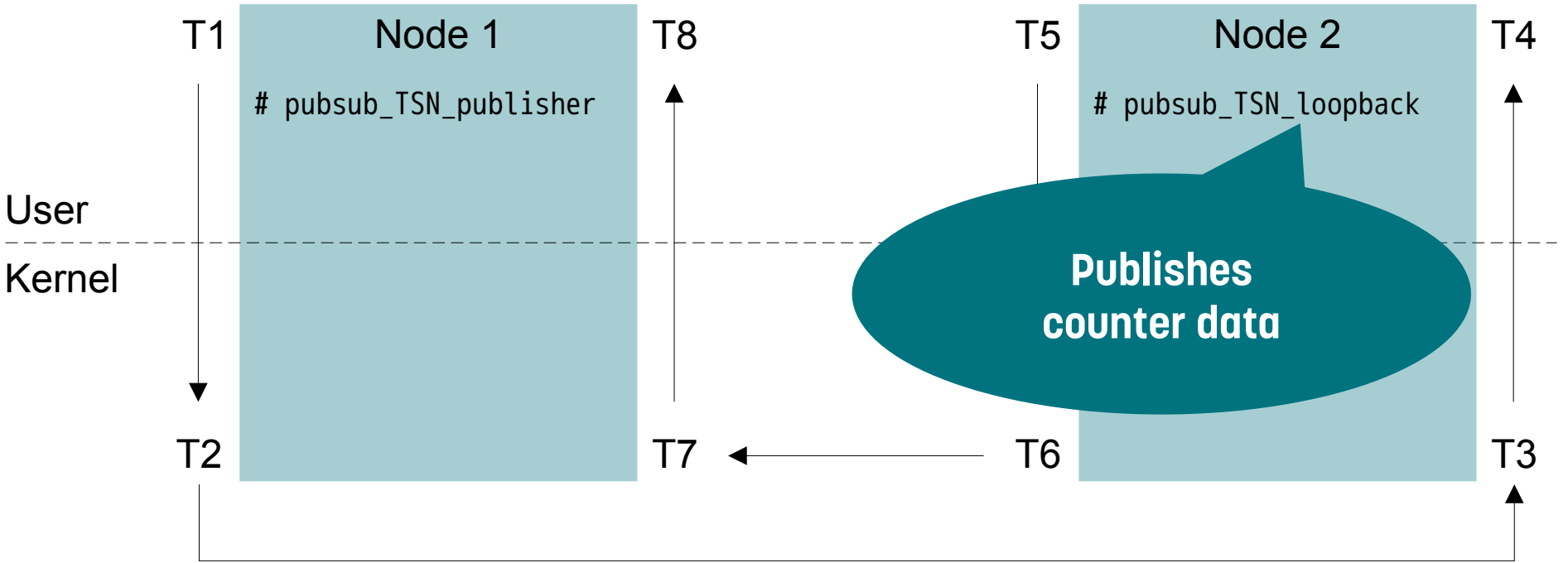
OSADL

Open62541: PubSub real-time example



T8 – T1 = Round trip time

Open62541: PubSub real-time example

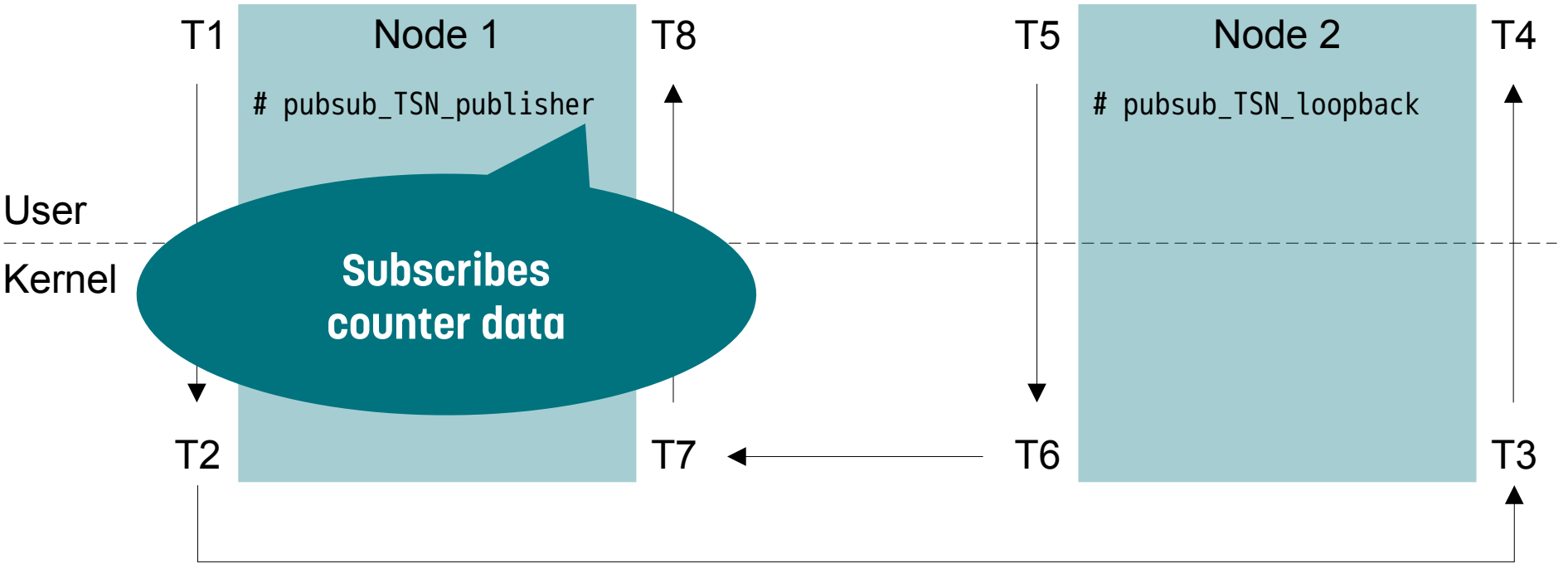


T8 – T1 = Round trip time

OPC UA TSN

OSADL

Open62541: PubSub real-time example

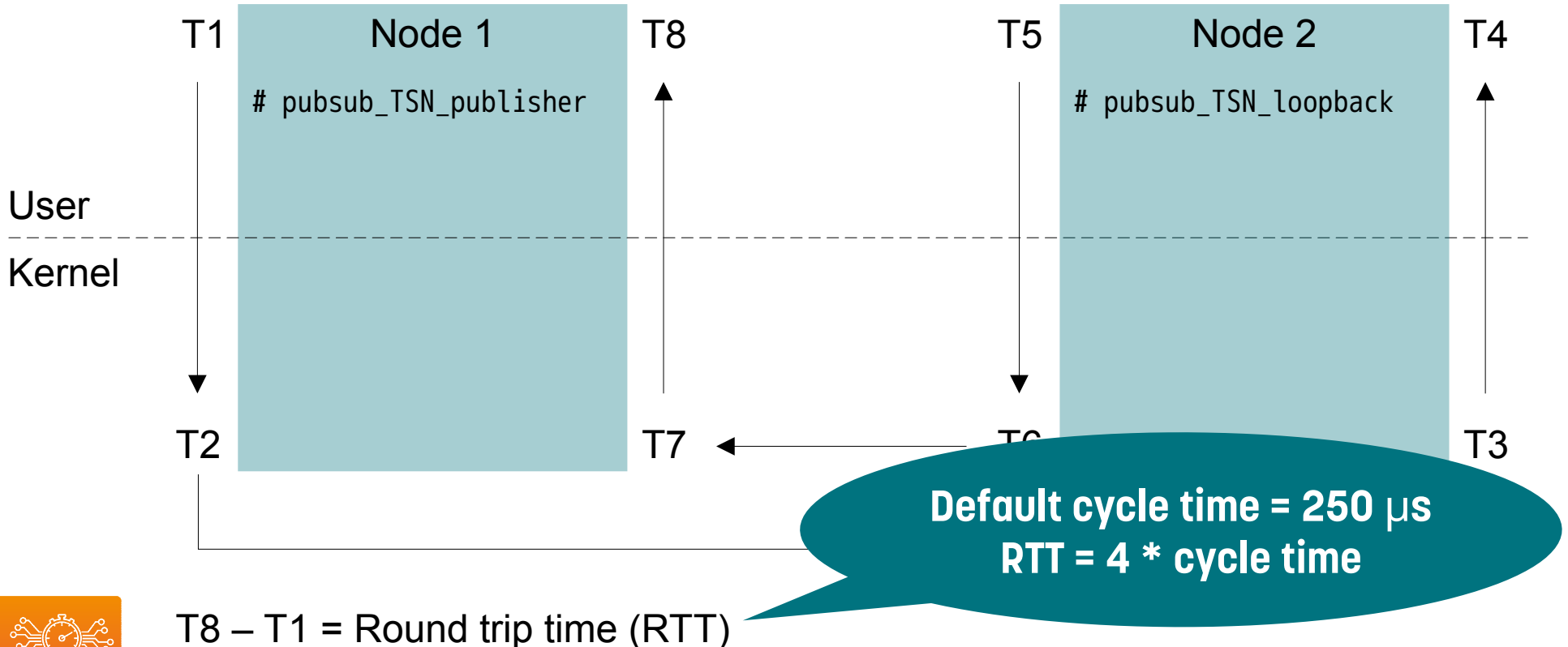


T8 – T1 = Round trip time

OPC UA TSN

OSADL

Open62541: PubSub real-time example



Open62541: PubSub real-time example

pubsub_TSN_publisher / pubsub_TSN_loopback			
Thread	**Default priority**	**Default CPU Core**	**Parameter**
Publisher	SCHED_FIFO 78	2	-pubPriority -pubCore
Subscriber	SCHED_FIFO 81	2	-subPriority -subCore
UserApplication	SCHED_FIFO 75	3	-userAppPriority -userAppCore

Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Start the loopback part first to avoid missed counters

OPC UA TSN

OSADL

Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

```
./pubsub_TSN_publisher \
-interface enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b3-0b:8.3
```

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

```
./pubsub_TSN_publisher \
-interface enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-
```

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Write the round trip latency to a CSV file (latencyT1toT8.csv)

Open62541: PubSub real-time example

Node 1 (**00:d0:93:46:b3:0b**):

```
./pubsub_TSN_publisher \
-interface enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b3-0b:8.3
```

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

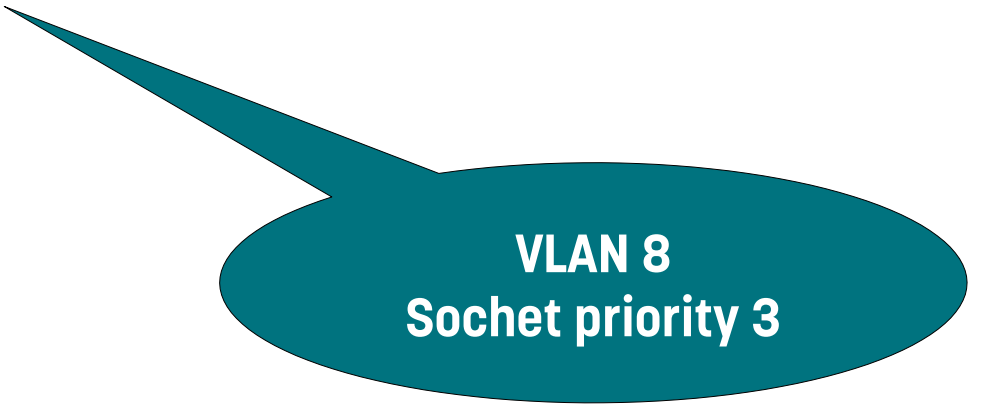
Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

```
./pubsub_TSN_publisher \
-interface enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b3-0b:8.3
```

Node 2 (**00:d0:93:46:b2:ff**):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```



Open62541: PubSub real-time example

Node 1 (**00:d0:93:46:b3:0b**):

```
./pubsub_TSN_publisher \
-interface enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b3-0b:8.3
```

Node 2 (00:d0:93:46:b2:ff):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Open62541: PubSub real-time example

Node 1 (00:d0:93:46:b3:0b):

```
./pubsub_TSN_publisher \
-interfac enp2s0 \
-enableBlockingSocket \
-enableLatencyCsvLog \
-pubMacAddress opc.eth://00-d0-93-46-b2-ff:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b3-0b:8.3
```

Node 2 (**00:d0:93:46:b2:ff**):

```
./pubsub_TSN_loopback \
-interface enp2s0 \
-enableBlockingSocket \
-pubMacAddress opc.eth://00-d0-93-46-b3-0b:8.3 \
-subMacAddress opc.eth://00-d0-93-46-b2-ff:8.3
```

Open62541: PubSub real-time example, latencyT1toT8.csv

```
997.716, 0, 0
999.623, 0, 0
1000.192, 0, 0
999.977, 0, 0
999.795, 0, 0
[...]
```