University of Stuttgart
Institute of Parallel and Distributed Systems

Distributed Systems

Fachpraktikum / Lab-Course

Software-Defined and Time-Sensitive Networking

Tutorial: Time Sensitive Networking
Part 1: Background and TSN in Linux

Frank Dürr

Summer Term 2023

IPVS

Agenda

- Motivation
- TSN Background
- TSN in Linux

Motivation

Many cyber physical systems are safety critical!

- Failure to react to physical world in time leads to substantial damage or even harm of people

➔ Deadlines must be met deterministically

Deterministic Real-Time Communication required for networked real-time systems

➔ Deterministically bounded network delay

Motivation

Example

Printing and Folding Machine

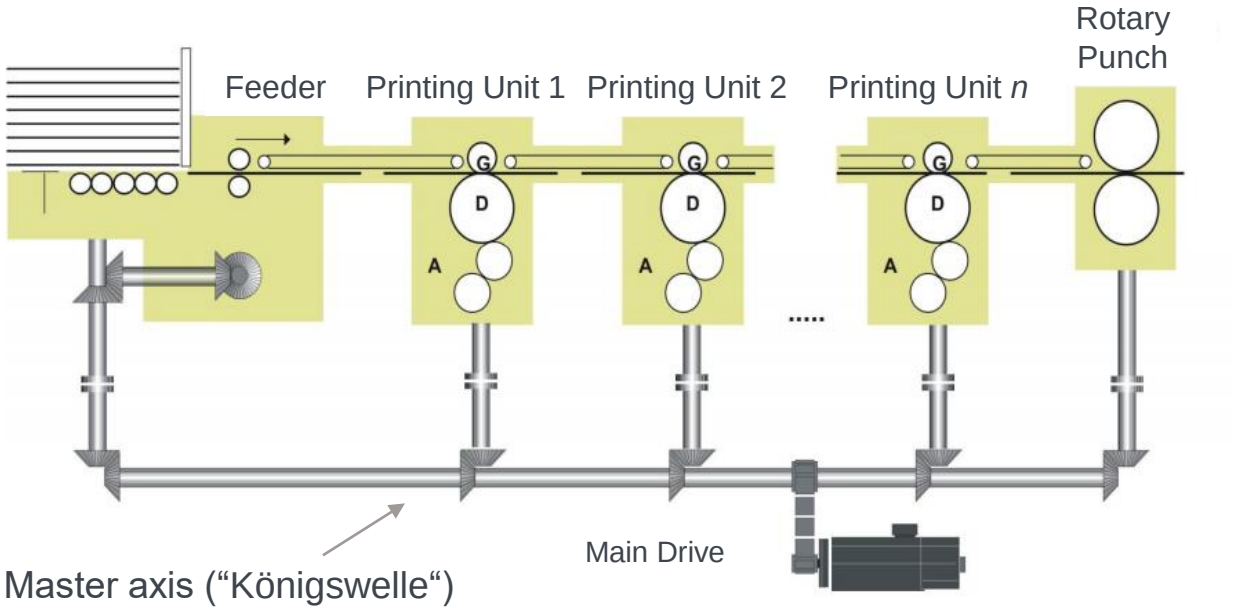
Motors:
> 50 axes

Final result: printed and folded box

[Source: Göpfer Maschinen GmbH]

Motivation

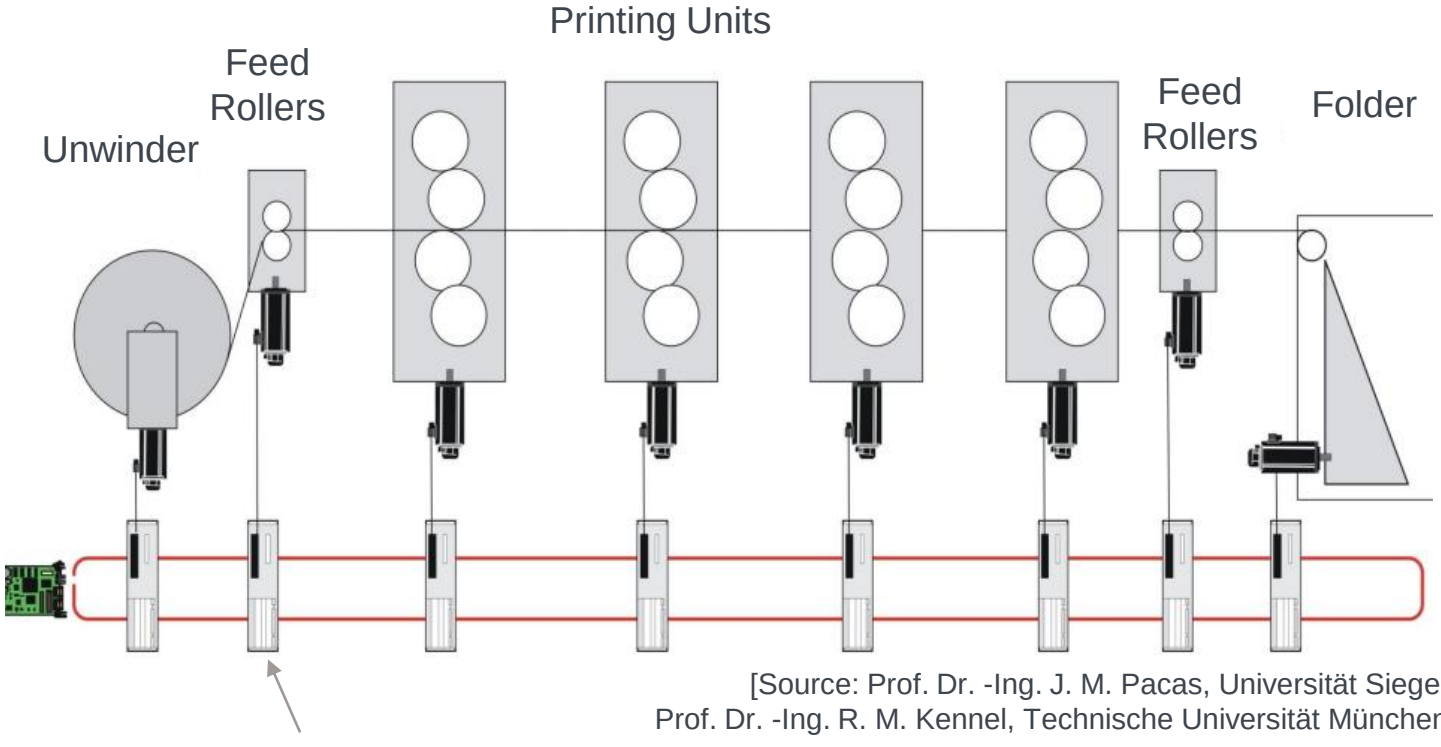
Old Concept: Mechanical Synchronization



[Source: Prof. Dr. -Ing. J. M. Pacas, Universität Siegen
Prof. Dr. -Ing. R. M. Kennel, Technische Universität München]

Motivation

New Concept: Electronic Motion Control



[Source: Prof. Dr. -Ing. J. M. Pacas, Universität Siegen
Prof. Dr. -Ing. R. M. Kennel, Technische Universität München]

Sensor and actuators:
servo motors connected to motion controller via **real-time communication network**

Motivation

Requirement: Low Deterministic Network Delay and Jitter

- Cycle time: time to read/update all sensors/actuators once
  - Defines update rate: each device once per cycle
  - Defines max. latency: all devices updated within same cycle (deadline = period)
- At high machine speeds, low cycle time is crucial
  - Printing machine: > 300 m/min material throughput speed
    - 1 ms cycle time means one update every 5 mm
- Industrial real-time networks: cycle times down to 31.25 µs
  - Example: SERCOS III (Ethernet-based)
    - Cycle times from 31.25 µs to 65 ms
    - 31.25 µs with 8 devices (axes), 500 µs – 1 ms with 100 devices (axes)
    - Jitter < 100 ns

Motivation

Converged Networks

- Initially, dedicated networks for real-time and non-real-time traffic
  - Separation of OT networks and IT networks
- Dedicated networks are costly
- Dedicated networks used precious space, weight, energy

➔ Converge OT and IT into one single network

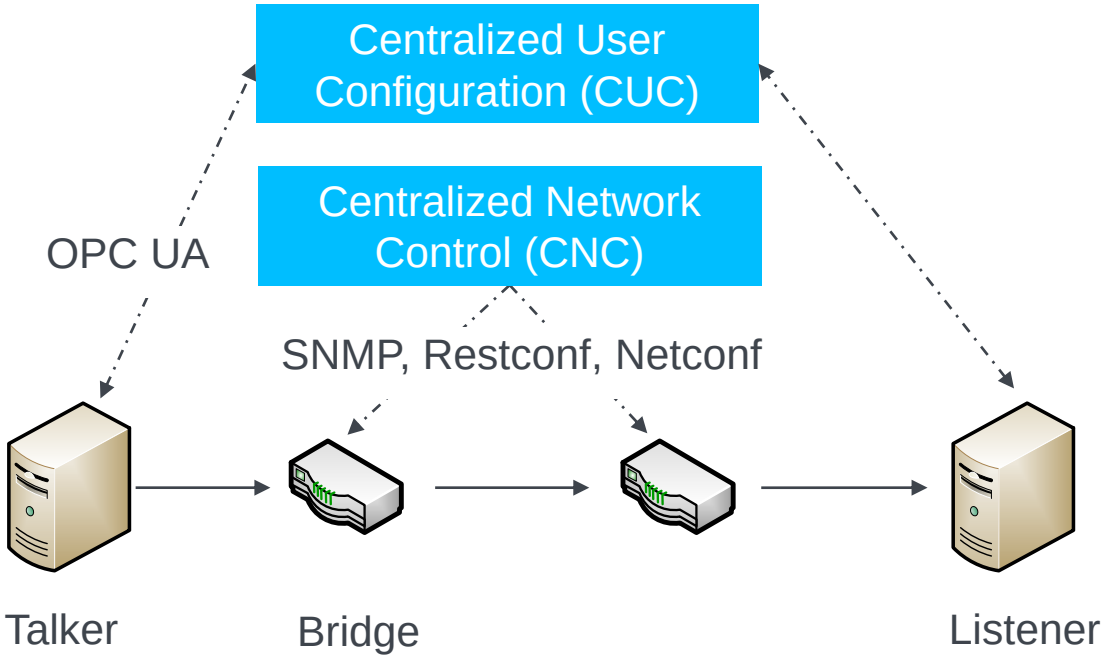
Agenda

- Motivation
- **TSN Background**
- TSN in Linux

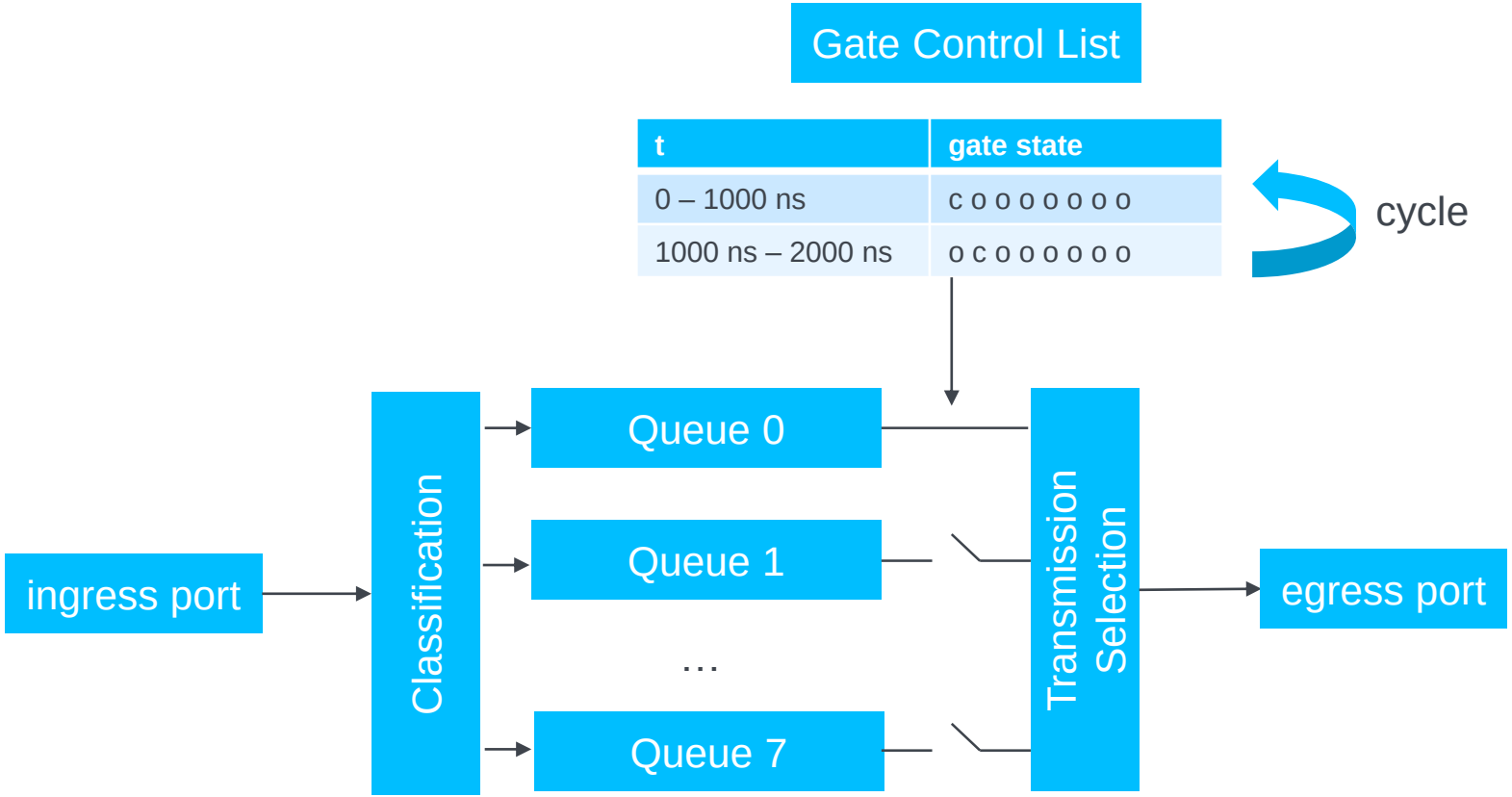
Time Sensitive Networking

- Time Sensitive Networking (TSN) is one technology for converged networks based on Ethernet Standard
- IEEE Standard 802.1Q
- Enhanced popular Ethernet technology (IEEE 802.3)
  - Precise time synchronization: Precision Time Protocol (PTP)
  - Scheduling with very low, deterministic network delay and jitter: Time-Aware Shaper
  - Frame Replication and Elimination to increase reliability
  - Frame Preemption to let time-critical traffic preempt transmission of non-time critical traffic

Architecture

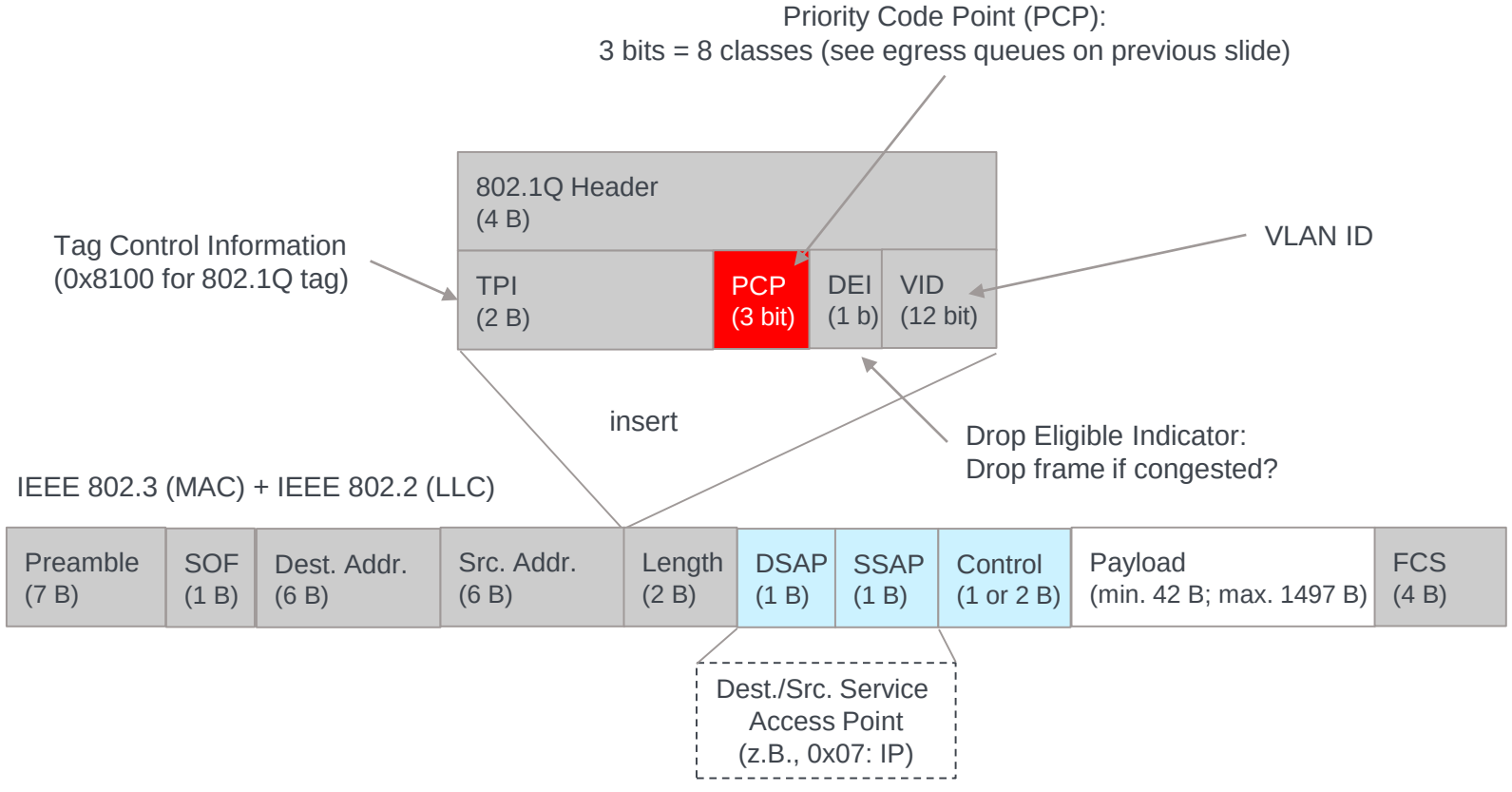


Time Aware Shaper (IEEE 802.1 Qbv)



Ethernet Frames

Definition of Priorities



Traffic Types

- Isochronous
  - Cyclic traffic with given period
  - Clocks of talkers and listeners are synchronized to working clock (network time as used by TSN bridges)
  - Transmission time of talkers can be aligned to gate control times
- Cyclic asynchronous
  - Cyclic traffic with given period
  - Clocks of talkers and listeners are not synchronized to working clock
  - Transmission cycles of talkers are not aligned to gate control times
- Sporadic / asynchronous
  - Aperiodic transmission

Agenda

- Motivation
- TSN Background
- **TSN in Linux**

Time-Aware Shaping in Linux

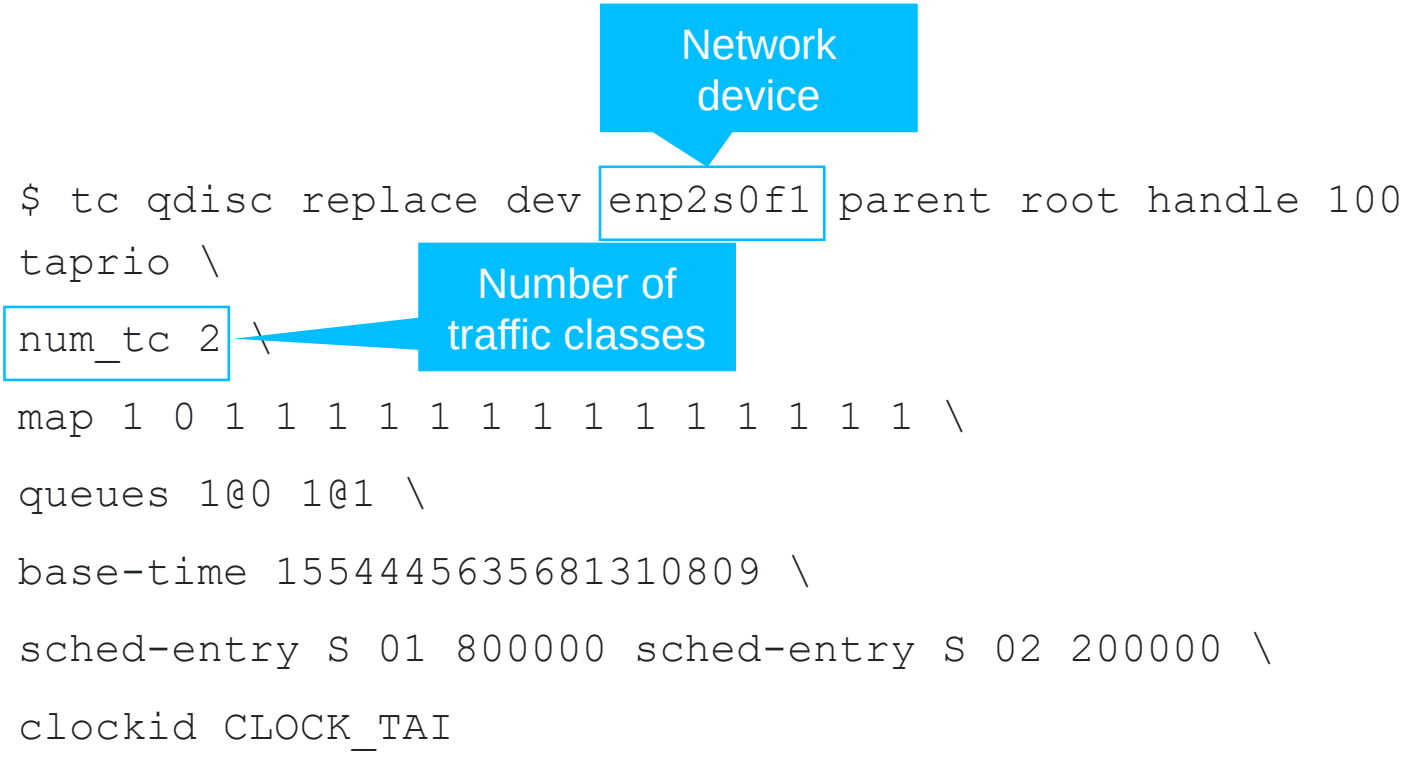
TAPRIO (Time Aware Priority Shaper)

Linux implements Time-Aware Shaping through a QDisc (Queueing Discipline): **TAPRIO (Time Aware Priority Shaper)**

- Controls egress traffic of network device
- Time-driven gates similar to Time-Aware Shaper
- Controlled through Command Line Interface
  - No control plane implementation (NETCONF, RESTCONF, SNMP)

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)



Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

```
$ tc qdisc replace dev enp2s0f1 parent root handle 100
taprio \
num_tc 2 \
map 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 \
queues 1@0 1@1 \
base-time 1554445635681310809 \
sched-entry S 01 800000 sched-entry S 02 200000 \
clockid CLOCK_TAI
```

Mapping of SKB priorities 0…15 to traffic classes

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

```
$ tc qdisc replace dev enp2s0f1 parent root handle 100
taprio \
num_tc 2 \
map 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
queues 1@0 1@1 \
base-time 1554445635681310809 \
sched-entry S 01 800000 sched-entry S 02 200000 \
clockid CLOCK_TAI
```

Mapping of traffic classes to egress queues of device:

- 1st class: 1 queue at queue index 0
- 2nd class: 1 queue at queue index 1

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

```
$ tc qdisc replace dev enp2s0f1 parent root handle 100
taprio \
num_tc 2 \
map 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 \
queues 1@0 1@1 \
base-time 1554445635681310809 \
sched-entry S 01 800000 sched-entry S 02 200000 \
clockid CLOCK_TAI
```

Cycle base time in nano-seconds

System clock to use:
TAI (Temps Atomique International):
no leap seconds

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

```
$ tc qdisc replace dev
taprio \
num_tc 2 \
map 1 0 1 1 1 1 1 1 1
queues 1@0 1@1 \
base-time 1554445635681[illegible]0809 \
sched-entry S 01 800000 sched-entry S 02 200000 \
clockid CLOCK_TAI
```

Gate Control List (Schedule):

- S 01: set gate state (only possibility)
- 01: gate 0 open (first bit)
- 02: gate 1 open (second bit)
- 800000: duration of entry in nano-secs

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

TAPRIO uses socket buffer (SKB) priorities for scheduling!

- Not directly the PCP
- Application can define priority for sockets:

```
setsockopt(
      socket,
      SOL_SOCKET,
      SO_PRIORITY,
      &priority, // int value
      sizeof(priority));
```

- For egress and ingress packets:
  PCP needs to be mapped to SKB priority and vice versa (see below)

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

TAPRIO requires network device to have 8 TX (transmit) queues

- Check number of TX queues of device:

```
$ ls /sys/class/net/enp2s0f1/queues/
rx-0 rx-1 rx-2 rx-3 rx-4 rx-5 rx-6 rx-7 tx-0 tx-1 tx-2
tx-3 tx-4 tx-5 tx-6 tx-7
```

- Create virtual Ethernet device (veth) with several TX queues:

```
$ sudo ip link add vethname1 numtxqueues 8 type veth
peer name vethname2 numtxqueues 8
```

Time-Aware Shaping in Linux

TAPRIO (Time Aware Priority Shaper)

Recommended reading:

- Man page of TAPRIO: `man taprio`
- Blog post: https://www.frank-durr.de/posts/2019/04/11/software_tsn_switch.html

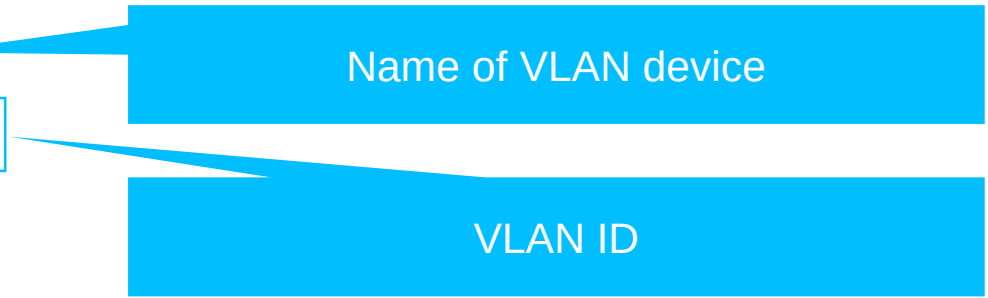
VLANs in Linux

Creating VLAN Devices

Sending a Ethernet frame with PCP requires a VLAN tag

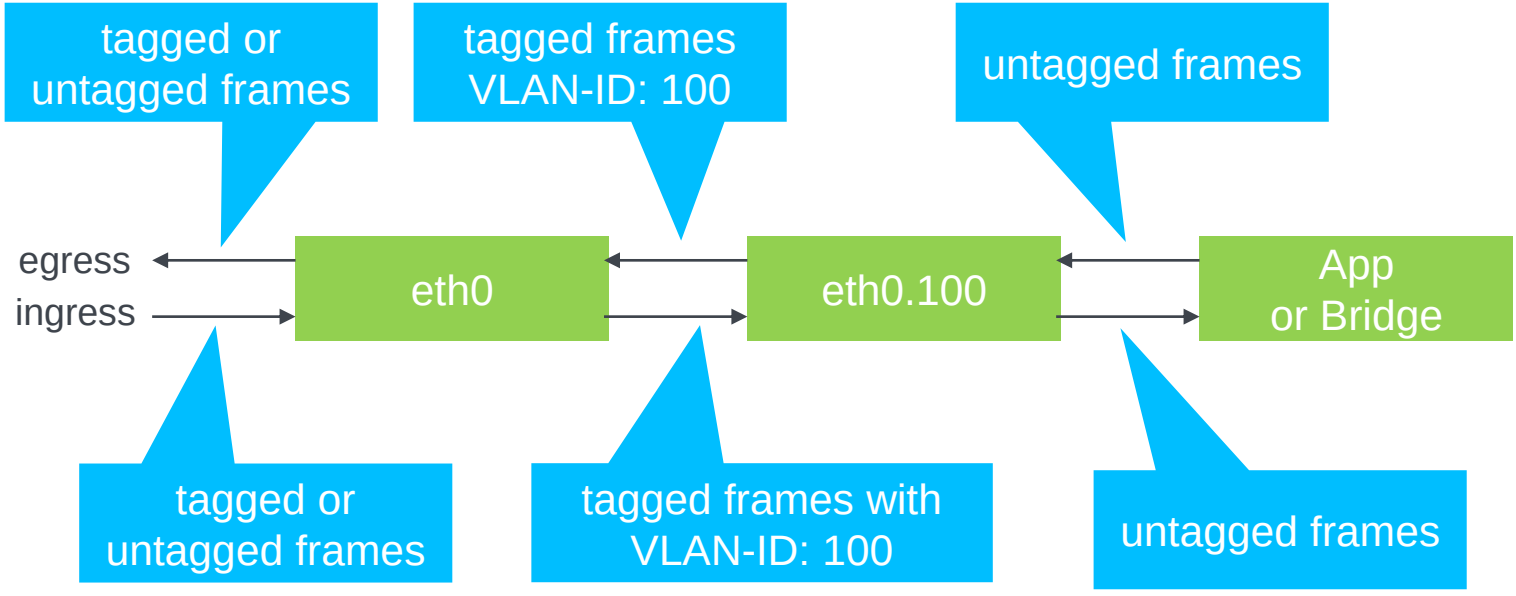
- Creating a VLAN interface for an existing device (eth0):

```
$ sudo ip link add link eth0 \
name eth0.100 \
type vlan id 100
```



VLANs in Linux

Tagged and Untagged Frames



VLANs in Linux

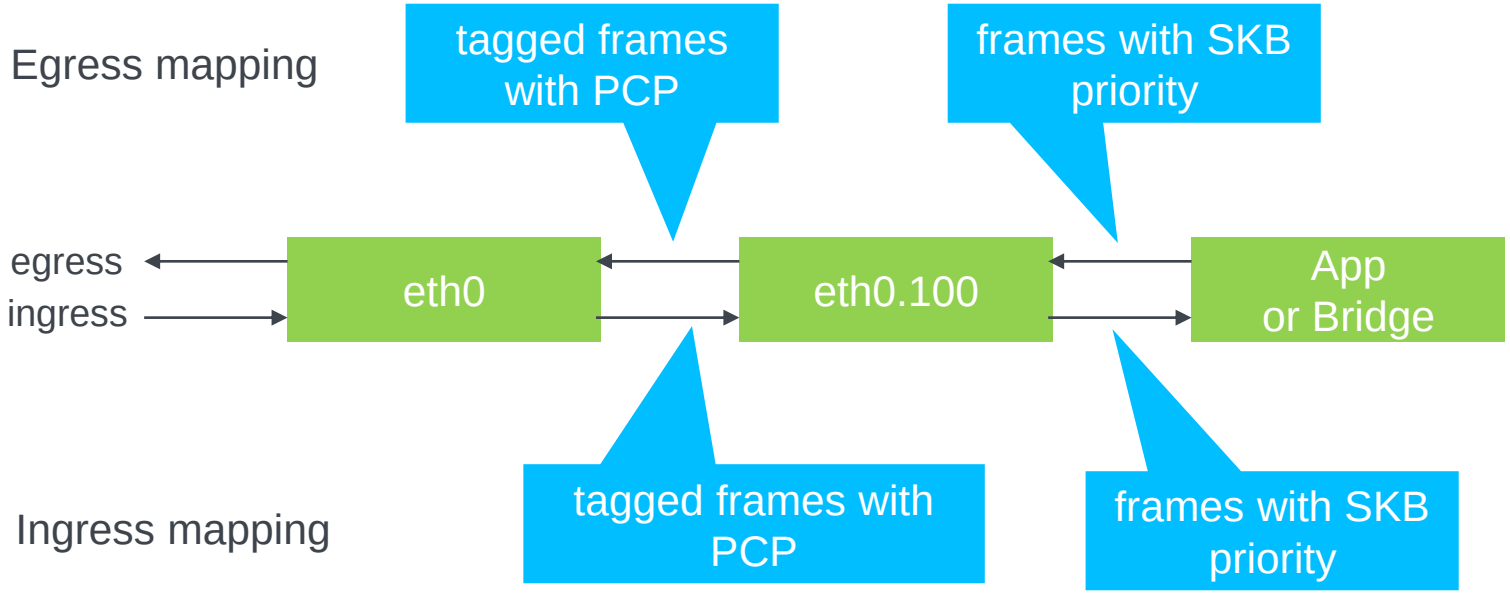
PCP Mapping

SKB priorities can be mapped to PCP values for ingress and egress traffic:

```
$ sudo ip link set eth0.100 \
type vlan \
egress 0:0 1:1
```

VLANs in Linux

Priority Mapping



Time and Clocks in POSIX

Getting Current Time

Implementing isochronous and cyclic talkers requires clocks:

Getting current time:

```
clock_gettime(CLOCK_MONOTONIC, &t);
```

- `CLOCK_MONOTONIC`: “A nonsettable system-wide clock that represents monotonic time since—as described by POSIX—"some unspecified point in the past”. [man page of clock_gettime]
- `CLOCK_TAI`: “ A nonsettable system-wide clock derived from wall-clock time but ignoring leap seconds. This clock does not experience discontinuities and backwards jumps caused by NTP inserting leap seconds as CLOCK_REALTIME does.” [man page of clock_gettime]

Time and Clocks in POSIX

Sleeping

Sleeping for an interval or until an absolute point in time:

```
int clock_nanosleep(
     clockid_t clockid, // Clocks see pervious slide
     int flags,         // TIMER_ABSTIME to sleep until
                        // a point in time
     const struct timespec *request,  // relative or
                                      // absolute time
     struct timespec *remain);    // remaining time if
                                  // sleep is interrupted
```

Questions?