DETERMINISTIC6G

Dependable6G Summer School

Introduction to Wired TSN

Stockholm, Sweden, September 10, 2025

Frank Dürr, Simon Egger, Lucas Haug (University of Stuttgart)
Mahin Ahmed (Silicon Austria Labs GmbH)

DETERMINISTIC6G has received funding from the European Union’s Horizon Europe programme under grant agreement No 101096504.

Agenda

- Motivation
- TSN Overview
- TSN Shapers:
  - Time-Aware Shaper
  - Credit-based Shaper
- Reliability
- Conclusion

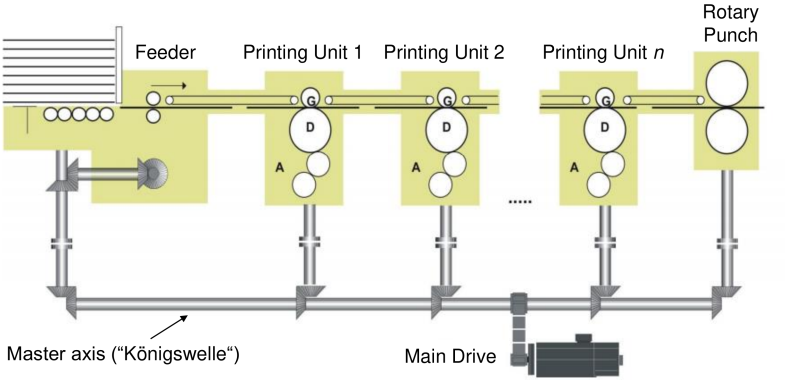
Example Usecase

Printing and Folding Machine:[a]

- Many axes that need to be synchronized
- Final result:

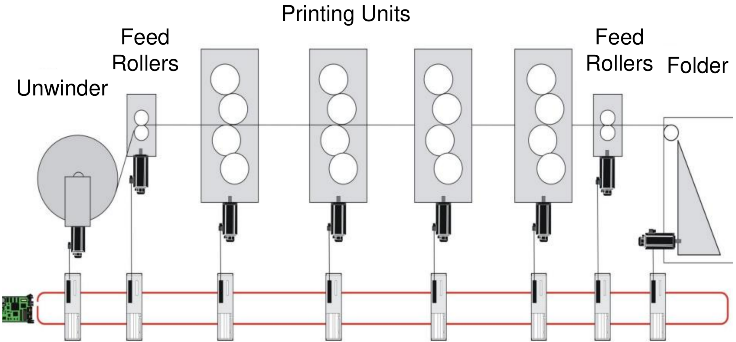
[a]Source: Göpfer Maschinen GmbH

Old Concept: Mechanical Synchronization



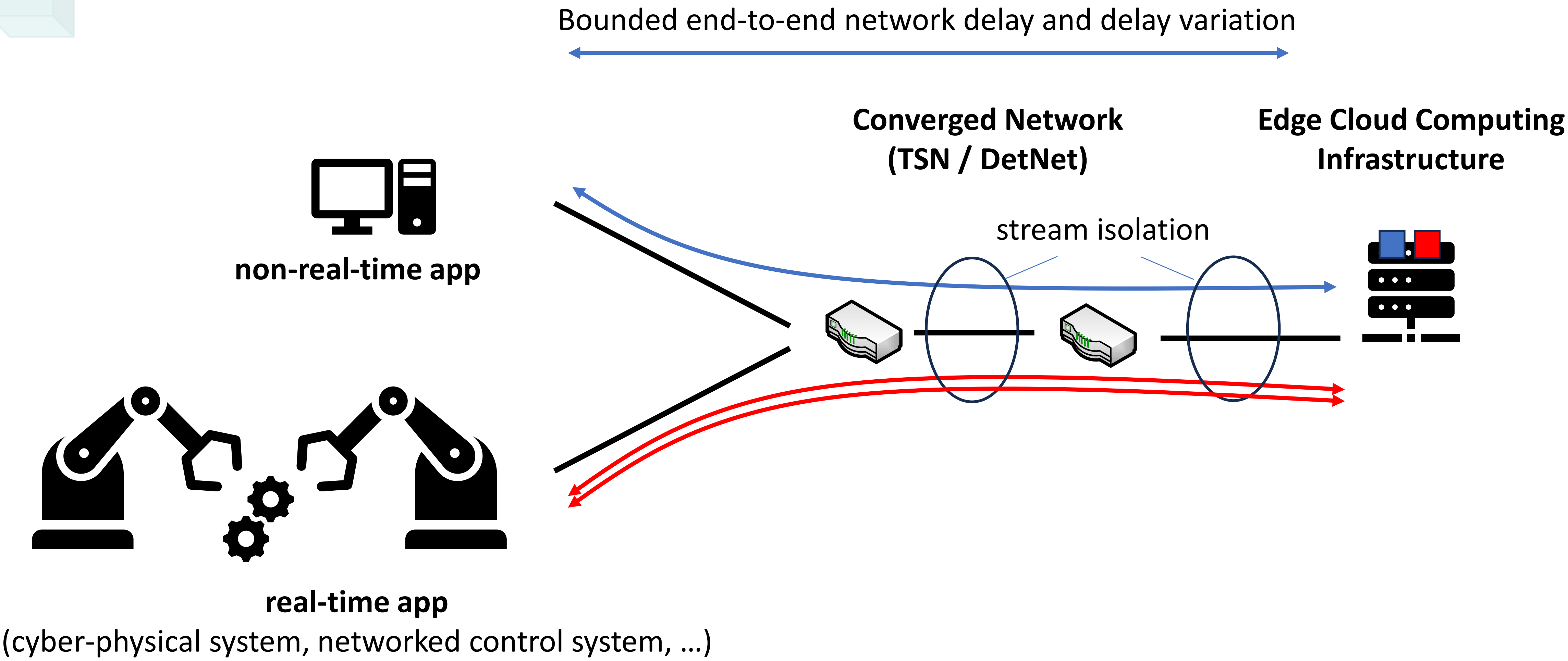
[Source: Prof. Dr.-Ing. J. M. Pacas (Universität Siegen),
Prof. Dr.-Ing. R. M. Kennel (Technische Universität München)]

New Concept: Electronic Motion Control



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

Use Case: Distributed Service with Edge Cloud



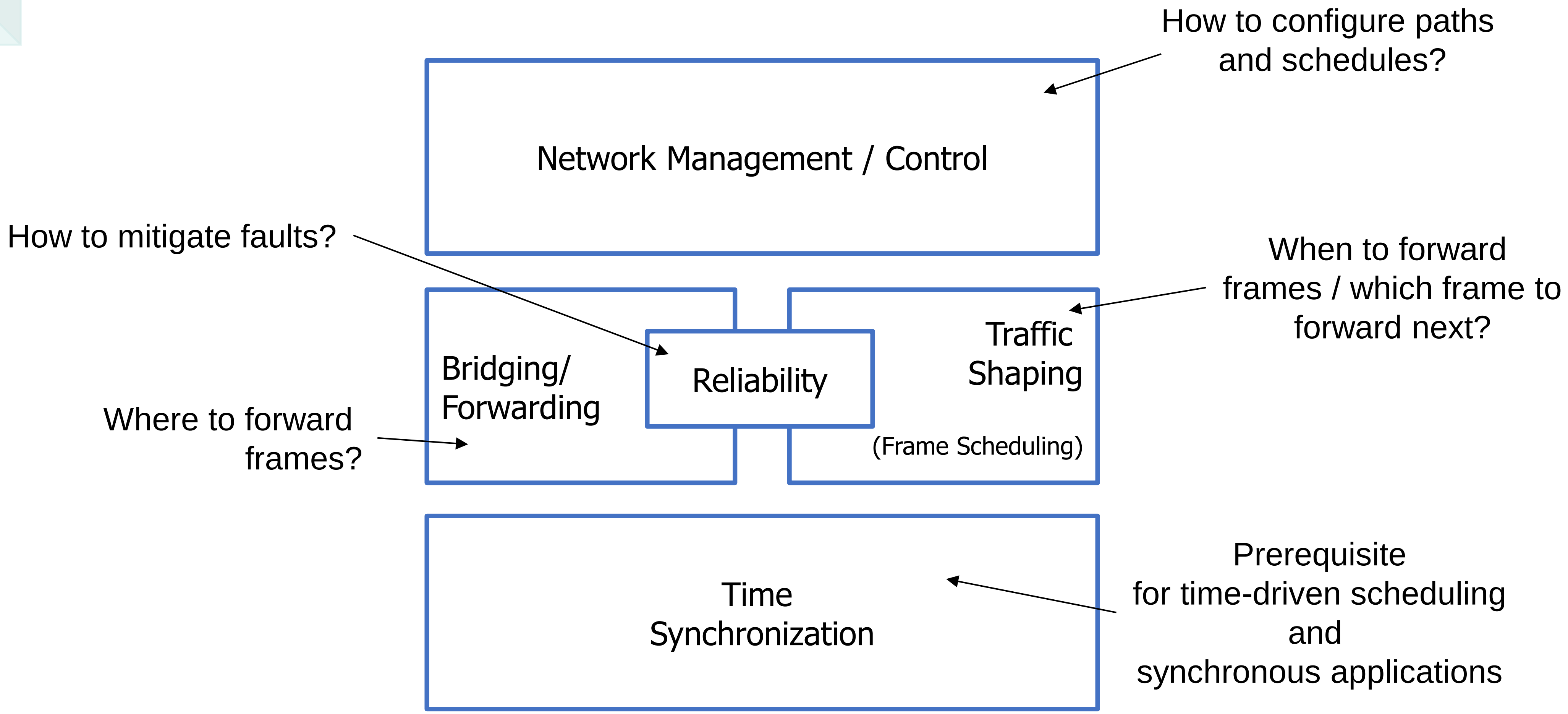
Agenda

- Motivation
- **TSN Overview**
- TSN Shapers:
  - Time-Aware Shaper
  - Credit-based Shaper
- Reliability
- Conclusion

Classification of Real-Time Protocols

Application Layer	
Transport Layer	e.g. RTP, RTCP
Network Layer	e.g. IntServ, DetNet
Data Link Layer	e.g. TSN, TTEthernet
Physical Layer	

Overview of TSN



Traffic Shaping (Frame Scheduling)

- **IEEE 802.1Qav: Forwarding and Queuing Enhancements** for Time-Sensitive Streams
  - Credit-Based Shaper (per-hop/per-queue traffic shaping)
  - Targeting audio video bridging (AVB)
- **IEEE 802.1Qbv: Enhancements for scheduled traffic**
  - Time-aware Scheduling: timed gating mechanism (TDMA, clock-driven scheduling scheme)
  - Targeting real-time communication with very low deterministic latency bounds
- **IEEE 802.1Qrc: Asynchronous Traffic Shaping**
  - Per-hop/per-stream shaping + priority scheduling
  - Does not require globally synchronized clocks
  - Targeting real-time communication with deterministic latency bounds

Traffic Shaping (Frame Scheduling)

- **IEEE 802.1Qch: Cyclic Queuing and Forwarding**
  - Double-buffering of frames and synchronized forwarding
  - Simpler than IEEE 802.1Qbv; limited to scenarios with longer cycle times
  - Targeting real-time communication with deterministic latency bounds

- **IEEE 802.1Qbu: Frame pre-emption**
  - Preempt transmission of non-time-critical low(er) priority frames by high(er) priority express frames
  - Reduces blocking time of high-priority frames by low-priority frames
  - Improves efficiency of Time-Aware Scheduling (IEEE 802.1Qbv)

Forwarding

- IEEE 802.1aq: Shortest Path Bridging (SPB)
  - Replacement for inefficient spanning tree protocols, allowing for using multiple available shortest paths
  - Based on IS-IS link-state protocol

- IEEE 802.1Qac: Path control and reservation
  - Capabilities beyond SPB: explicit path control (also non-shortest paths), bandwidth reservation, and redundancy (protection, restoration) for data flows

Reliability

- **IEEE 802.1Qci: Per-Stream filtering and policing**
  - Detection and mitigation of disruptive transmissions

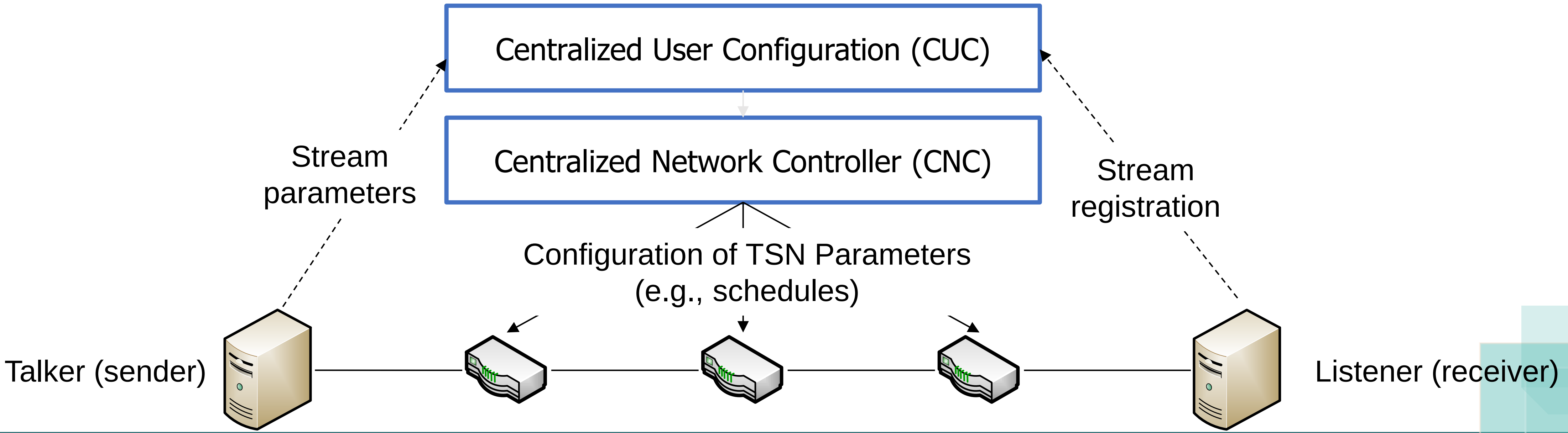
- **IEEE 802.1CB: Frame replication and elimination for reliability**
  - Frame replication on separate disjunct paths, sequence numbering, duplicate elimination

Network Management / Control

- ❑ **IEEE 802.1Qat: Stream reservation protocol (SRP)**
- ❑ **IEEE P802.1Qcc: SRP Enhancements and Performance Improvements**

➔ Decentralized and centralized models for TSN configuration of paths and schedules

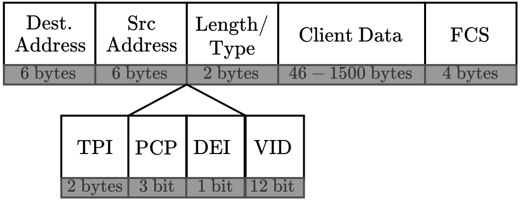
Centralized Model



Clock Synchronization

- **IEEE P802.1AS: Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks**
  - Generalized Precision Time Protocol (gPTP)
  - Synchronization of network elements (bridges) and end stations

802.1Q Header in Ethernet Frames



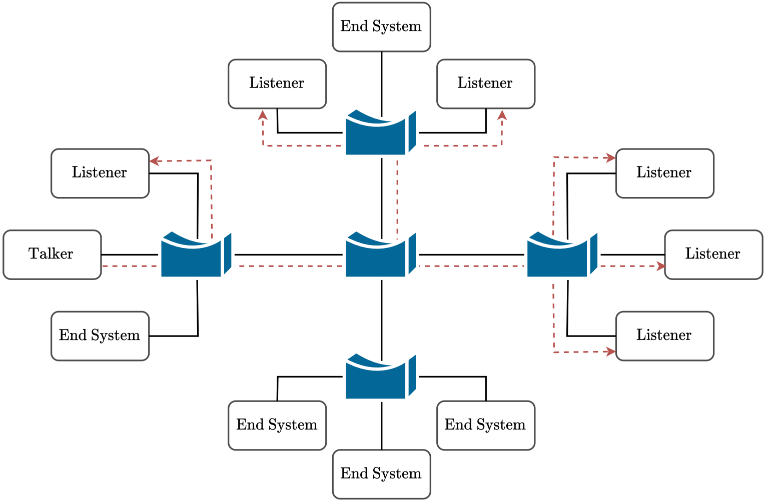
Tag Protocol Identifier (TPI) identifies 802.1Q-tagged frames

Priority Code Point (PCP) specifies priority of the TSN stream

Drop Eligible Indicator (DEI) can mark frames that exceed their committed rate

VLAN Identifier (VID) specifies the logical broadcast domain of the TSN stream

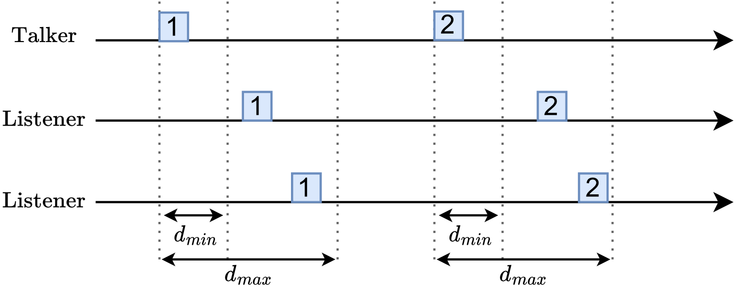
Talkers and Listeners in TSN



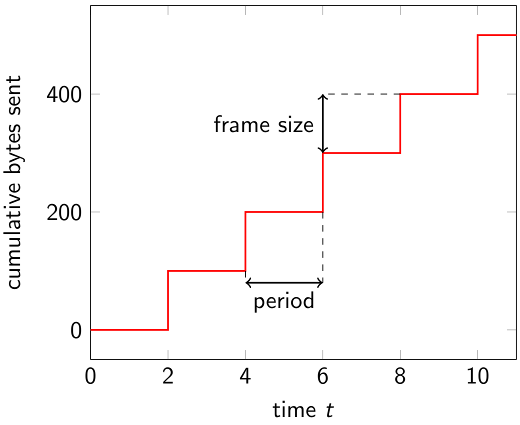
Real-Time Guarantees $\neq$ Fast

Explicit punctuality guarantees for each message stream f:

$$d_{max} \leq f.e2e\_latency \quad \text{and} \quad d_{max} - d_{min} \leq f.jitter$$



Time-Triggered Traffic



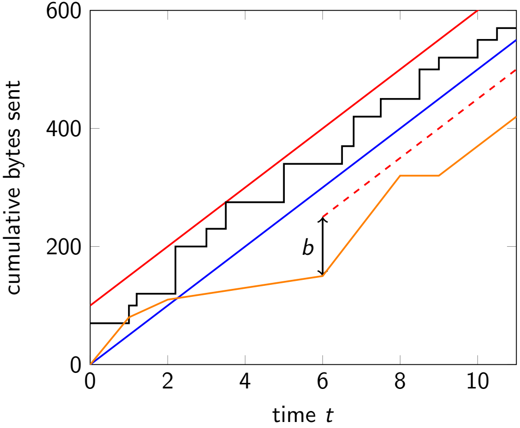
Traffic Characteristics:

- Periodic
- Fixed frame size

(Typical) Requirements:

- Deadline or latency
- Low tolerance to jitter
- Low tolerance to frame loss

Rate Constrained Traffic



Traffic Characteristics:

- Token bucket $TB(r, b)$

(Typical) Requirements:

- Latency or bandwidth
- Low to medium criticality
- Some tolerance to frame loss

Traffic and Quality of Service Specification

Typically, an application invokes network service with two specifications:

- **Traffic specification (TSPEC)**
  - Contract with network (promise to network) about talker traffic (traffic bounds)
  - Obviously, no real network can guarantee any QoS bounds for unbounded traffic

- **Quality of Service (QoS) specification**
  - From IEC/IEEE 60802 TSN Profile for Industrial Automation:
    - **Frame Latency**: “data delivery of a frame for a given Talker-Listener pair occurs within a bounded timespan.”
    - **Flow Latency:** “data delivery up to a certain number of frames or data size (including bursts of frames) occurring over a defined period.”
    - **Deadline:** “data delivery of a frame to a given Listener occurs at or before a specific point in time.”

TSPEC in Stream Reservation Protocol (SRP)

TSPEC

- **Interval**
  - Below limits must not be exceed in this period
- **Max frames per interval**
- **Max frame size**
- **Transmission selection**
  - Strict priority, CBS, …

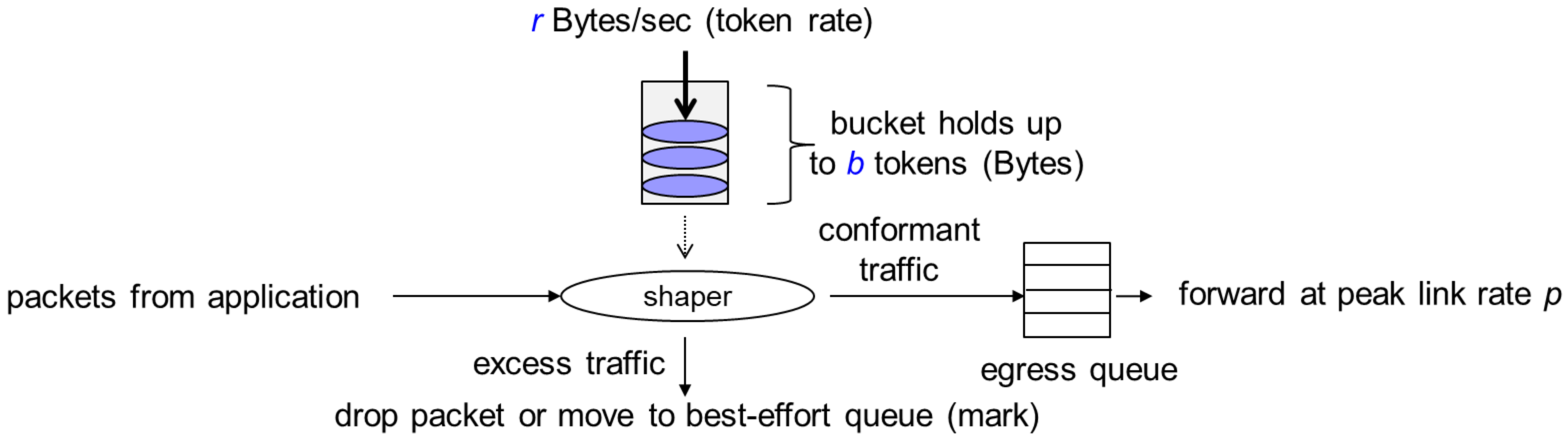
TSPEC Time Aware (Talker sync to network)

- **Earliest Transmit Offset**
  - Earliest time offset in interval when stream can be scheduled by network
- **Latest Transmit Offset**
  - Latest time offset in interval when stream can be scheduled by network
- **Jitter**
  - Maximum difference in time between the Talker's transmit offsets and the ideal synchronized network time

Token Bucket TSPEC

Bursty traffic with limited burst length and size (e.g., MPEG video stream with I/P/B frames)

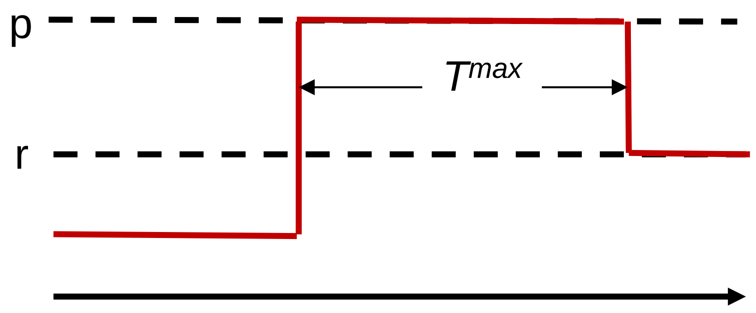
- **Token bucket:**
  - **Token rate: r** [bytes of IP datagrams/s]
  - **Bucket size: b** [byte])
- **Peak rate of egress link/port: p** [bytes of IP datagrams/s]



Limited Burst Size and Sustainable Rate

Maximum burst size:

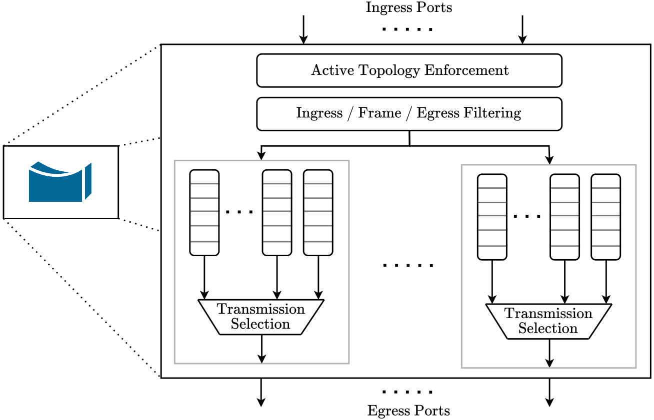
- Let *p* be the max. (peak) transmission rate, *b* and *r* as above
- T^{max} ... max. burst time
- B^{max} ... max. burst size (in bits)



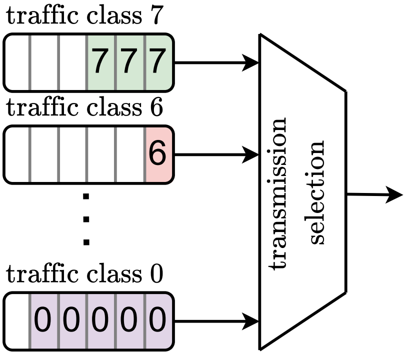
$$T^{max} = \frac{b}{p-r} \quad \text{for} \quad r < p$$

$$B^{max} = \frac{b}{p-r} \cdot p$$

TSN Bridge Model



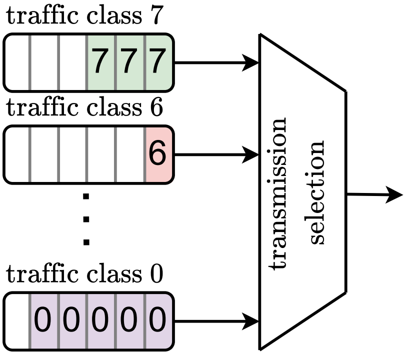
Strict Priority Transmission Selection



Policy: *The first frame of queue i is eligible for transmission if all queues $j > i$ are empty.*

Example: $7, 7, 7, 6, \ldots, 0, 0, 0, 0, 0$

Strict Priority Transmission Selection



Policy: *The first frame of queue i is eligible for transmission if all queues $j > i$ are empty.*

Example: $7, 7, 7, 6, \ldots, 0, 0, 0, 0, 0$

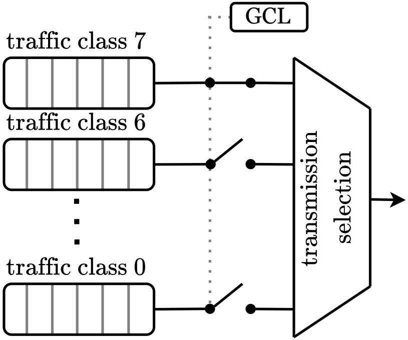
Can you think of potential problems?

Agenda

- Motivation
- TSN Overview
- **TSN Shapers:**
  - **Time-Aware Shaper**
  - Credit-based Shaper
- Reliability
- Conclusion

DETERMINISTIC6G

IEEE 802.1Qbv Time-Aware Shaper



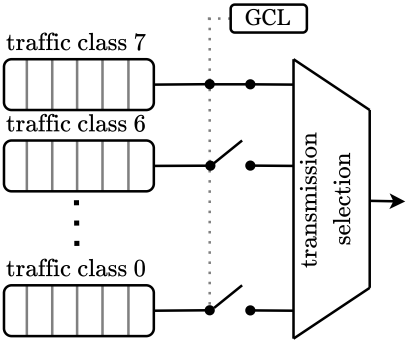
Gate Control List (GCL): Timetable that specifies when gates open and close.

Example:

GCL

T01: ococococ
T02: cocococo
⋮
T40: occccccc
T41: cocccccc
T42: ccoccccc

IEEE 802.1Qbv Time-Aware Shaper



Gate Control List (GCL): Timetable that specifies when gates open and close.

Policy: *The first frame of queue i is eligible for transmission if its gate is open and no other frame of queue $j > i$ is eligible.*

Calculating Time-Aware Shaper Schedules

(Just) one simple example for calculating schedules (gate control lists) for Time-aware Shaper:
No-wait Packet Scheduling

- Idea
  - Adapt a well-known problem from operations research: No-wait Job Shop Scheduling Problem
  - Solve as Integer Linear Program (ILP) (shown next; slow but exact solution) or with fast heuristics (see paper; fast but not always optimal solution)
- Details:
  - Frank Dürr, Naresh Nayak: *No-wait Packet Scheduling for IEEE Time-sensitive Networks (TSN).* In Proceedings of the 24th International Conference on Real-Time Networks and Systems (RTNS 2016), October 2016, DOI: 10.1145/2997465.2997494.

Integer Linear Programming (ILP)

- Integer Linear Programming (ILP) is a powerful method to solve optimization problems that can be expressed with linear constraints and objectives
  - Other alternative formal methods: Constrained Programming (CP), SMT, heuristics

- ILP incorporates application-specific constraints and objectives

- In general, ILPs consist of:
  - objective function (e.g. $\max \sum_i c_i \cdot x_i$)
  - linear constraints (e.g. $a_1 \cdot x_1 + a_2 \cdot x_2 \leq b_1$)
  - parameters (e.g. $a_1, a_2, a_3, \ldots, b_1, b_2, b_3, \ldots, c_1, c_2, c_3, \ldots$)
  - variables (e.g. $x_1, x_2, x_3, \ldots$)

DETERMINISTIC6G

Integer Linear Programming (ILP): Solvers

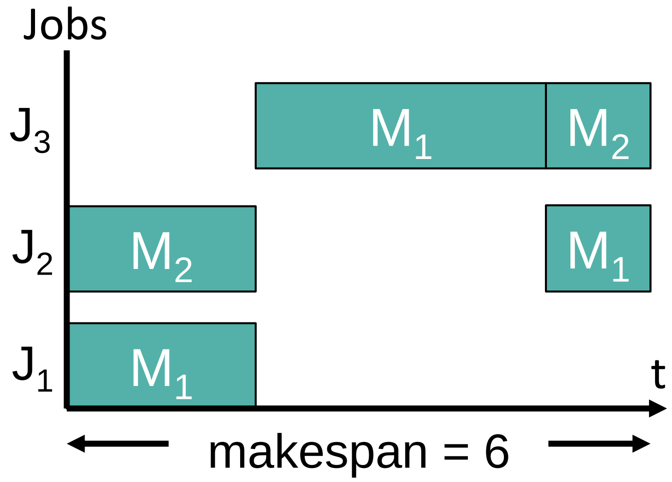
ILP Solver (software tool) will solve the specified problem and provide solutions

- Exacts solutions!
  - If problem is feasible: variable values that fulfill constraints and provide optimal solution according to objective
  - If problem is infeasible: definite answer that there is no solution fulfilling the constraints
- Many mature products (often free to use for academic research): IBM Cplex, Gurobi, …

- Possibly long time to find optimal solution!
  - In general, NP-hard constrained optimization problems (see example with Job Shop Scheduling next)
  - Often, solution fulfilling constraints is found relatively quickly
  - Proof of optimality often takes long time

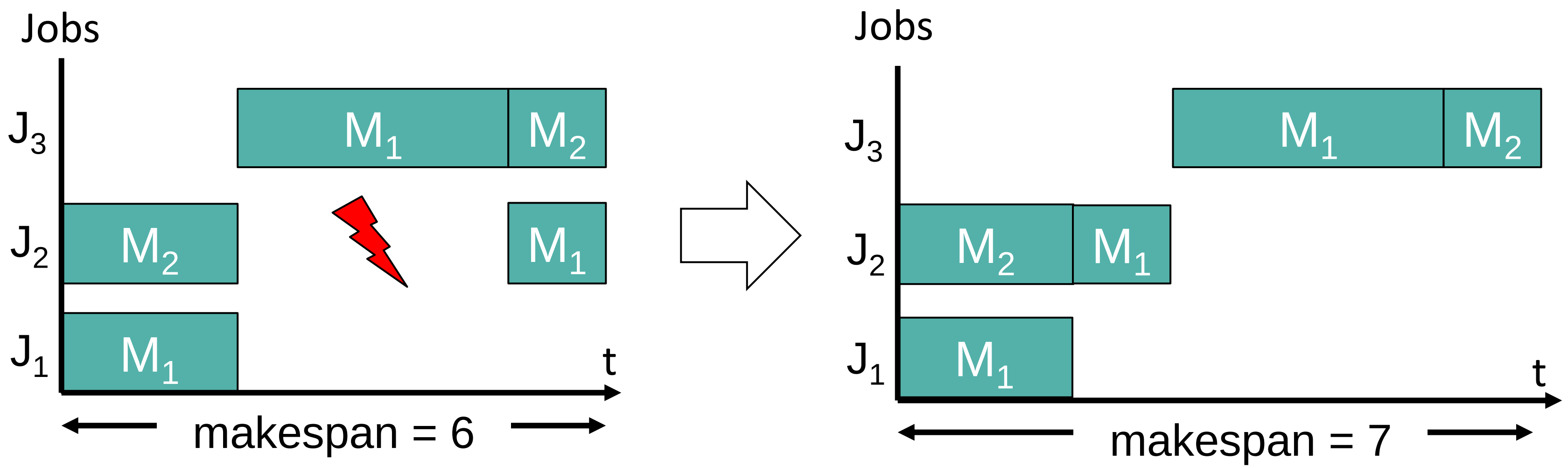
Background: Job Shop Scheduling Problem

- Well-known problem in operations research (NP-hard)
- Scheduling of set of jobs (sequence of operations) on a set of machines on a shop floor
  - Compute schedules with minimal makespan (duration all jobs)
- Example: 2 machines (M_1, M_2) executing 3 jobs (J_1, J_2, J_3)
  - Job J1:
    1. M1, execution time = 2
  - Job J2:
    1. M2, execution time = 2
    2. M1, execution time = 1
  - Job J3:
    1. M1, execution time = 3
    2. M2, execution time = 1



Background: No-wait Job Shop Scheduling Problem

- Job Shop Scheduling Problem + no-wait constraint
  - Once a job starts, it must run to completion
  - E.g., forging metal immediately after heating

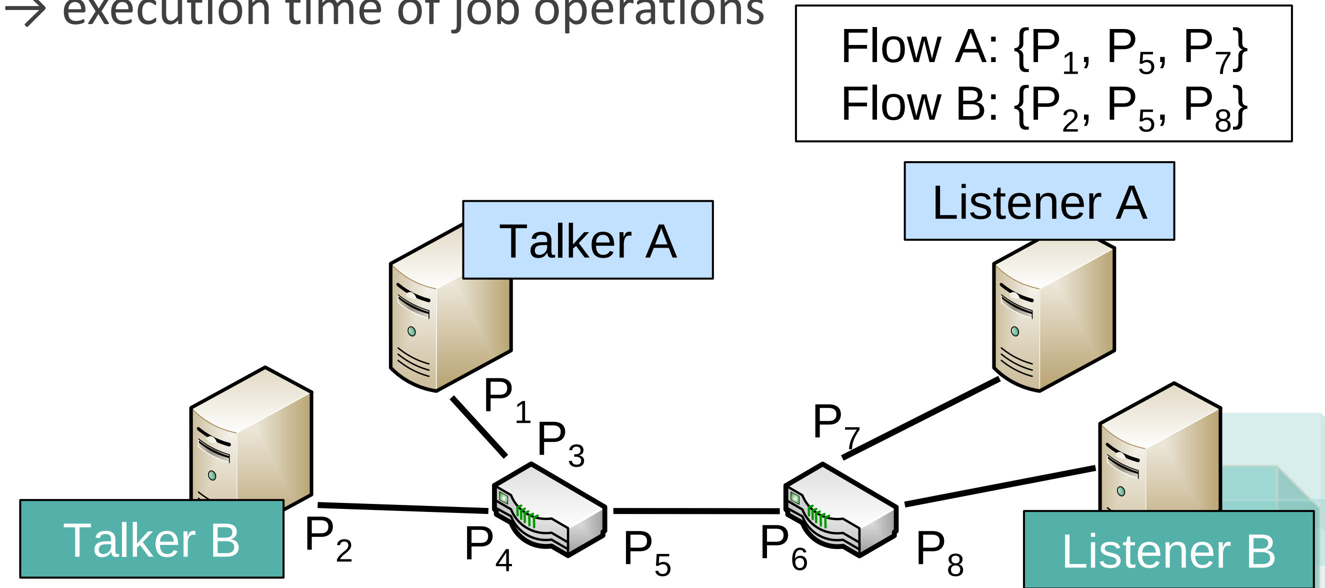


no-wait constraint violated for job J_2

no-wait constraints respected

No-wait Packet Scheduling Problem

- Switch ports and host network interfaces → Machines
- Flows of scheduled traffic → Jobs
  - Sequence of forwarding operations by switch ports/interfaces along network path → sequence of job operations
  - Time for each operation = time for serializing packet on the corresponding port (depending on known packet size and link speed) → execution time of job operations
- No-wait constraint
  - Scheduled traffic must not be queued (zero queuing delay)
  - ➔ Minimal end-to-end delay



ILP Formulation (1)

Parameters

- Set of flows (streams): $F = \{F_1, \dots, F_n\}$
- Set of switch ports/host interfaces: $P = \{P_1, \dots, P_m\}$
- For each flow F_i, set of transmission operations $O_i = \{O_{i,1}, \dots, O_{i,ni}\}$
- Mapping R of each transmission operation to an egress switch port: $R: O \rightarrow P$
- Transmission delay for each transmission operation $O_{i,j}$ of flow F_i (depending on given packet sizes and link speeds): $d_{i,j}^{trans}$
- Packet processing delay (same for all switches): d^{proc}
- Propagation delay (same for all links): d^{prop}

ILP Formulation (2)

Variables

- Start times for each flow f_i (time of injecting packet at sources):
  $T = \{t_1, \dots, t_n\}$

- The gate opening time for forwarding operation $O_{i,j}$ can be calculated by adding the cumulative network delay (constraints in ILP)
  $D_{i,j-1} + d^{prop} + d^{proc}$ (see next slide)

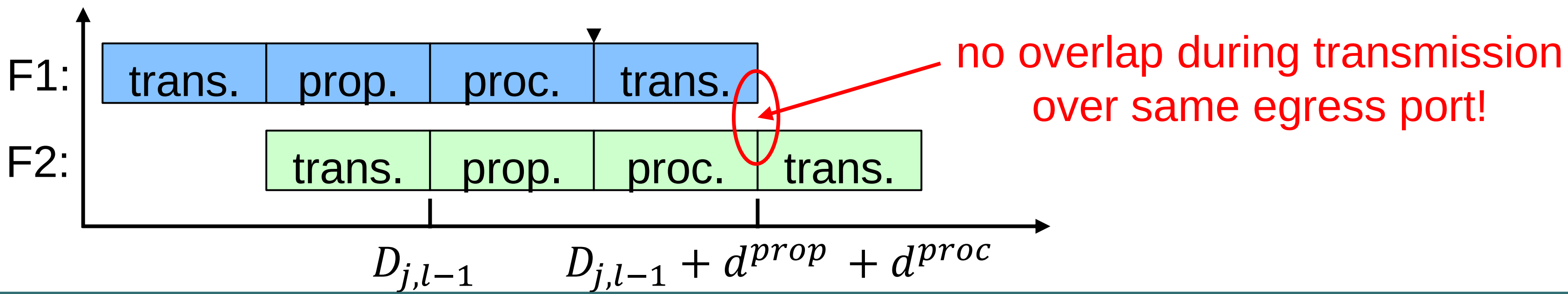
ILP Formulation (3)

- Cumulative network delay $D_{i,j}$ (processing, propagation, and transmission delay) up to and including forwarding operation $O_{i,j}$:

$$D_{i,j} = (j-1)(d^{prop} + d^{proc}) + \sum_{k=1,\dots,j} d_{i,j}^{trans}$$

- Zero-queuing of TSN packets: For each pair of conflicting transmission operations $O_{i,k}$ and $O_{j,l}$ over same egress port $R(O_{i,k}) = R(O_{j,l})$
  - $t_j - t_i \geq D_{i,k} - (D_{j,l-1} + d^{prop} + d^{proc})$ or
  - $t_i - t_j \geq D_{j,l} - (D_{i,k-1} + d^{prop} + d^{proc})$

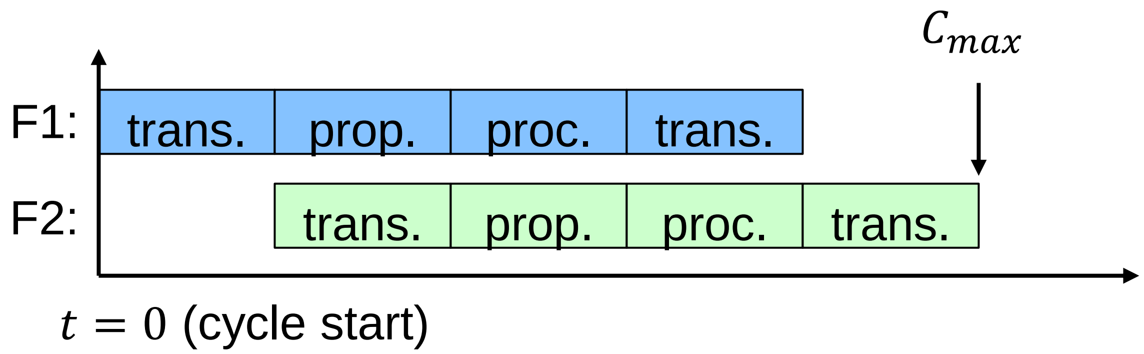
Disjunction can be translated to conjunction as required by ILP (see paper)



ILP Formulation (4)

Objective

Minimize flowspan C_{max}, i.e., finishing time of latest time-sensitive flow in cycle: $\min C_{max}$

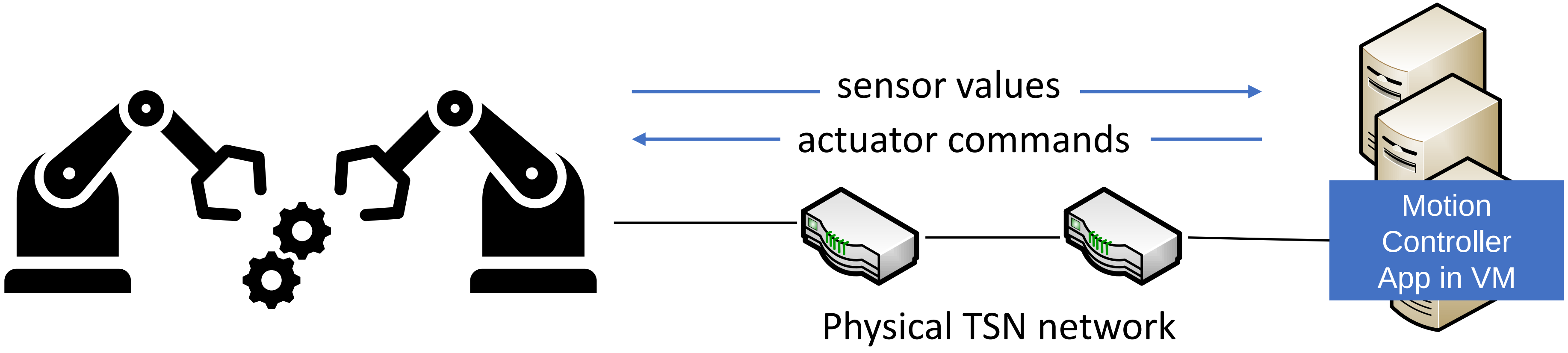


- ➔ Compact schedule: time-sensitive (TS) flows not scattered
- ➔ Higher chance of “back-to-back” TS traffic helps minimizing Gate Control List size
- ➔ Larger continuous space for best-effort traffic; higher best-effort throughput

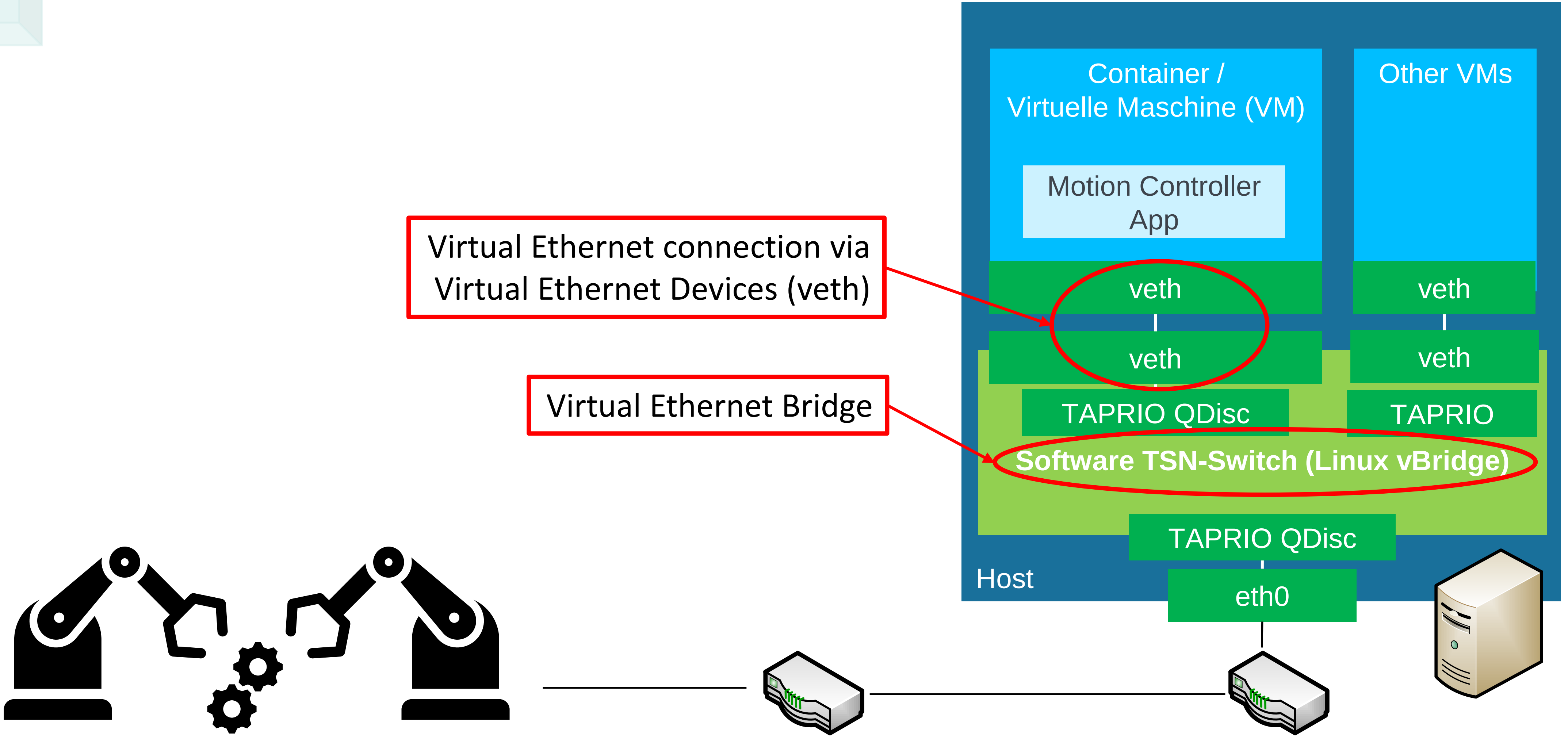
Time-Aware Shaper on Host

- ❑ Real-time systems can offload functions to Edge cloud computing infrastructure
- ❑ Edge cloud host runs virtual network to connect virtual machines, containers, etc.

How to connect VMs, containers, etc. to the TSN network on the host?



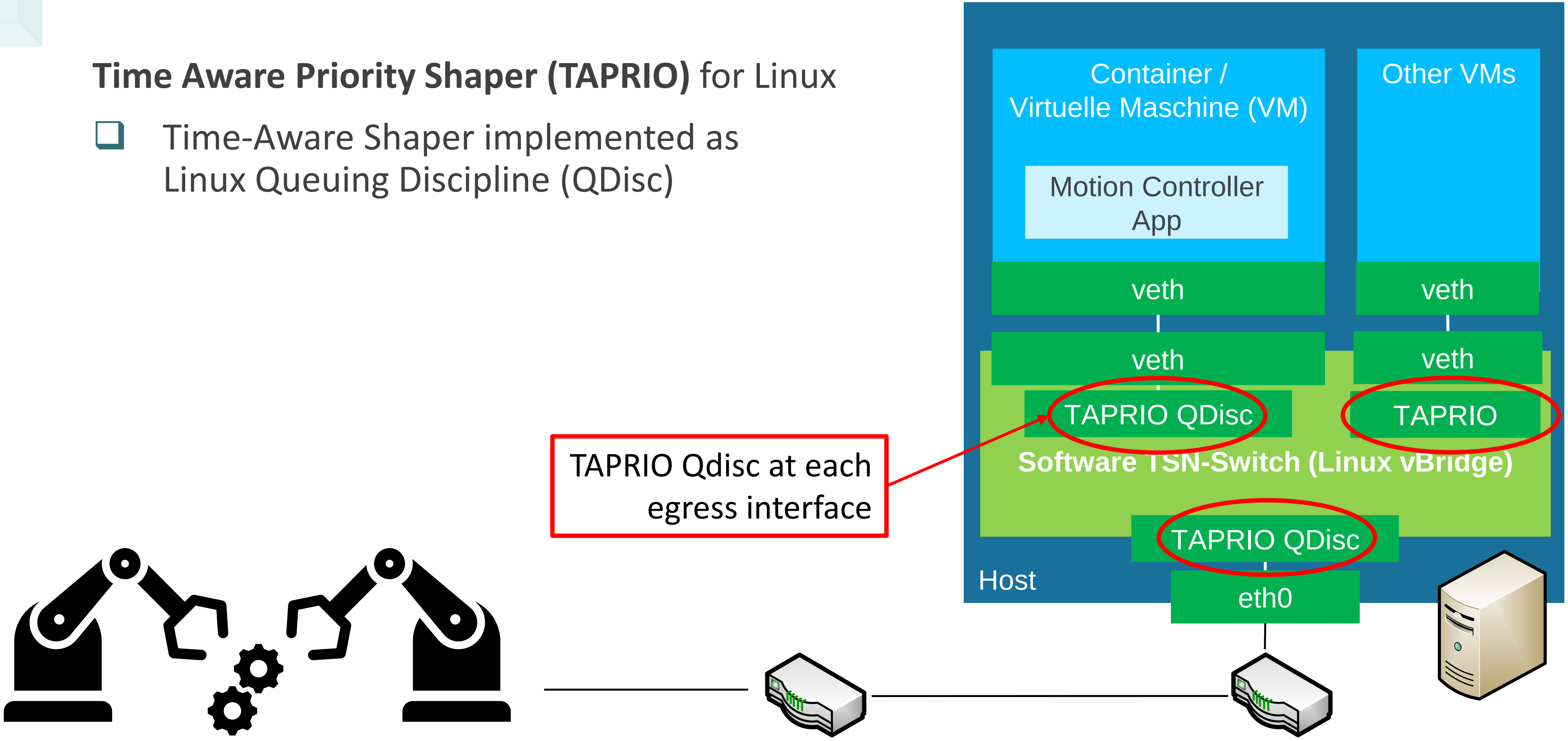
Virtual Ethernet-Switch + Software Time-Aware Shaper (TAPRIO QDisc) on Linux



Virtual Ethernet-Switch + Software Time-Aware Shaper (TAPRIO QDisc) on Linux

Time Aware Priority Shaper (TAPRIO) for Linux

- Time-Aware Shaper implemented as Linux Queuing Discipline (QDisc)



TAPRIO Configuration

Configuration with traffic control command (tc)

```
$ 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
```

Detailed information how to configure a software TSN bridge with Time-Aware Shaper:
DETERMINISTC6G blog: https://blog.deterministic6g.eu/posts/2024/11/02/software_tsn_switch.html

TAPRIO Configuration

Configuration with traffic control command (tc)

```
$ tc qdisc replace dev enp2s0f1 parent root
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
```

Gate Control List (Schedule):

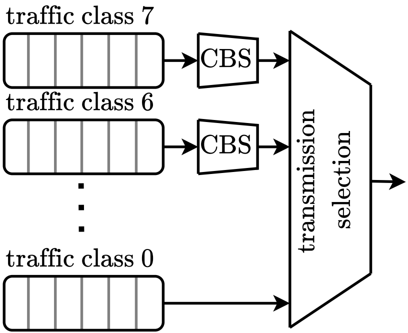
- S 01: set gate state
- 01: gate 0 open (first bit)
- 02: gate 1 open (second bit)
- 800000: duration of entry in nano-secs

Detailed information how to configure a software TSN bridge with Time-Aware Shaper:
DETERMINISTC6G blog: https://blog.deterministic6g.eu/posts/2024/11/02/software_tsn_switch.html

Agenda

- Motivation
- TSN Overview
- TSN Shapers:
  - Time-Aware Shaper
  - **Credit-based Shaper**
- Reliability
- Conclusion

IEEE 802.1Qav Credit-Based Shaper (CBS)

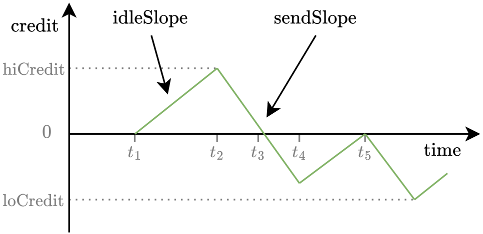


Possible Configuration:

- CBS manages *credit per queue*
- Can, optionally, be combined with TAS

Policy: *The first frame of CBS queue i is eligible for transmission if its credit is greater than or equal to zero and no other frame of queue $j > i$ is eligible.*

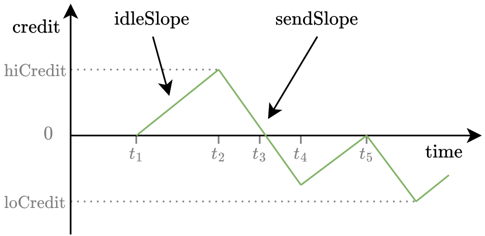
IEEE 802.1Qav Credit-Based Shaper (CBS)



Managing Credits Per Queue:

- Credit increases while frames are enqueued but not eligible for transmission
  - e.g., while having negative credit or while higher-priority traffic is transmitting
- Credit decreases while frames of the CBS queue are transmitted

IEEE 802.1Qav Credit-Based Shaper (CBS)



Example:

- At t_1, three frames arrive: f_1, f_2, and f_3
- f_1 is delayed until t_2 by higher-priority traffic $\rightarrow$ **credit increases**
- Then, f_1 and f_2 start their transmission at t_2 and t_3 resp. $\rightarrow$ **credit decreases**
- f_3 has to wait until t_5 to accumulate enough credit

Guarantees of CBS

- Credit-based Shaper **guarantees per queue service**
  - No starvation of queue (if slopes are correctly configured)
  - Bursts limited by accumulated credit at each hop
- Formal analysis of **per flow** guarantees possible, but complex:
  - “Intense calculations and knowledge of topology required for per flow guarantees.“ [Norman Finn (Cisco Systems)]
  - Approaches (not discussed here):
    - Luxi Zhao, Paul Pop, Zhong Zheng, Qiao Li: *Timing Analysis of AVB Traffic in TSN Networks using Network Calculus*. In Proceedings of IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), 2018
    - Jonas Diemer, Jonas Rox, Rolf Ernst: *Modeling of Ethernet AVB Networks for Worst-Case Timing Analysis*. IFAC Proceedings Volumes, 45(2), 2012

CBS Analysis

Selection of send slope and idle slope: $sendSlope = idleSlope - portTransmitRate$

- Determine loCredit and hiCredit
  - $loCredit = sendSlope \times (maxFrameSize/portTransmitRate)$
  - $hiCredit = idleSlope \times maxInterferenceTime$
    - hiCredit depends on higher-priority traffic

- Maximum burst size:

$$B_{max} = \texttt{portTransmitRate} \times \frac{\texttt{hiCredit} - \texttt{loCredit}}{\texttt{portTransmitRate} - \texttt{idleSlope}}$$

Guarantees of CBS

- Credit-based Shaper **guarantees per queue service**
  - No starvation of queue (if slopes are correctly configured)
  - Bursts limited by accumulated credit at each hop
- Formal analysis of **per flow** guarantees possible, but complex:
  - “Intense calculations and knowledge of topology required for per flow guarantees.“ [Norman Finn (Cisco Systems)]
  - Approaches (not discussed here):
    - Luxi Zhao, Paul Pop, Zhong Zheng, Qiao Li: *Timing Analysis of AVB Traffic in TSN Networks using Network Calculus*. In Proceedings of IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), 2018
    - Jonas Diemer, Jonas Rox, Rolf Ernst: *Modeling of Ethernet AVB Networks for Worst-Case Timing Analysis*. IFAC Proceedings Volumes, 45(2), 2012

Agenda

- Motivation
- TSN Overview
- TSN Shapers:
  - Time-Aware Shaper
  - Credit-based Shaper
- **Reliability**
- Conclusion

DETERMINISTIC6G

Common Failure Models

Type of Failure	Description
Crash Failure	A network entity crashes unannounced
Omission Failure	A frame is unexpectedly dropped
Timing Failure	A network entity is not synchronized correctly
Babbling Idiot	A network entity generates unexpected traffic
Byzantine Failure	A network entity deviates from its expected behavior arbitrarily

In general, we have

$$\begin{aligned}\text{Crash Failure} \subseteq \text{Omission Failure} \subseteq \text{Byzantine Failure}\\ \text{Timing Failure} \subseteq \text{Byzantine Failure}\\ \text{Babbling Idiot} \subseteq \text{Byzantine Failure}\end{aligned}$$

Reliability Considerations

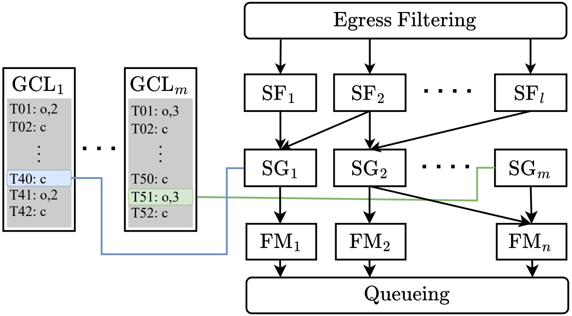
Stream Reliability: *Percentage of frames that arrive at their listener within the stream's QoS requirements.*

There are two aspects to ensure/improve stream reliability:

	Protecting High-Priority Traffic	Improving Stream Reliability
Problem	Timing Failures & Babbling Idiots	Crash Failures & Omission Failures
Solution	IEEE 802.1Qci PSFP	IEEE 802.1CB FRER

DETERMINISTIC6G

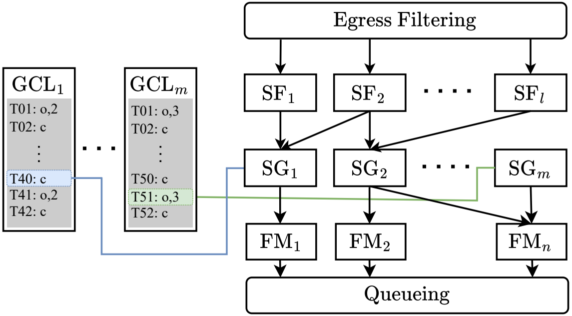
IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP)



Stream Filter (SF) identifies incoming frames by

- src/dst MAC address,
- PCP value, and VID.

IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP)



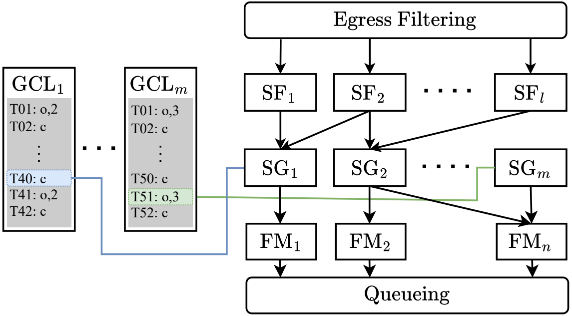
Stream Gate (SG) is managed by GCL, similar to TAS:

- closed: discard frame
- open: enqueue with IPV

Internal Priority Value (IPV)

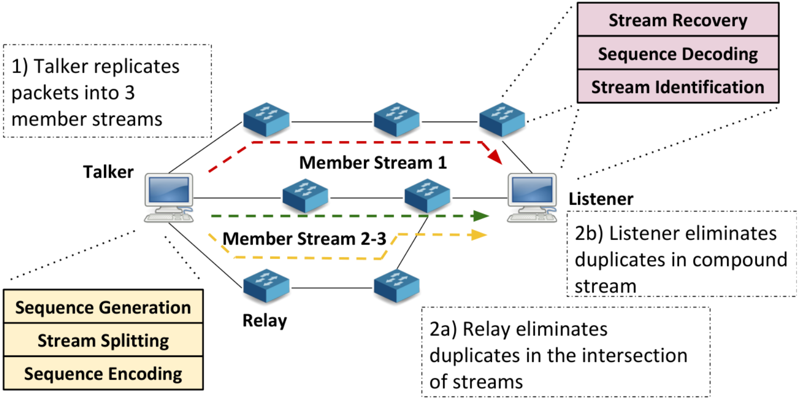
- can change egress queue,
- w/o overwriting PCP field!

IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP)



Flow Meter (FM) may discard frames that violate a token bucket specification $TB(r, b)$.

IEEE 802.1CB Frame Replication and Elimination (FRER)



[Source: D. Ergenç, M. Fischer, On the Reliability of IEEE 802.1CB FRER, IEEE INFOCOM 2021]

Agenda

- Motivation
- TSN Overview
- TSN Shapers:
  - Time-Aware Shaper
  - Credit-based Shaper
- Reliability
- **Conclusion**

Conclusion

- Many critical cyber-physical systems require dependable real-time communication
  - Bounds on end-to-end delay and delay variation
- TSN as enabling technology
  - Time-Aware Shaper
    - Clock-driven (schedule)
    - Schedules complex (NP-hard) to calculate: ILPs, CP, SMT, heuristics, …
    - Also available for Edge Cloud (Linux TAPRIO)
  - Asynchronous: Credit-based Shaping, Asynchronous Traffic Shaper
    - Analysis of stream-based guarantees complex (e.g. Network Calculus)
  - Reliability
    - Per-Stream Filtering and Policing
    - Frame Replication and Elimination

DETERMINISTIC6G Grant Agreement No. 101096504

The DETERMINISTIC6G project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101096504.

If you need further information, please contact the coordinator:

Janos Harmatos, ERICSSON

E-Mail: coordinator@deterministic6g.eu

or visit: www.deterministic6g.eu

@DETERMINISTIC6G   DETERMINISTIC6G