Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Institute of Computer Science
Chair of Communication Networks
Prof. Dr. Tobias Hoßfeld

Time-Sensitive Networking
For Plug-and-Play Networks:
How can the tools be used in practice?

Tobias Hoßfeld
tobias.hossfeld@uni-wuerzburg.de

Alexej Grigorjew
alexej.grigorjew@uni-wuerzburg.de

From Field Busses to Ethernet

Bus systems

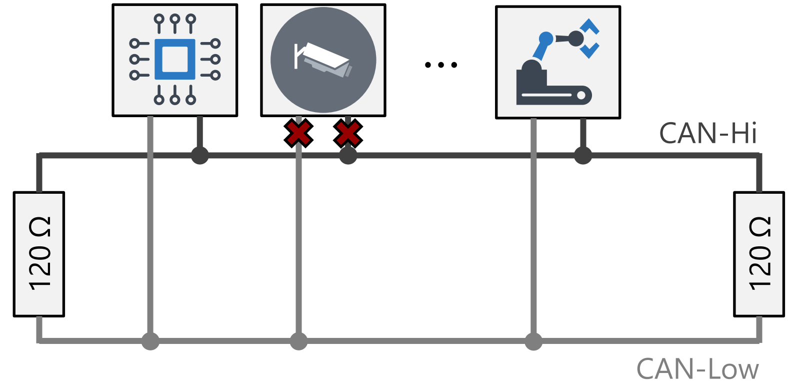
- Profibus, CAN, FlexRay
- “Simple“ behavior
- Limited bandwidth

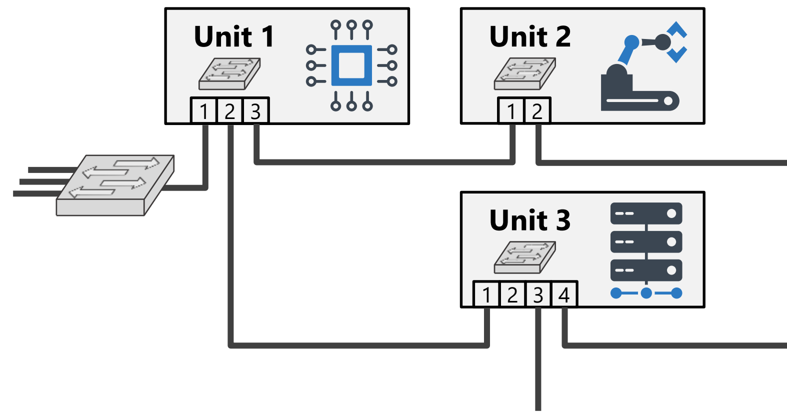
Ethernet-based systems

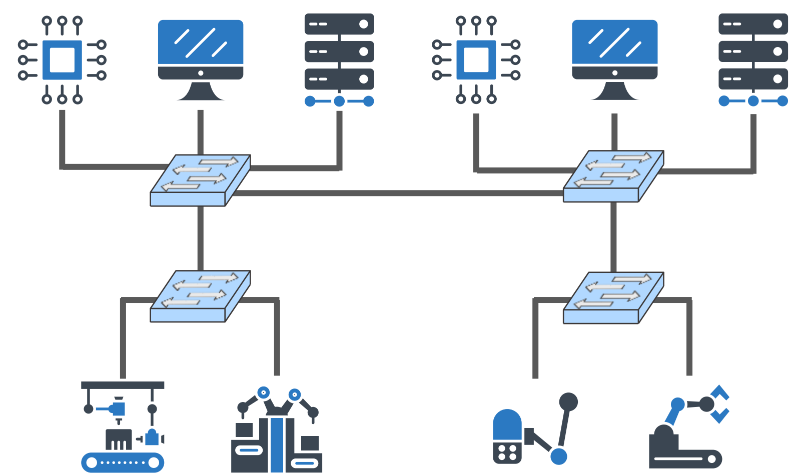
- PROFINET, EtherCAT, AFDX
- More bandwidth
- Limited features
- Often used like a bus
- Incompatible

Time-Sensitive Networking

- Enhancements of Ethernet standard
- Unified toolset for determinism
- More complex (queuing!)
- Shapers, Schedulers, Filters, …





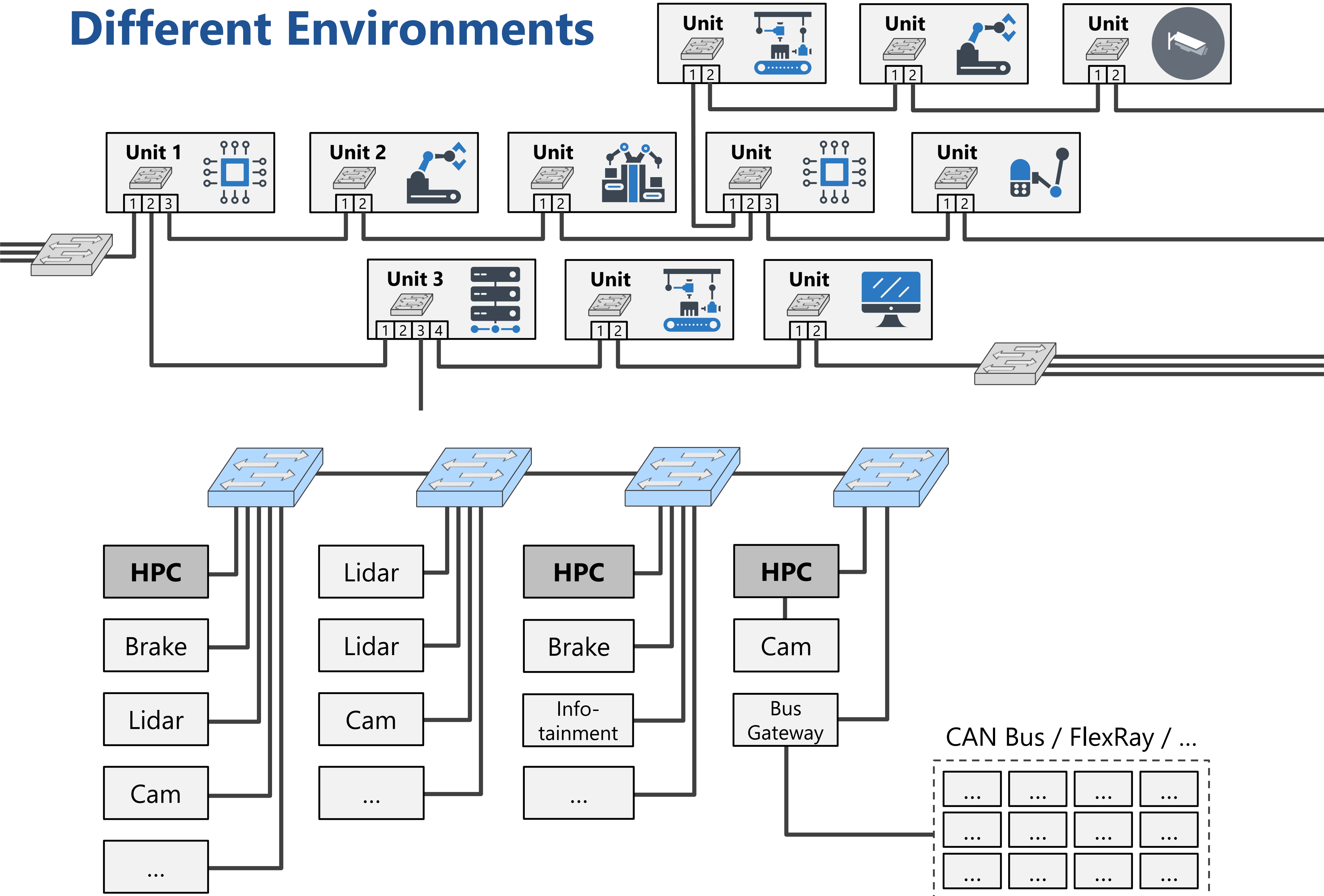


Definition of Deterministic Networking

How would you define deterministic transmission behavior?

- **Reliability**
  - 0% packet loss due to congestion
  - Redundancy and filters for further protection against failures
- **Bounded latency**
  - Per-hop latency?
  - End-to-end latency?
  - Is there even a difference?
- **Bounded jitter**
  - Individual frame delay jitter? $(max\ delay - min\ delay)$
  - Inter-arrival time jitter? $(timing\ frame_2 - timing\ frame_1)$

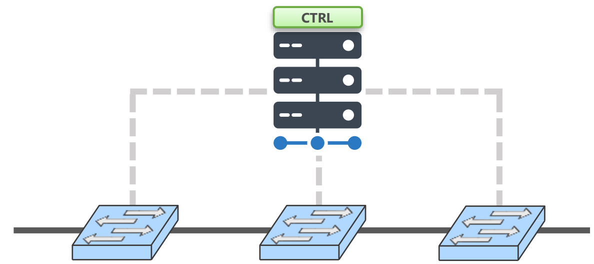
Different Environments

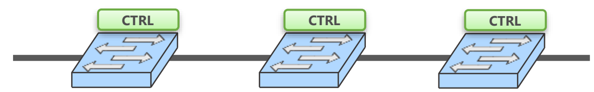


Network Architecture and Control

Control plane

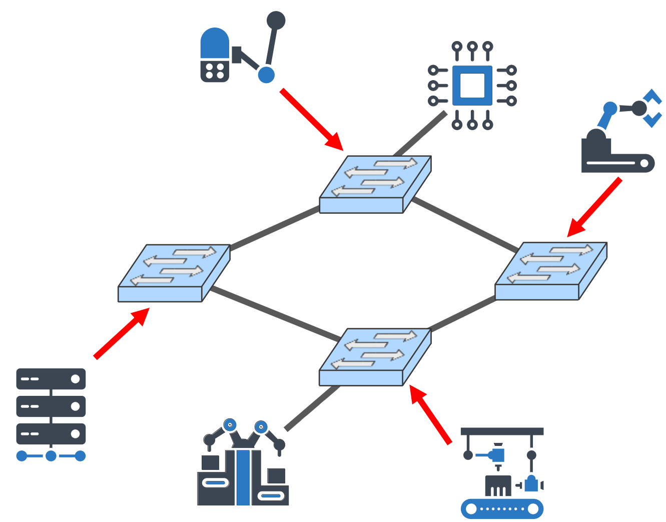
- Central controller
- Distributed in switches





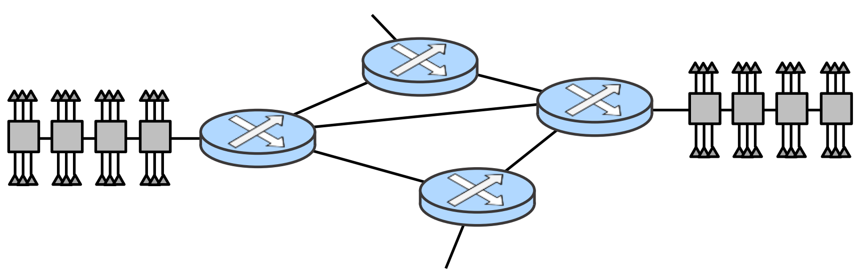
Dynamics

- Static network, fixed traffic demands
- Dynamic plug-and-play



Network size

- LAN
- WAN
- Interconnected Domains



What is Really Necessary for Determinism?

Time-Sensitive Networking – Main Components

Audio Video Bridging (802.1BA)	Fronthaul (802.1CM)	Industrial Automation (IEC/IEEE 60802)	Automotive In-Vehicle (P802.1DG)	Service Provider (P802.1DF)

Time Synchronization

Timing and Synchronization (802.1AS-2020)
Includes a profile of IEEE 1588
…

Ultra Reliability

Frame Replication & Elimination (802.1CB)
Path Control (802.1Qca)
Per-Stream Filtering (802.1Qci)
Time Synchronization (802.1AS-2020)
…

Bounded Low Latency

Priority Transmission Selection (802.1Q)
Credit Based Shaper (802.1Qav)
Preemption (802.3br, 802.1Qbu)
Scheduled Traffic (802.1Qbv)
Cyclic Queuing & Forwarding (802.1Qch)
Asynchronous Traffic Shaping (P802.1Qcr)
…

Dedicated Resources & API

Stream Reservation Protocol (802.1Qat)
TSN configuration (802.1Qcc)
YANG (802.1Qcp)
Link-local Registration Protocol (802.1CS)
Resource Allocation Protocol (P802.1Qdd)
…

Learning Goals of this Tutorial

We will be talking about TSN mechanisms:

1. What problems do they solve?
2. How do they influence the latency computation?
3. When should you be using which mechanism? [very roughly]

Agenda for the next hour:

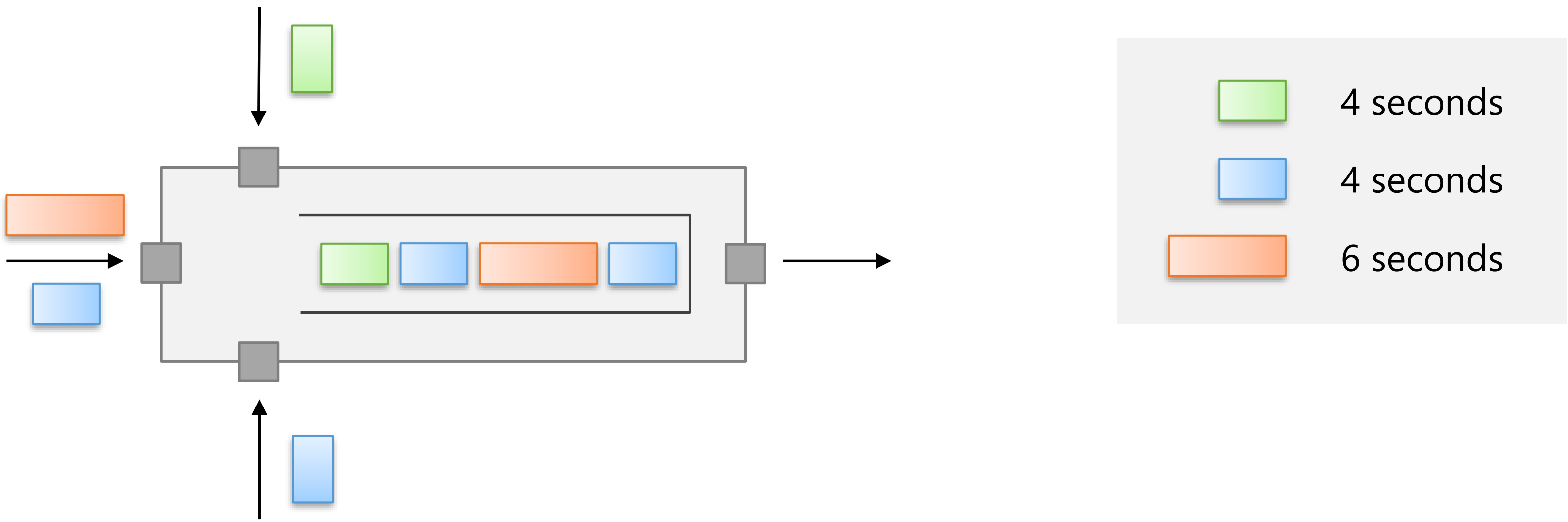
1. Traffic Model and Simple Examples
2. Challenges of Bounded Latency Computation
3. TSN Transmission Selection – Bounded Latency Mechanisms
   - Strict Priority
   - Per-Hop Reshaping (CBSA, ATS)
   - Synchronized Shapers (TAS, CQF)
4. Practical Example – Latency Computation in Python
5. Brief Comparison and Conclusion

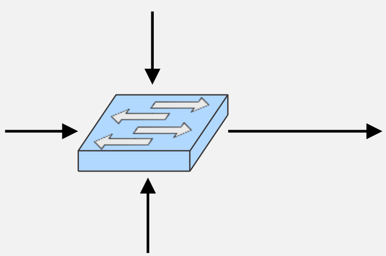
Section 1

TRAFFIC MODELS AND SIMPLE EXAMPLES

A Very Simple Analogy…

How long does it take the green car to cross the street (worst case)?





How many cars? → How many packets?

How fast do cars drive? → Packet size?

A Very Simple Analogy…

How long does it take the green car to cross the street (worst case)?

nds

nds

nds

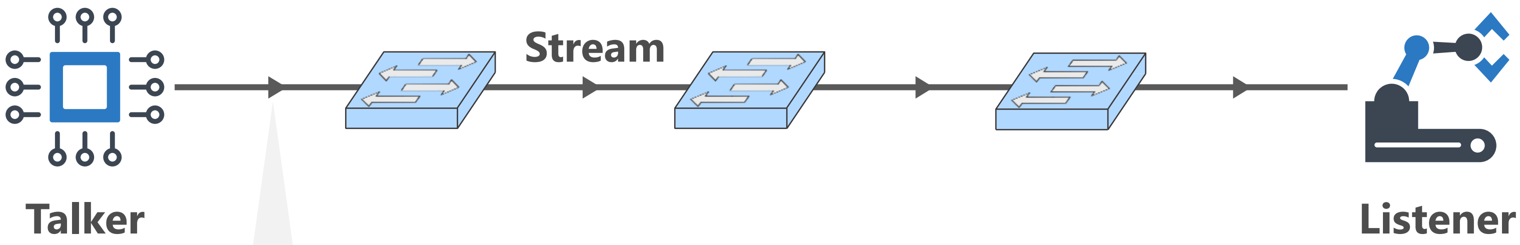
We learned (1):

Shaping is not required for latency bounds.
We just need some information about
the data volume and the order of transmission.

(But shaping does improve the efficiency.)

How fast do cars drive? → Packet size?

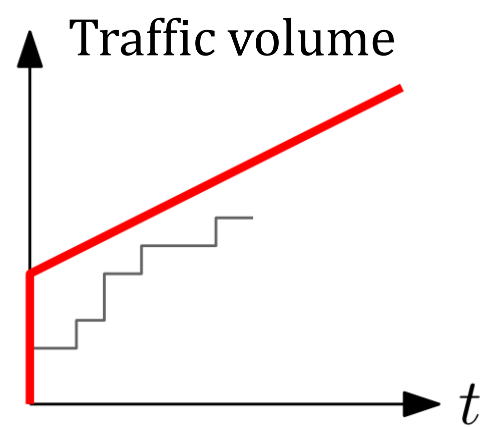
How much data can arrive at a given time?



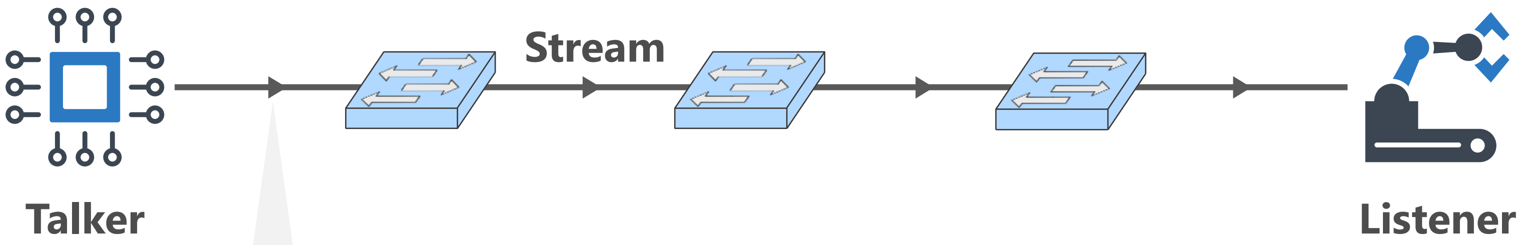
Example Stream

- **Data rate:** 67 kBit/s
- Amount of data in 1 h: 30.15 Mbyte
- Amount of data in 10 µs: 0.67 bit?

- Network traffic is not handled continuously
- Frames are received instantaneously as a whole
- **Burst:** How much data can be received in an instant?
- Typically: Maximum frame size of that stream



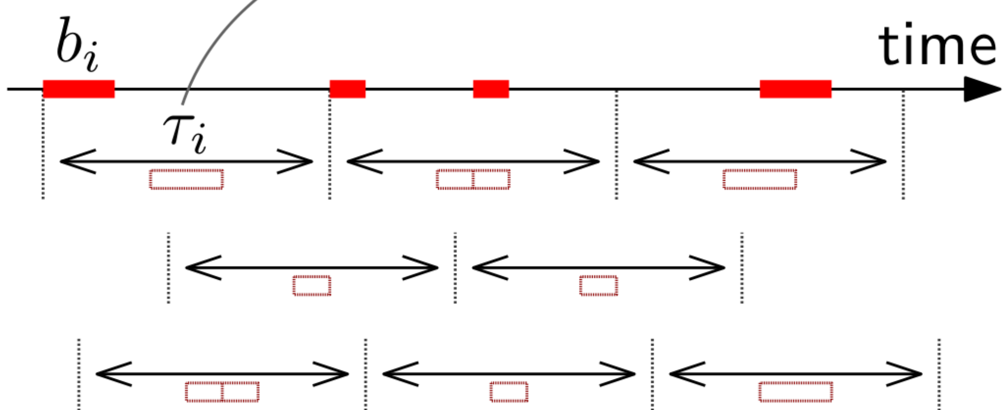
General Traffic Model

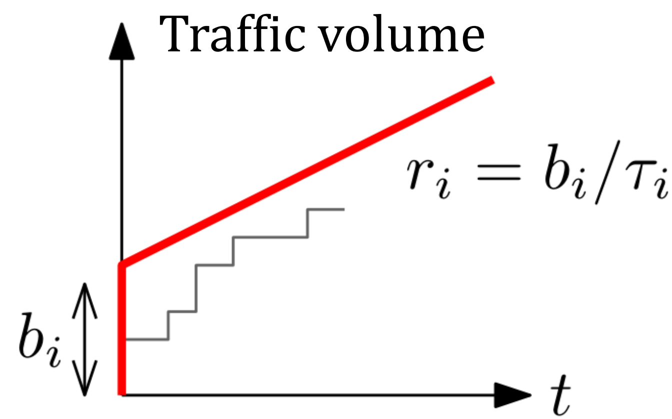


Traffic Specification of Stream *i*

- Traffic class (priority) p_i
- Min frame size $\check{\ell}_i$
- Max frame size $\hat{\ell}_i$
- Burst size b_i
- Data rate r_i

$\tau_i = b_i / r_i$ Burst interval





The MSRP Traffic Model

- In the SRP, burst and data rate are specified indirectly
- Number of frames during a measurement interval
- From IEEE Std 802.1Qcc, Section 35 (SRP):

IEEE Std 802.1Qcc-2018
IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—
Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements

Figure 35-11 specifies the encoding of the value for the TrafficSpecification TLV.

	Octet	Length
Interval		
MaxFramesPerInterval		
MaxFrameSize		
TransmissionSelection	9	1

$$Burst = MaxFramesPerInterval \cdot MaxFrameSize$$
$$Data\ Rate = Burst\ /\ Interval$$

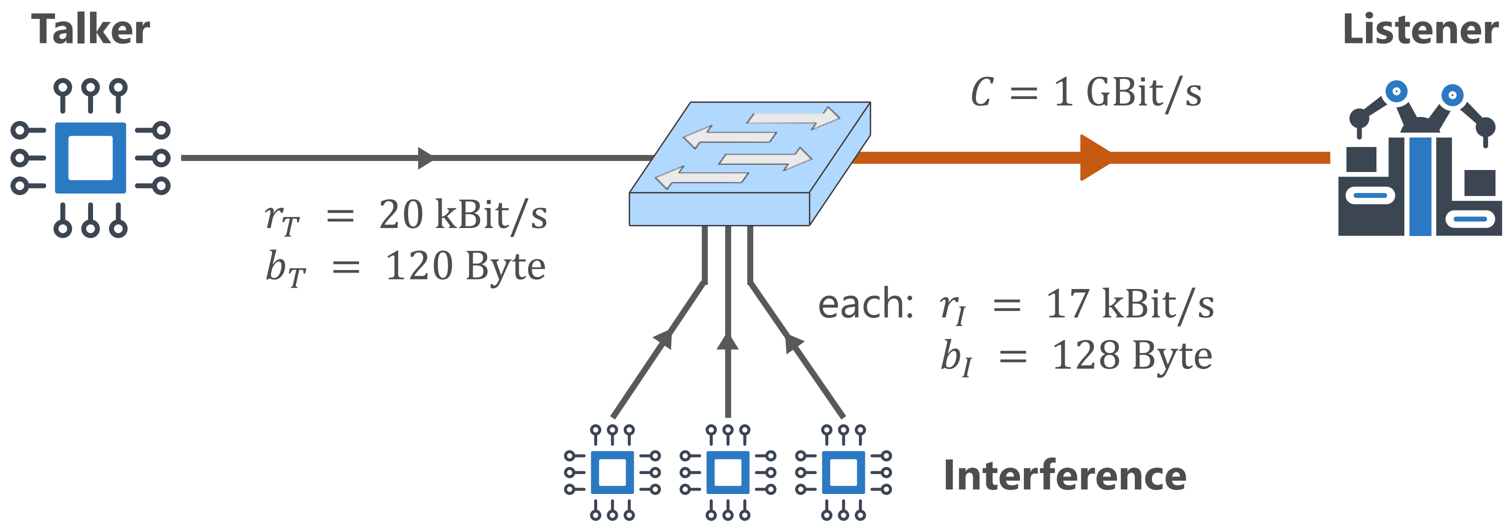
Figure 35-11—Value of TrafficSpecification TLV

Figure 35-12 specifies the encoding of the value for the TSpecTimeAware TLV. The presence of the optional TSpecTimeAware TLV is handled as specified in 46.2.3.5 for the presence of the TSpecTimeAware group.

	Octet	Length
EarliestTransmitOffset	1	4
LatestTransmitOffset	5	4
Jitter	9	4

Figure 35-12—Value of TSpecTimeAware TLV

Example Latency Bound



- **Assume**: only one frame of each stream is in flight at the same time
- **Worst case**: each interference frame arrives before talker
- Queuing delay: $d_T^Q = 3 \cdot 128 \cdot 8$ Bit / 1 Gbit/s $= 3$ µs
- Transmission delay: $d_T^T = 120 \cdot 8$ Bit / 1 Gbit/s $= 1$ µs

Section 2

CHALLENGES OF BOUNDED LATENCY COMPUTATION

Is it really that simple?

IEEE 802.1BA

AUDIO VIDEO BRIDGING (AVB) SYSTEMS — IEEE Std 802.1BA-2011

As discussed in L.3 of IEEE Std 802.1Q, there are a number of factors that increase the latency contribution of a Bridge. AVB Talkers and Bridges shall report each hop's worst-case latency via SRP. The following equation gives an example of how worst-case latency can be calculated for Class A streams:

$$\text{Max Latency} = t_{\text{Device}} + t_{\text{MaxPacketSize+IPG}} + (t_{\text{AllStreams}} - t_{\text{StreamPacket+IPG}}) \times \text{Rate/MaxAllocBand} + t_{\text{StreamPacket}}$$

where

$\mathbf{t_{Device}}$ = the internal delay of the device (in increments of 512 bit times)

NOTE 3—t_{Device} is an integral multiple of 512 bit times so that it scales with the speed of the media.

$\mathbf{t_{MaxPacketSize+IPG}}$ = the transmission time for a maximum size interfering frame (1522 octets to 2000 octets) plus its preamble and start of frame delimiter (SFD) (8 octets), and the following inter-packet gap (IPG) (12 octets)

$\mathbf{t_{StreamPacket}}$ = the transmission time for the maximum frame size of the stream that is being reserved, plus its preamble and SFD (8 octets)

$\mathbf{t_{StreamPacket+IPG}}$ = the transmission time for the maximum frame size of the stream that is being reserved, plus its preamble and SFD (8 octets) and the following IPG (12 octets)

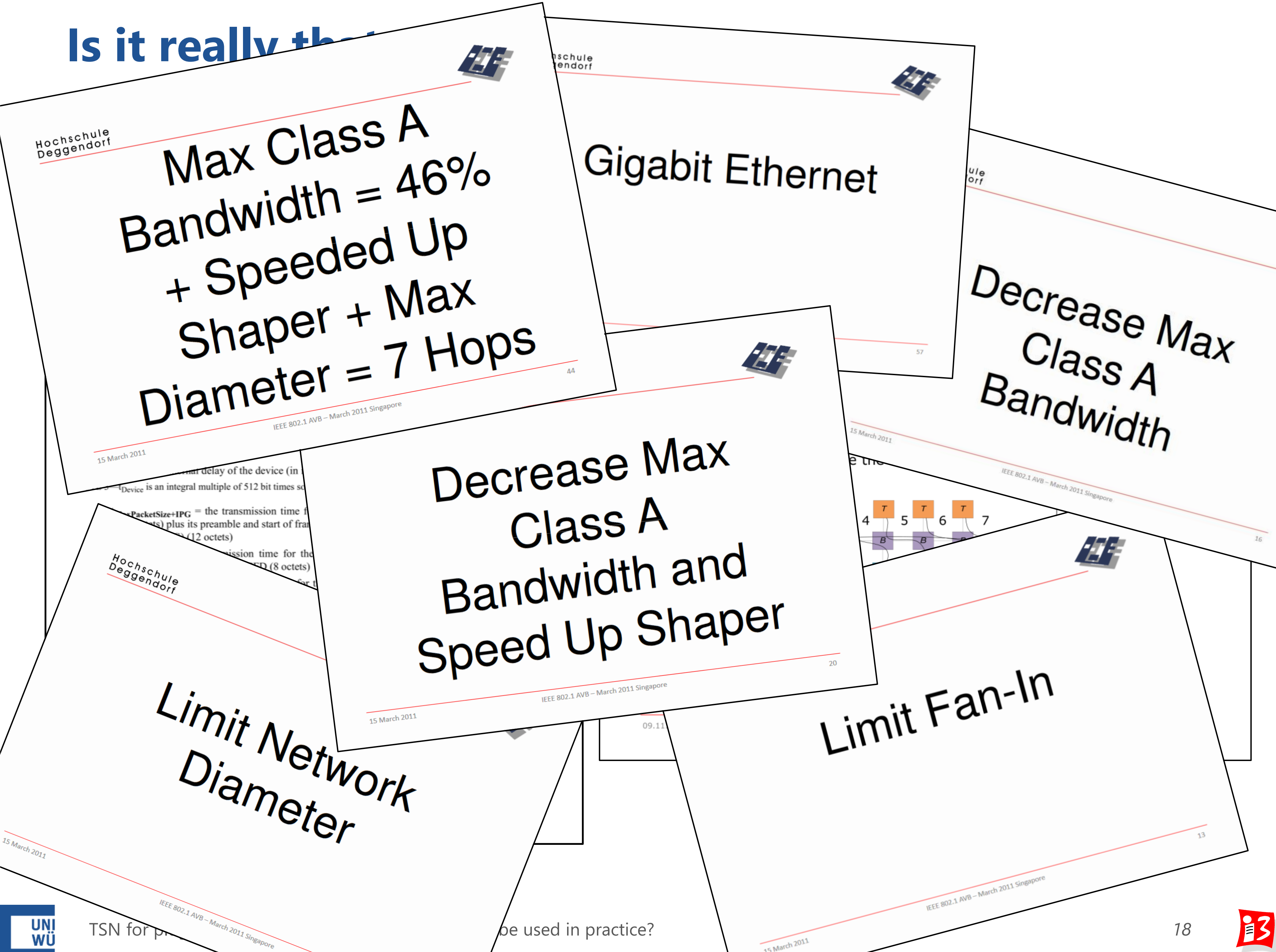
Rate = the transmission rate of the port

MaxAllocBand = maximum allocatable bandwidth, the maximum amount of bandwidth the AVB system is able to allocate for Class A streams on the port

$\mathbf{t_{Interval}}$ = the Class A observation interval or 125 µs

$\mathbf{t_{AllStreams}}$ = (MaxAllocBand × t_{Interval}) / tRate = the sum of the transmission times of all Class A stream frames the AVB System is able to allocate in an observation interval (125 µs) on a port

Is it really th



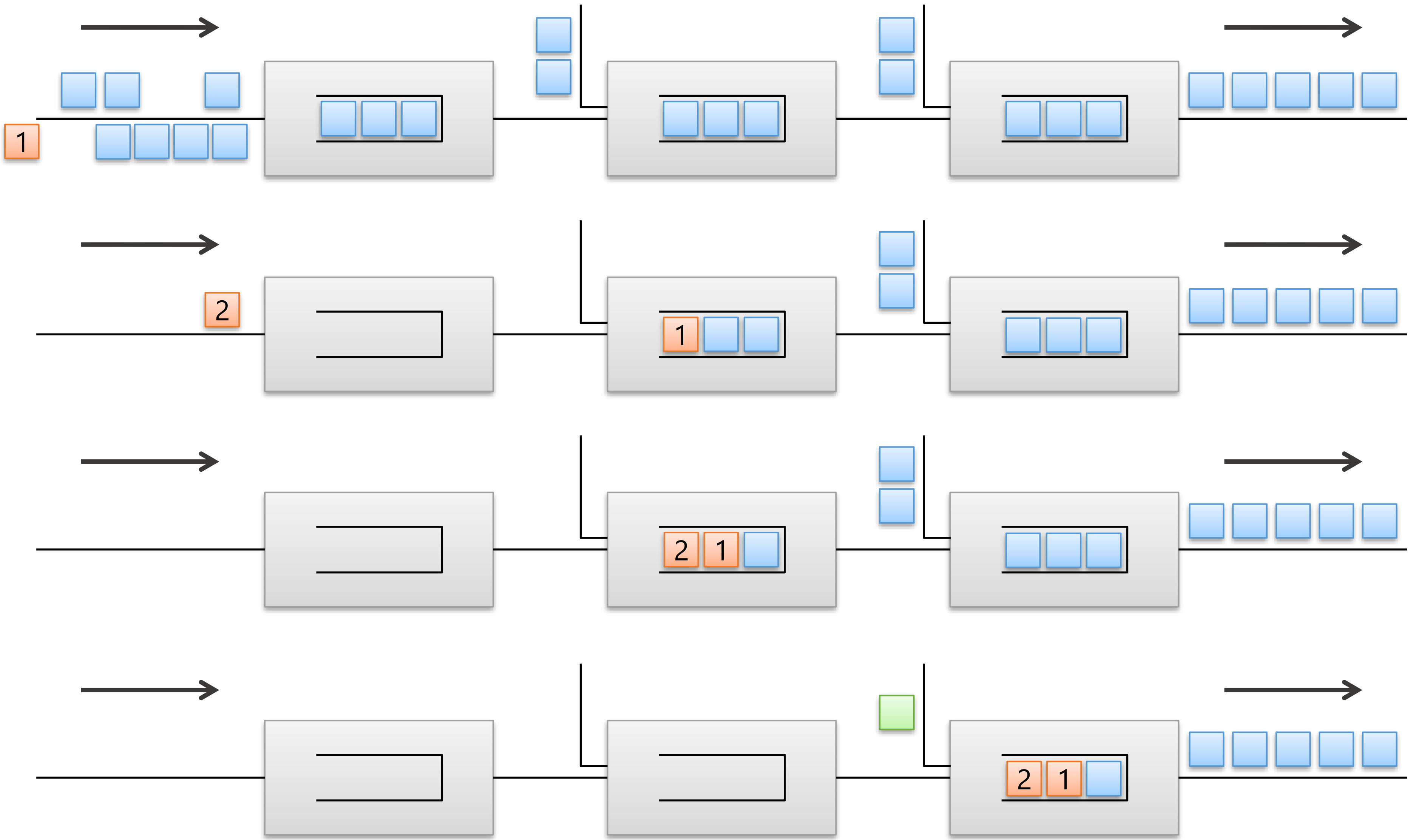
Is it really th…

We learned (2):

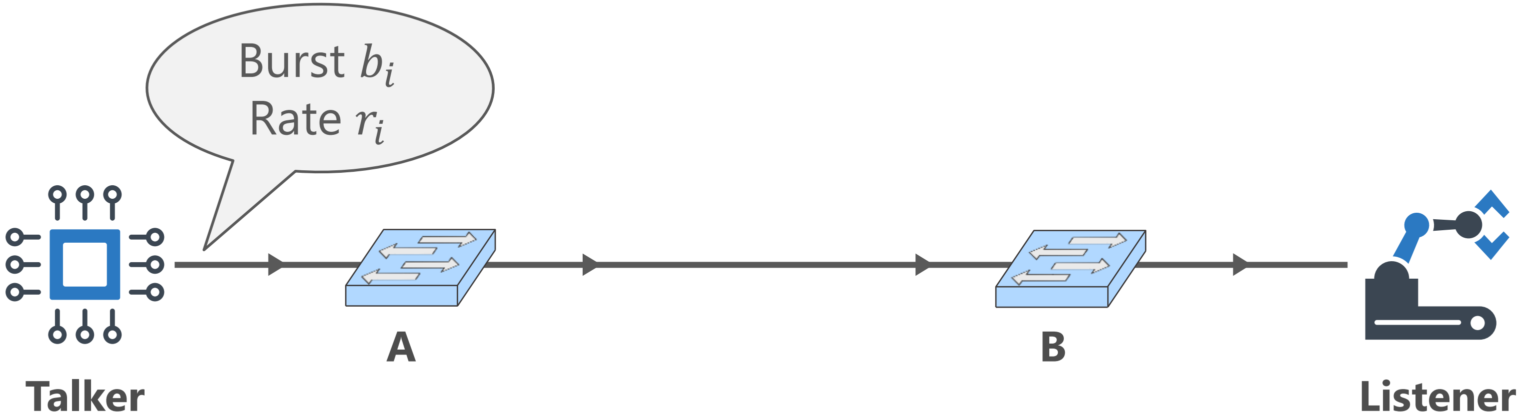
Don't just take any latency formula and use it.
Seek a formal proof for the upper bound.

(Simulation can only disprove a bound.
It cannot prove it.)

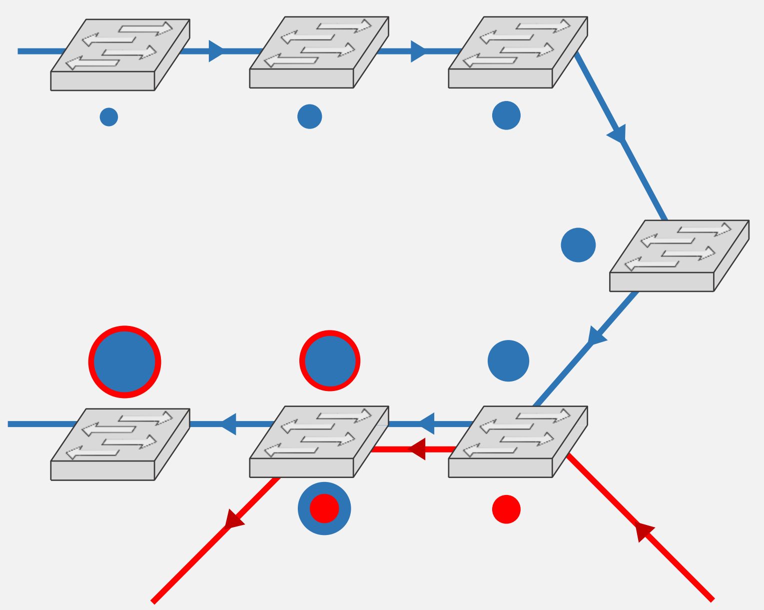
What can go Wrong?



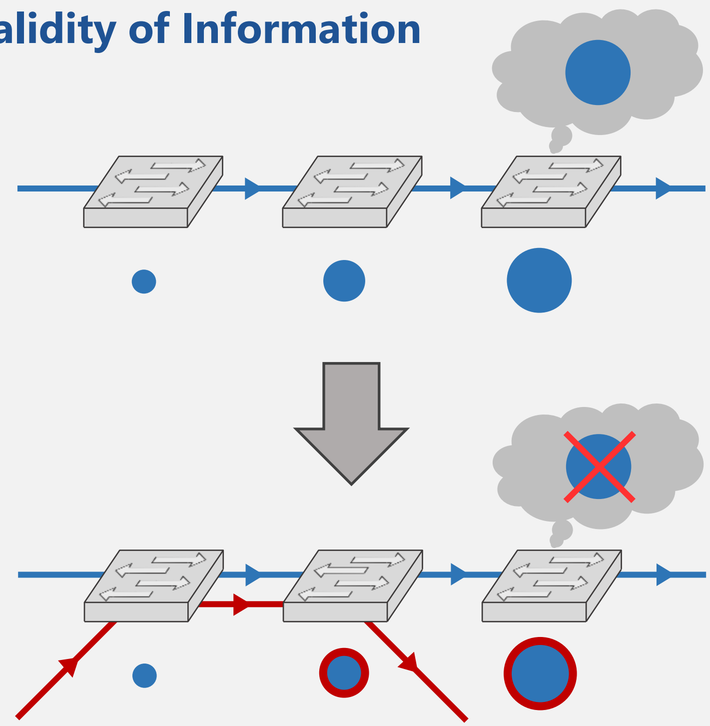
Two Challenges of Dynamic Stream Reservation



Snowball Effect

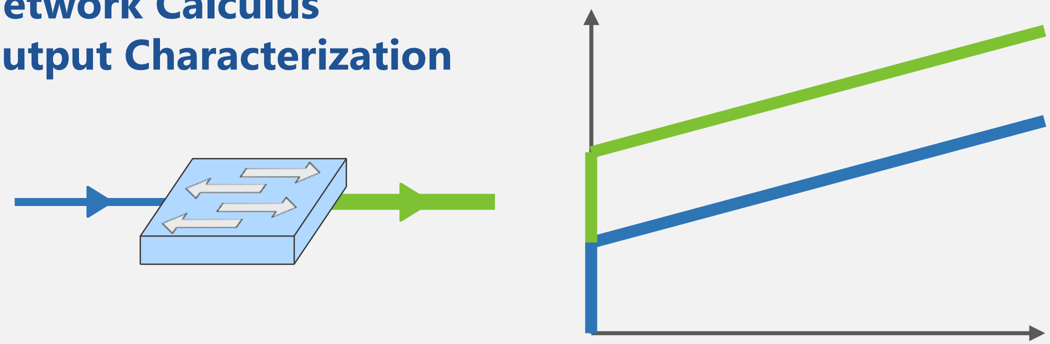


Validity of Information



Two Challenges of Dynamic Stream Reservation

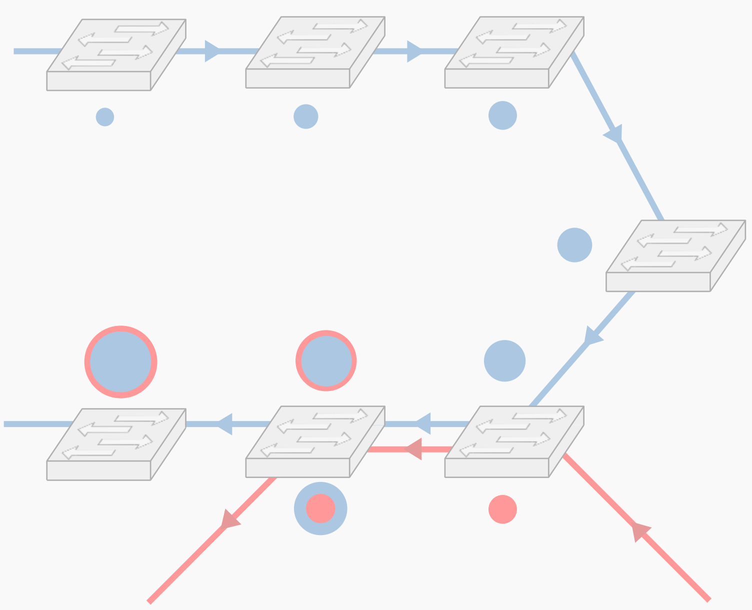
Network Calculus Output Characterization



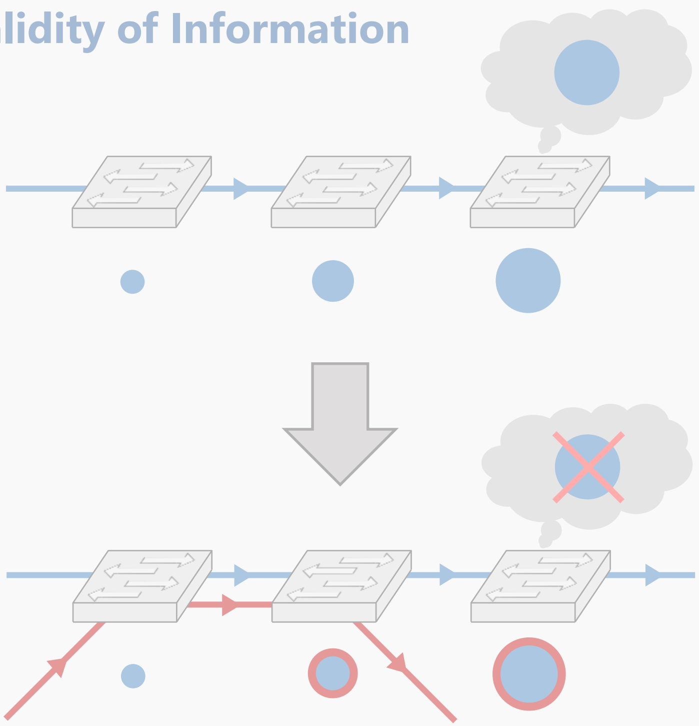
Out: $\alpha^*(t) = \rho t + \sigma + \rho T$

In: $\alpha(t) = \rho t + \sigma$

Snowball Effect

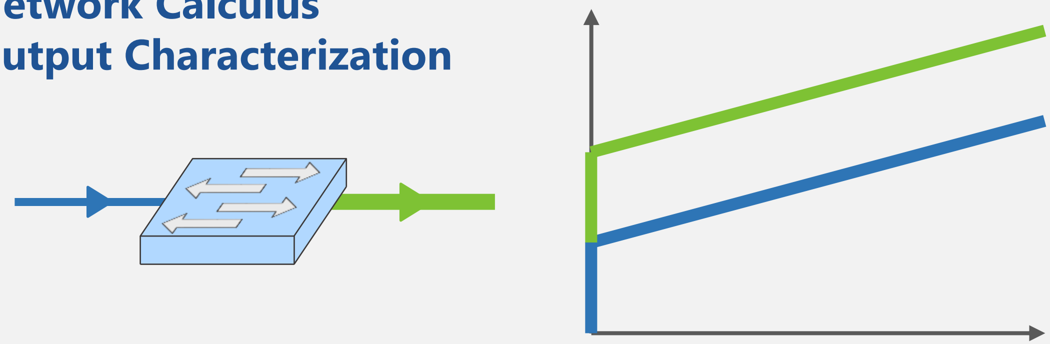


Validity of Information



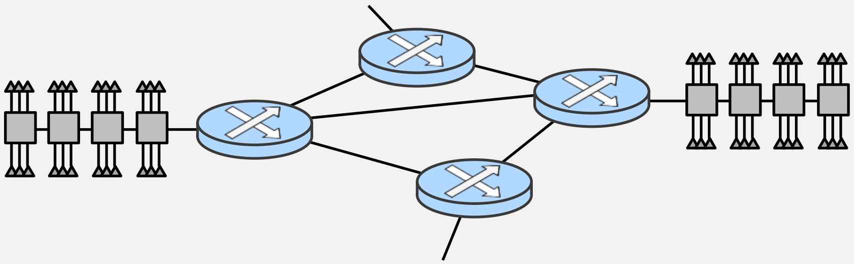
Two Challenges of Dynamic Stream Reservation

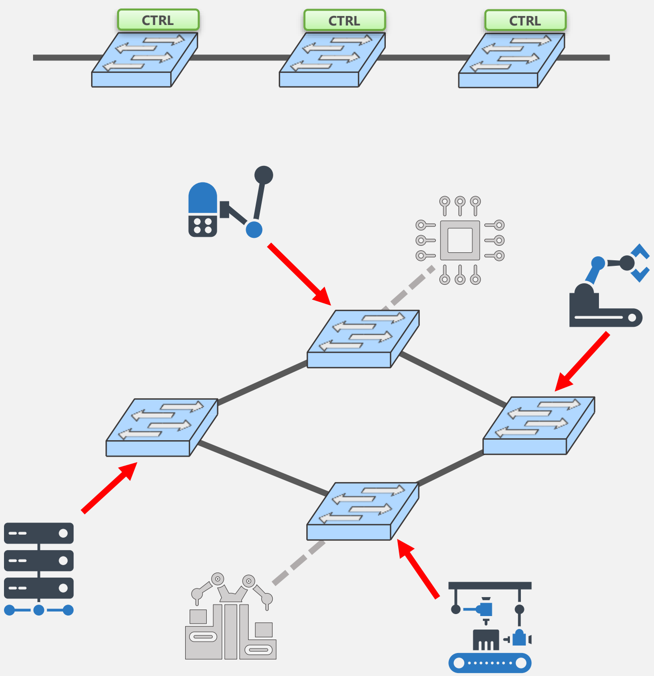
Network Calculus Output Characterization



Out: $\alpha^*(t) = \rho t + \sigma + \rho T$

In: $\alpha(t) = \rho t + \sigma$





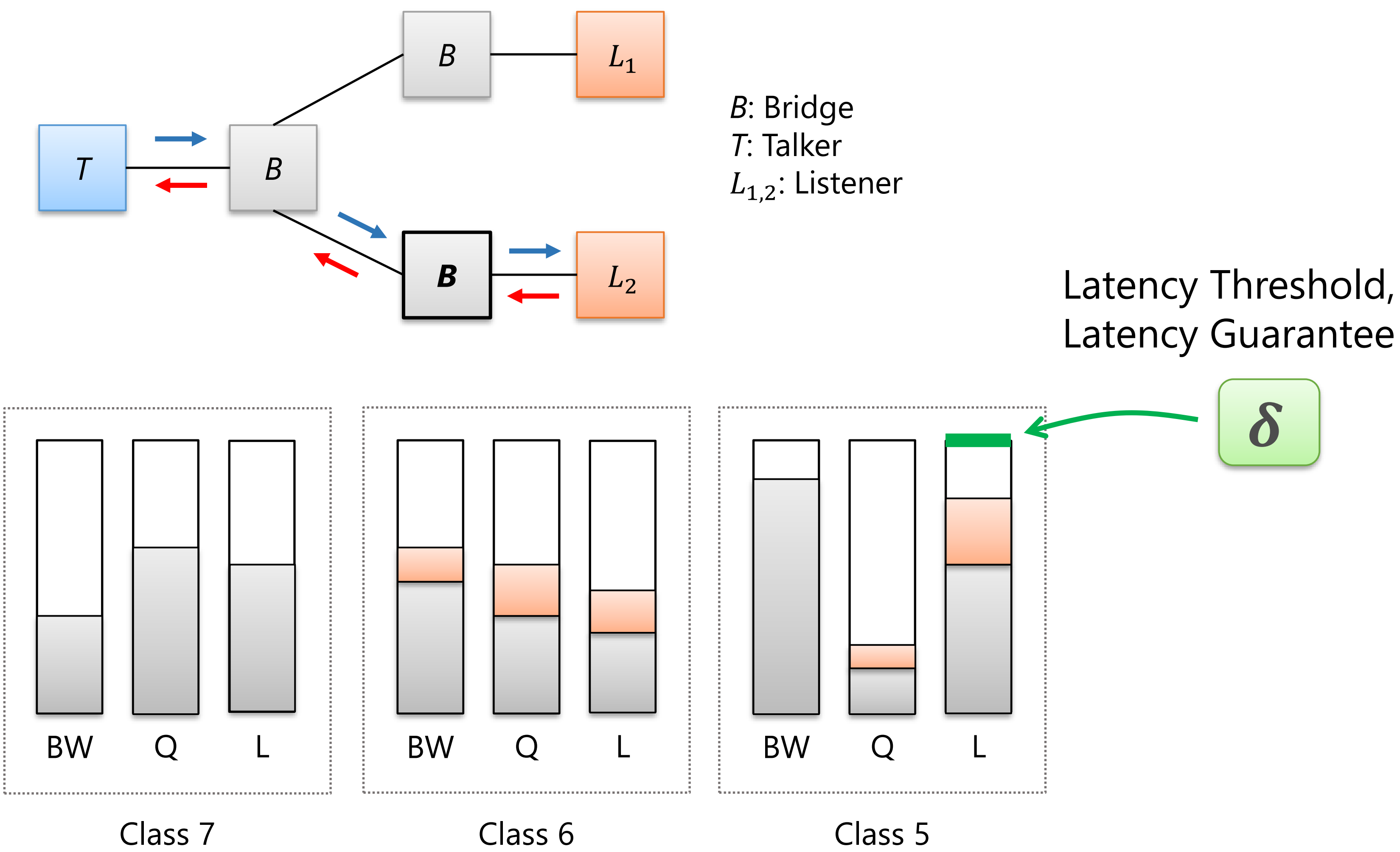
Remember our premise:

- Distributed control plane
- No central authority
- No "chatty" network
- But... changes in one part influence the entire network!

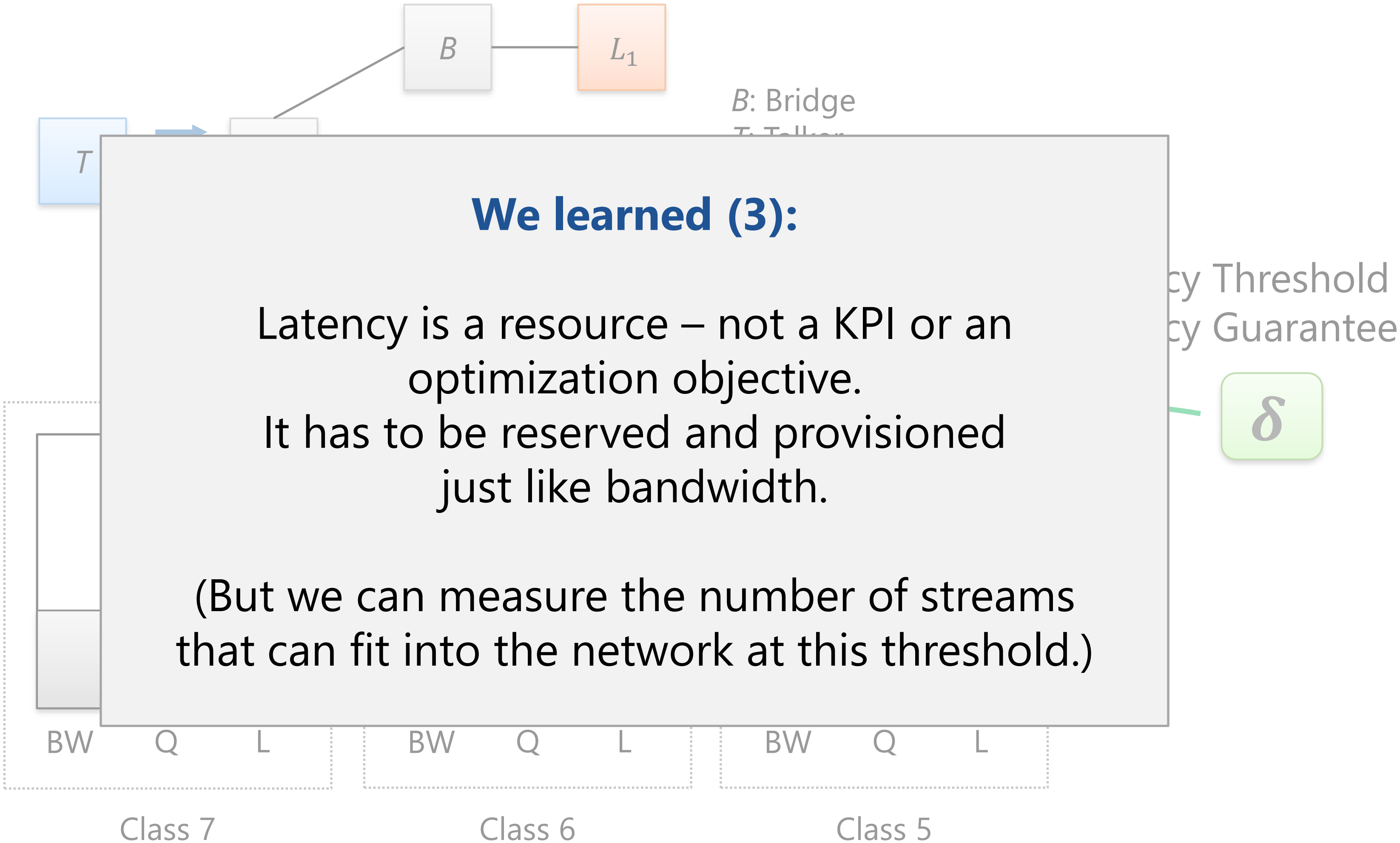
Section 3

TSN TRANSMISSION SELECTION – BOUNDED LATENCY MECHANISMS

Pre-configured Latency Thresholds



Pre-configured Latency Thresholds

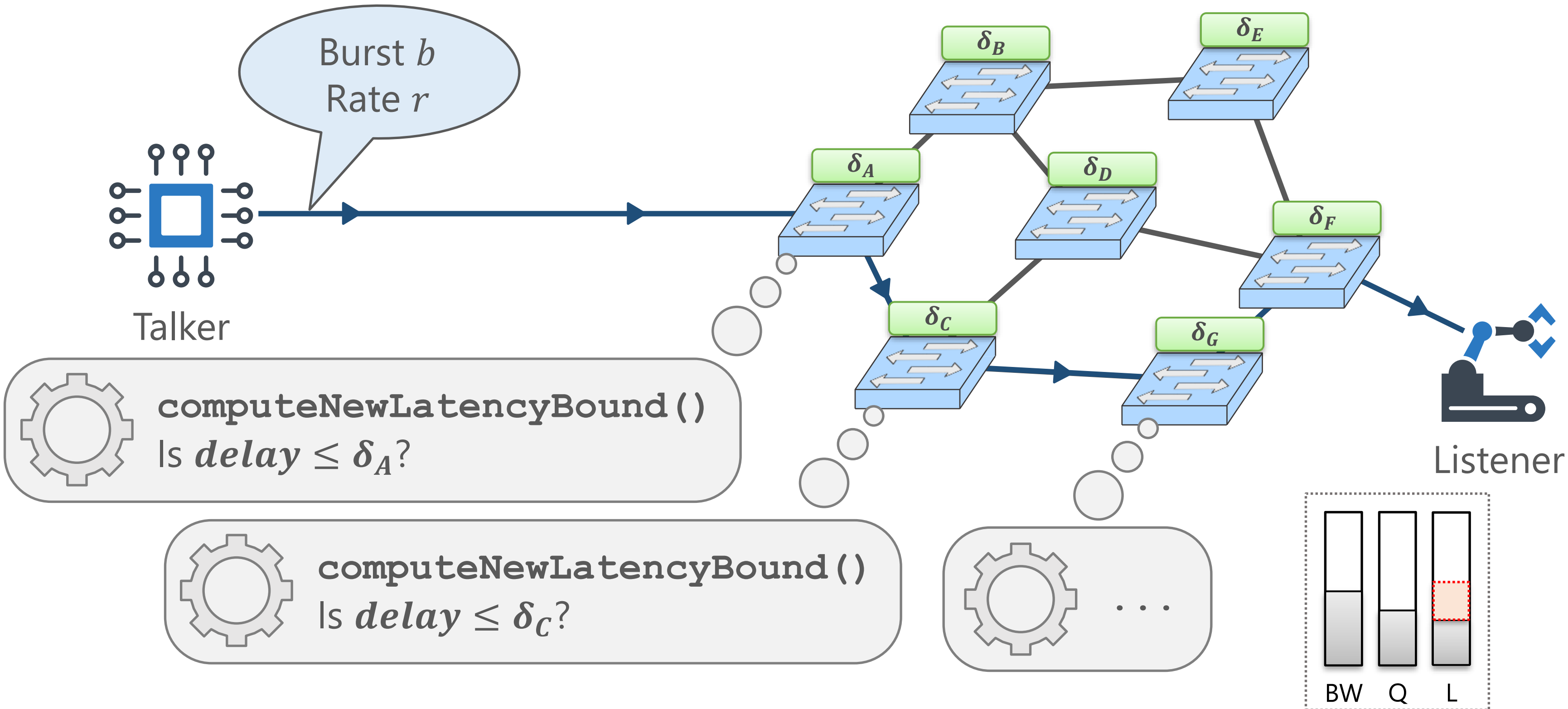


We learned (3):

Latency is a resource – not a KPI or an optimization objective.
It has to be reserved and provisioned just like bandwidth.

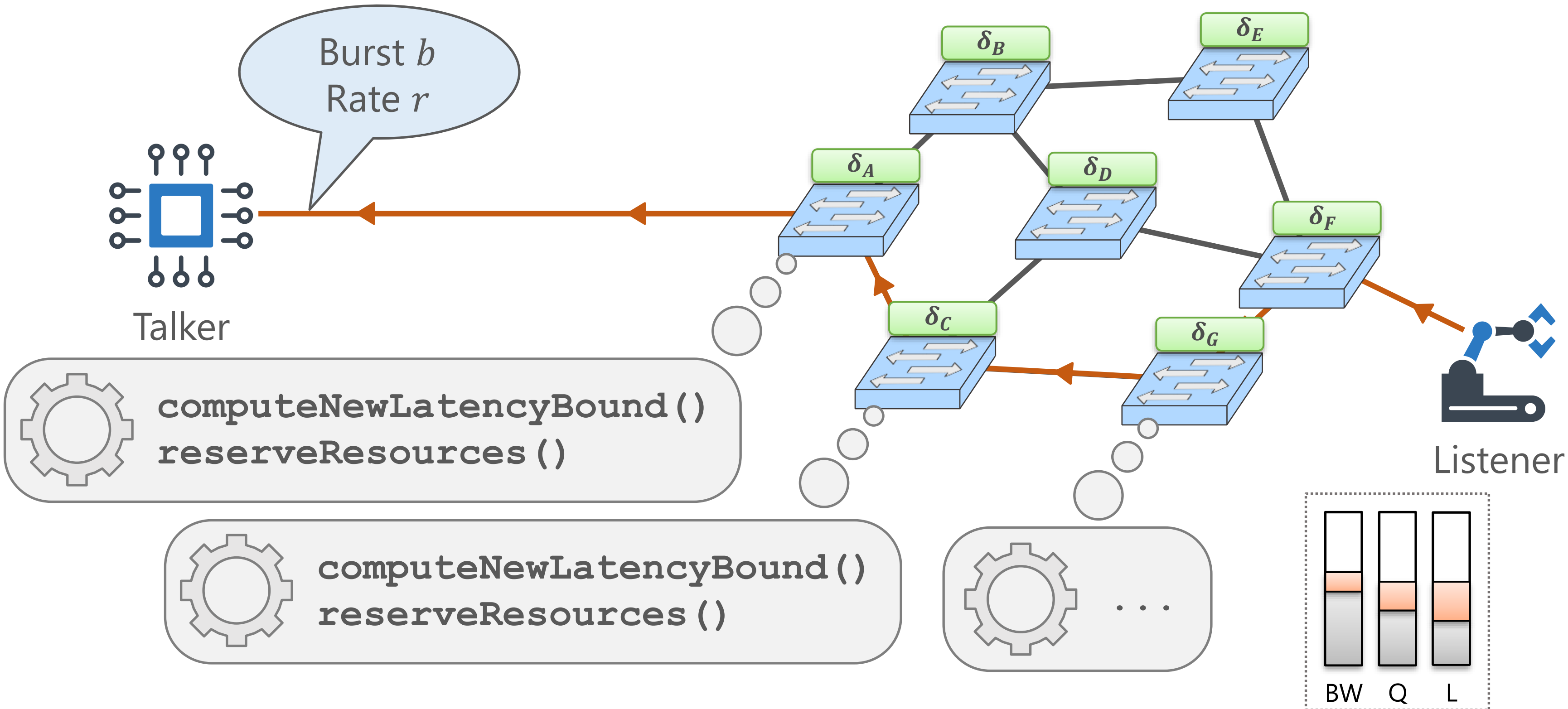
(But we can measure the number of streams that can fit into the network at this threshold.)

Dynamic Stream Reservation Process



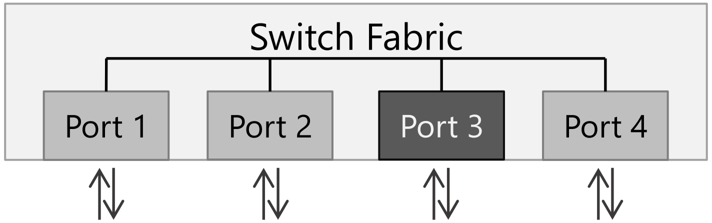
- Pre-configured delay guarantees δ_X
- Traffic specification
- Worst-case delay computation (based on shaper)

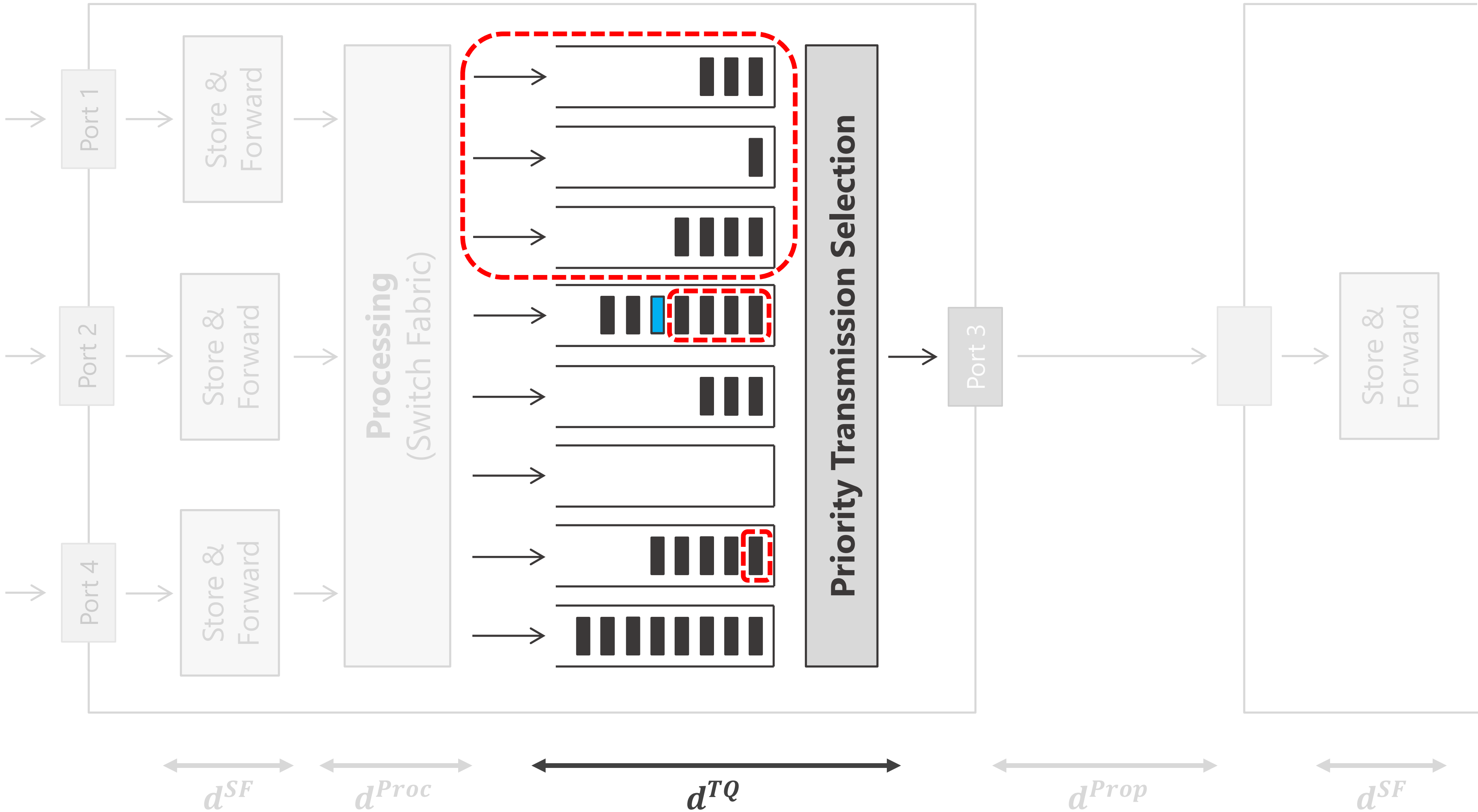
Dynamic Stream Reservation Process



- Pre-configured delay guarantees δ_X
- Traffic specification
- Worst-case delay computation (based on shaper) ...
- Listener subscription, reserve resources, …

General Bridge Latency Model





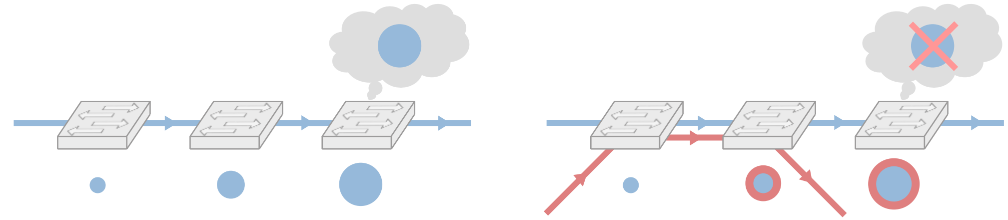
Section 3.1

TSN TRANSMISSION SELECTION – BOUNDED LATENCY MECHANISMS

-- STRICT PRIORITY

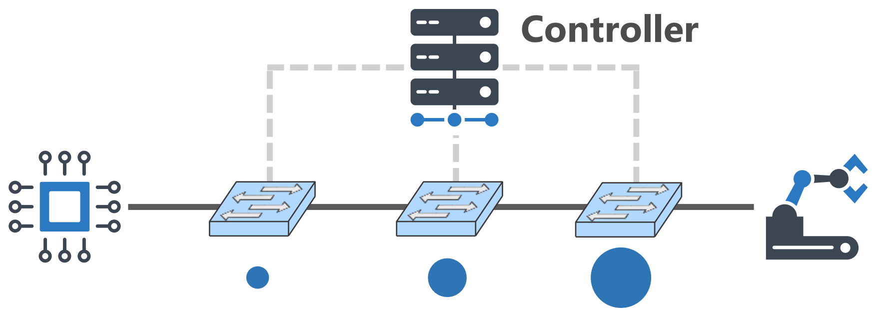
Strict Priority Latency

Snowball Effect without Shaping



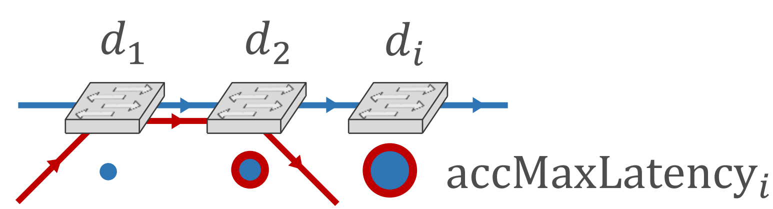
Consider Snowball Effect (increasing Bursts) during Reservation

- How does the control plane deal with changing network state?
- Assume that we do have **full knowledge** of all streams in the network



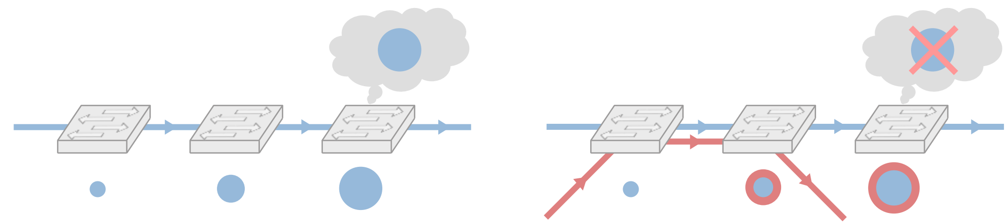
- Delay budget in switch i depends on **accumulated max. latency** in switches $1 - i$:

$$\mathrm{accMaxLatency}_i = d_1 + d_2 + \cdots + d_i$$



Strict Priority Latency

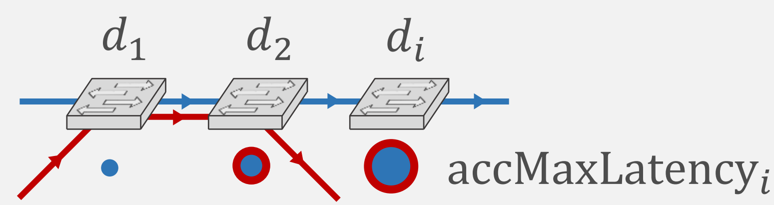
With Distributed Plug-and-Play



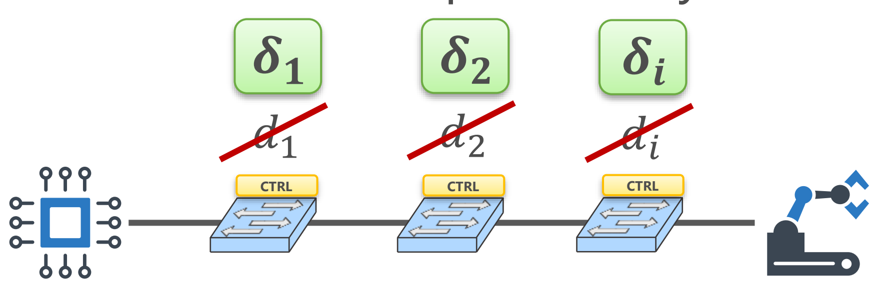
Consider Snowball Effect (increasing Bursts) during Reservation

- Delay budget (and effective burst size) depend on acc. max latency
- Assume that we do have **full knowledge** of all streams in the network

$$\text{accMaxLatency}_i = d_1 + d_2 + \cdots + d_i$$



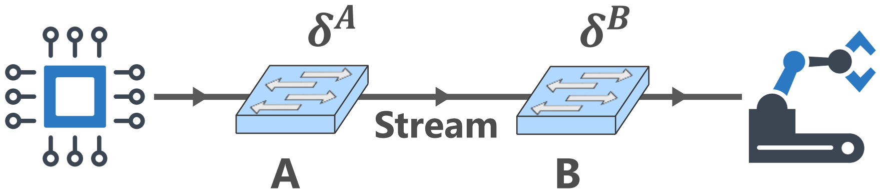
- Distributed control plane & dynamically changing demands



- Do not use **traffic-dependent** delay d_i, but **constant** threshold $\boldsymbol{\delta}_i$

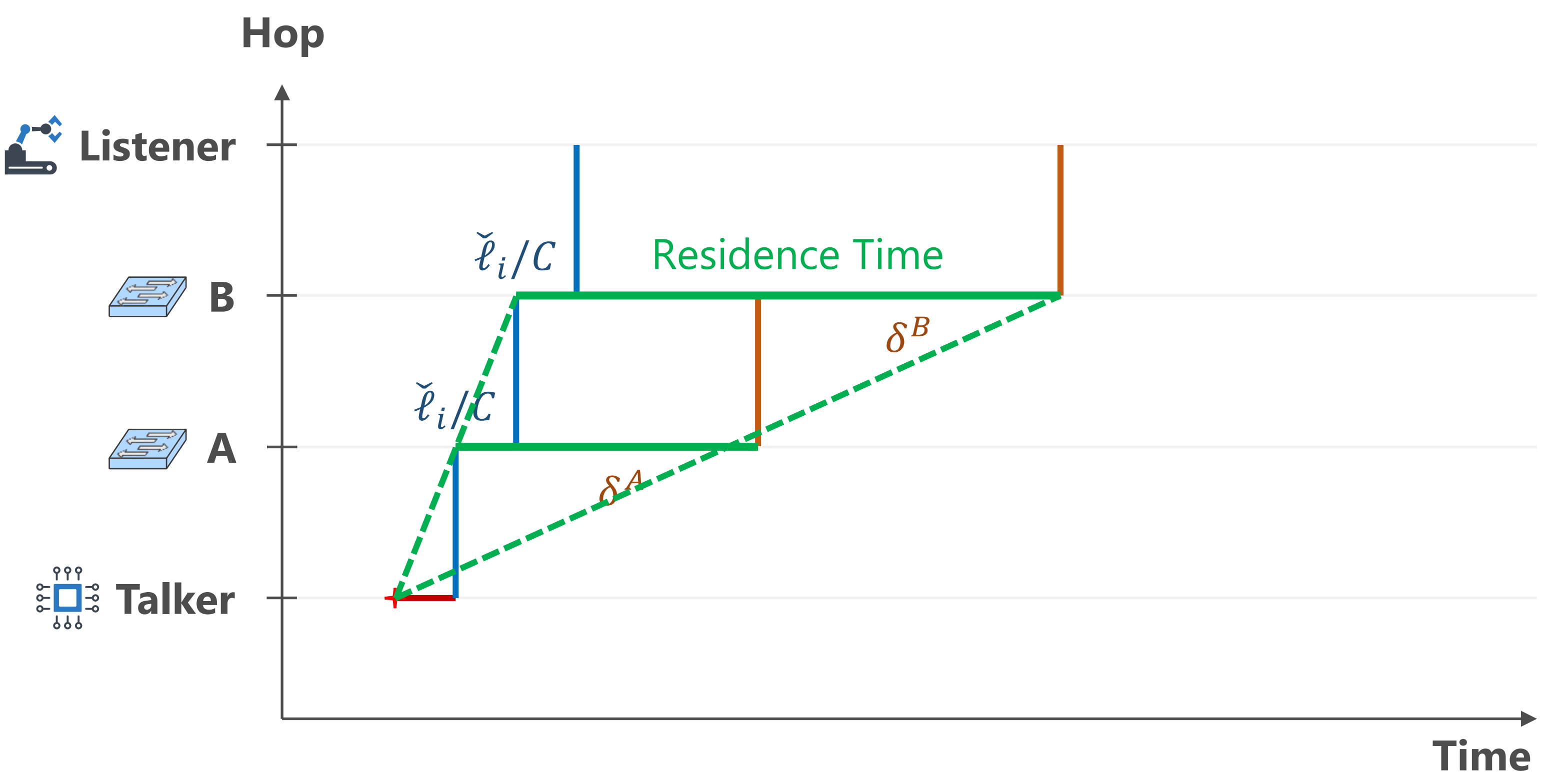
$$\text{accMaxLatency}_i = \boldsymbol{\delta}_1 + \boldsymbol{\delta}_2 + \cdots + \boldsymbol{\delta}_i$$

Frame Residence Times

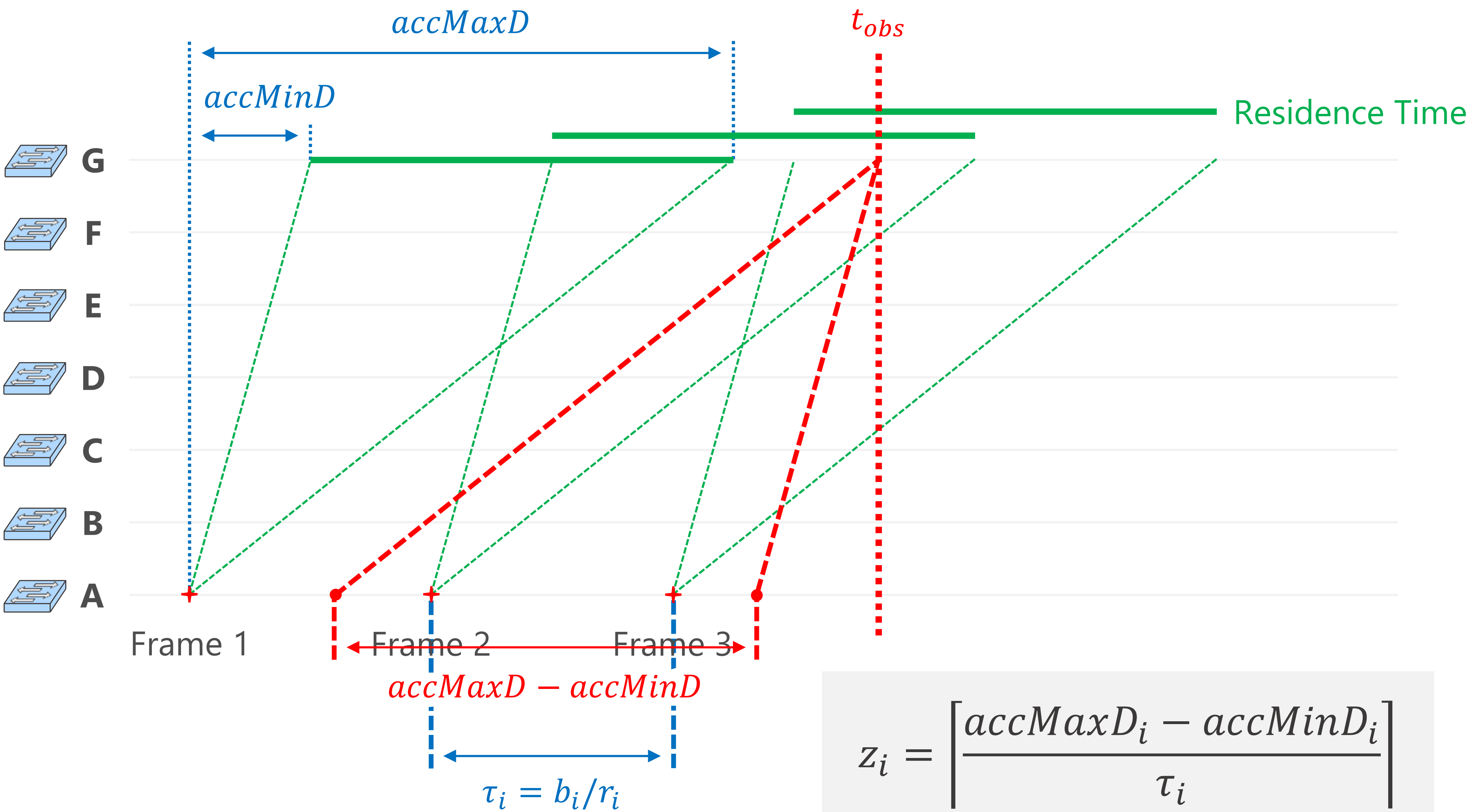


Snowball Effect

- Accumulated Max. Latency $accMaxD = b_i/C \quad + \quad \delta^A \quad + \quad \delta^B$
- Accumulated Min. Latency $accMinD = \check{\ell}_i/C \quad + \quad \check{\ell}_i/C \quad + \quad \check{\ell}_i/C$



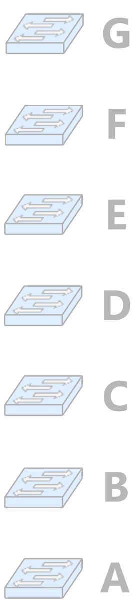
How many Frames at the Same Time?



$$z_i = \left\lceil \frac{accMaxD_i - accMinD_i}{\tau_i} \right\rceil$$

A. Grigorjew, F. Metzger, T. Hoßfeld, J. Specht, F.-J. Götz, F. Chen, and J. Schmitt, “Bounded latency with bridge-local stream reservation and strict priority queuing,” in 2020 11th International Conference on Networks of the Future (NoF). IEEE, 2020.

How many Frames at the Same Time?



We learned (1):

Shaping is not required for latency bounds.
We just need some information about
the data volume and the order of transmission.

(But shaping does improve the efficiency.)

We learned (4):

Assume: Pre-defined latency thresholds.
Then we can derive **worst-case** latency bounds
for **dynamic** networks with **distributed** admission control.

(But full network-wide overview
may allow more optimization.)

Section 3.2

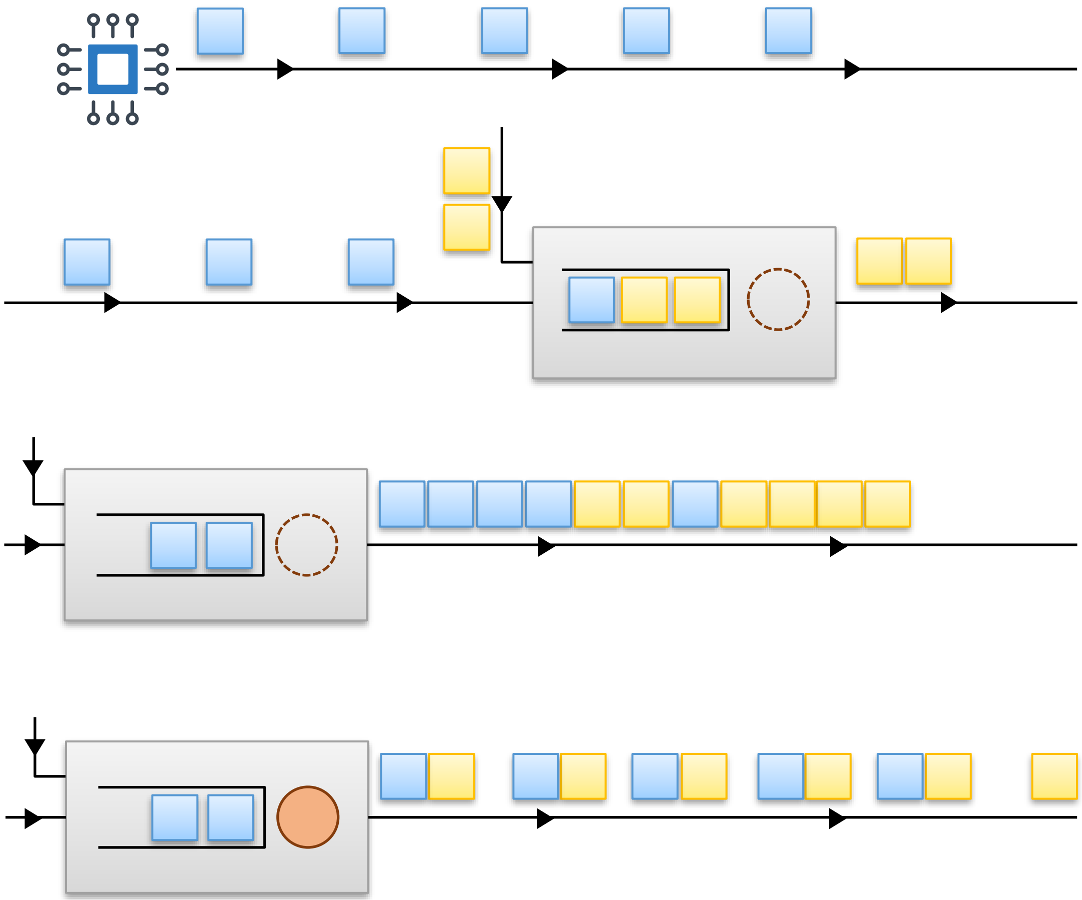
TSN TRANSMISSION SELECTION – BOUNDED LATENCY MECHANISMS

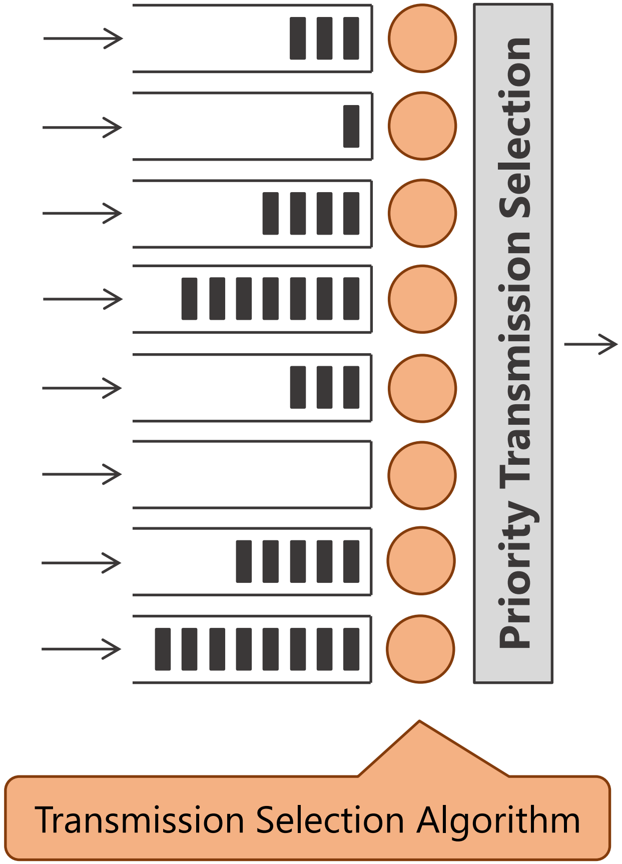
-- PER-HOP RESHAPING (CBSA, ATS)

Per-Hop Reshaping

General idea:

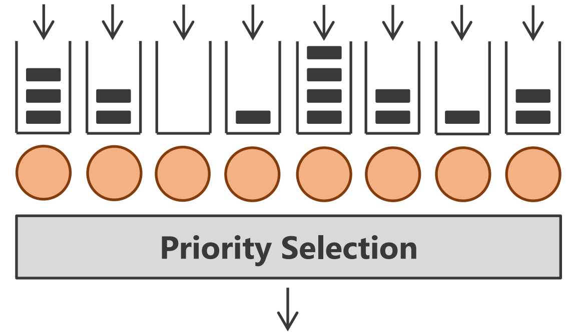
- Prevent accumulating bursts
- Slow down frames that are too fast





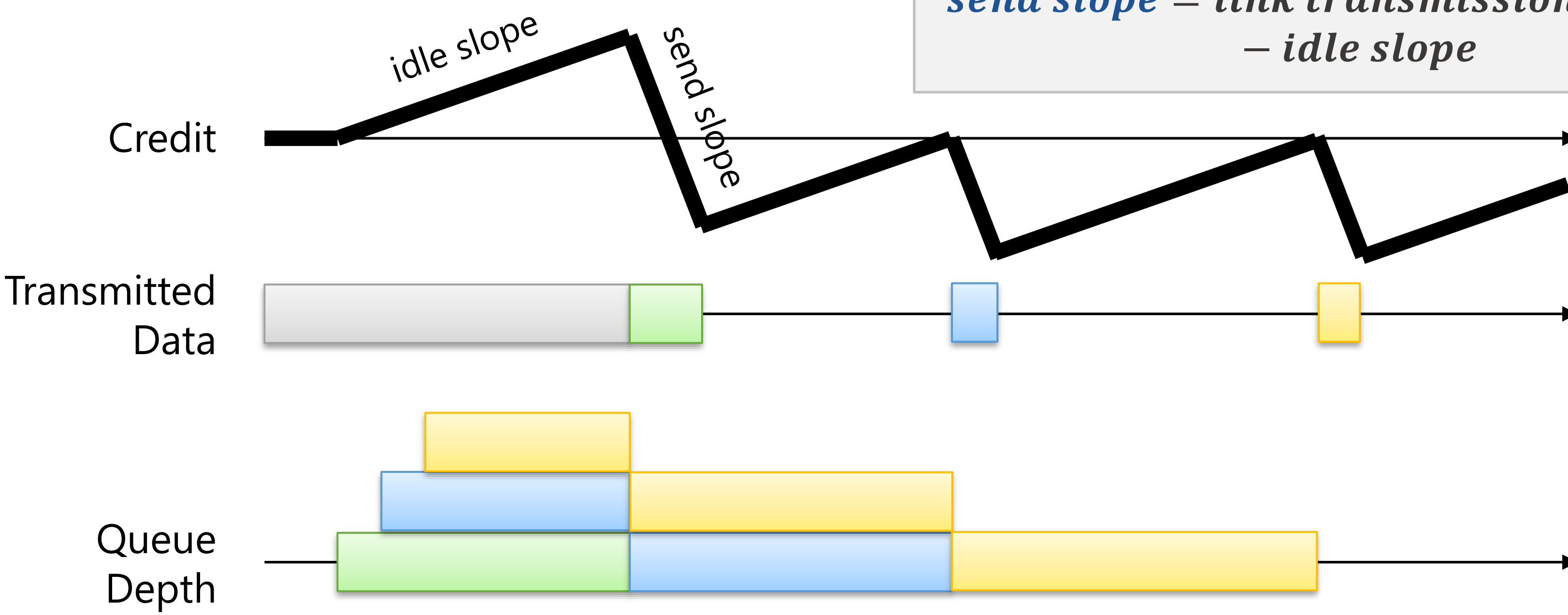
Credit-Based Shaping Algorithm (CBSA)

- One shaper for each priority
- Shared credit between all streams
- Can only send when $credit \geq 0$



$$\mathbf{idle\ slope} = \sum \mathbf{stream\ data\ rates}$$

$$\mathbf{send\ slope} = \mathbf{link\ transmission\ speed} - \mathbf{idle\ slope}$$



Remember: Is it really that simple?

IEEE 802.1BA

AUDIO VIDEO BRIDGING (AVB) SYSTEMS — IEEE Std 802.1BA-2011

As discussed in L.3 of IEEE Std 802.1Q, there are a number of factors that increase the latency contribution of a Bridge. AVB Talkers and Bridges shall report each hop's worst-case latency via SRP. The following equation gives an example of how worst-case latency can be calculated for Class A streams:

$$\text{Max Latency} = t_{\text{Device}} + t_{\text{MaxPacketSize+IPG}} + (t_{\text{AllStreams}} - t_{\text{StreamPacket+IPG}}) \times \text{Rate/MaxAllocBand} + t_{\text{StreamPacket}}$$

where

t_{Device} = the internal delay of the device (in increments of 512 bit times)

NOTE 3—t_{Device} is an integral multiple of 512 bit times so that it scales with the speed of the media.

$t_{\text{MaxPacketSize+IPG}}$ = the transmission time for a maximum size interfering frame (1522 octets to 2000 octets) plus its preamble and start of frame delimiter (SFD) (8 octets), and the following inter-packet gap (IPG) (12 octets)

$t_{\text{StreamPacket}}$ = the transmission time for the maximum frame size of the stream that is being reserved, plus its preamble and SFD (8 octets)

$t_{\text{StreamPacket+IPG}}$ = the transmission time for the maximum frame size of the stream that is being reserved, plus its preamble and SFD (8 octets) and the following IPG (12 octets)

Rate = the transmission rate of the port

MaxAllocBand = maximum allocatable bandwidth, the maximum amount of bandwidth the AVB system is able to allocate for Class A streams on the port

t_{Interval} = the Class A observation interval or 125 µs

$t_{\text{AllStreams}}$ = (MaxAllocBand × t_{Interval}) / tRate = the sum of the transmission times of all Class A stream frames the AVB System is able to allocate in an observation interval (125 µs) on a port

NOTE 4—Talkers that are not aware of their maximum allocatable bandwidth and Bridges should use maximum values for their maximum allocatable bandwidth (75% of the port transmission rate) and for their $t_{\text{AllStream}}$ (93.75 µs in the case of Class A).

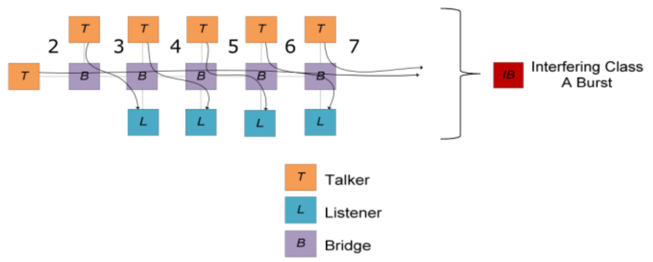
NOTE 5—The logic behind the above formulation for Max Latency can be found in the paper "AVB Latency Math" [B1]. Further discussion of latency issues can be found in the paper "Class A Latency Issues" [B2]. Only SR class A example calculations are shown; SR class B examples can be similarly derived.

Class A Bridge Latency Calculations

Hochschule Deggendorf

Bridge Latency @FE – Case 4(1)

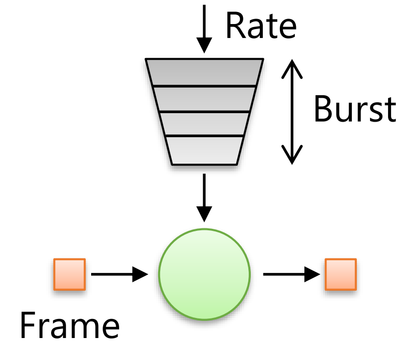
With a limit of 7 hops for the interfering class A burst (is there a limit?) the following topology should be the worst case for the interfering burst:

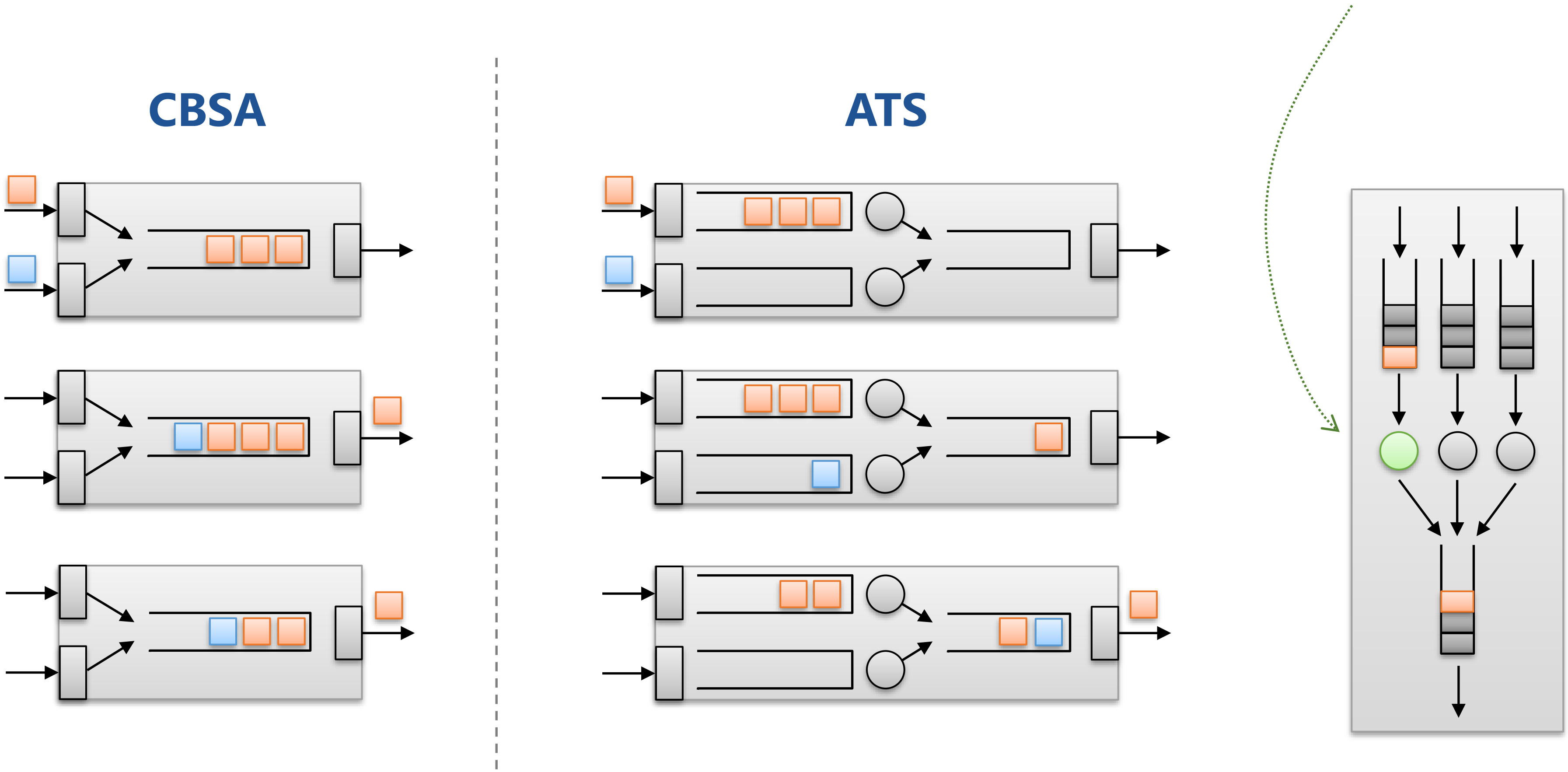


09.11.2010 IEEE 802.1 AVB – November 2010 Dallas, TX 22

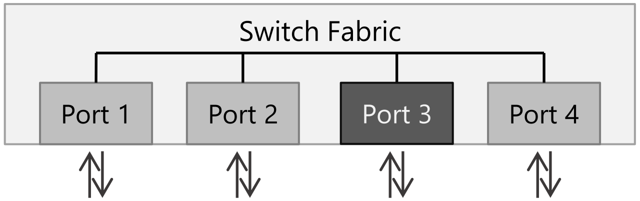
Asynchronous Traffic Shaping (ATS)

- Fine-grained token bucket shaper
- **Each stream** has its own token bucket state
- Can only send when token bucket has sufficient tokens





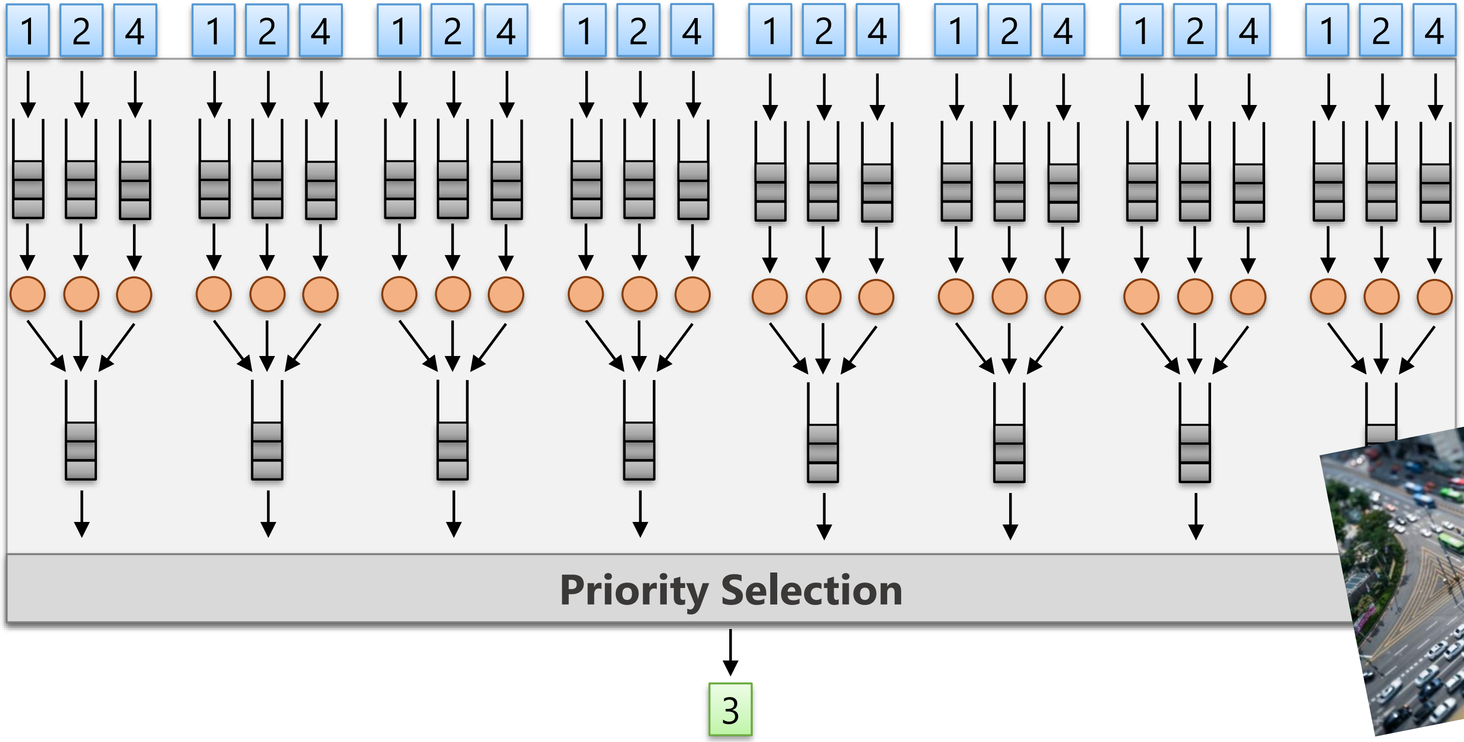
Is ATS Basically RSVP IntServ 2.0?



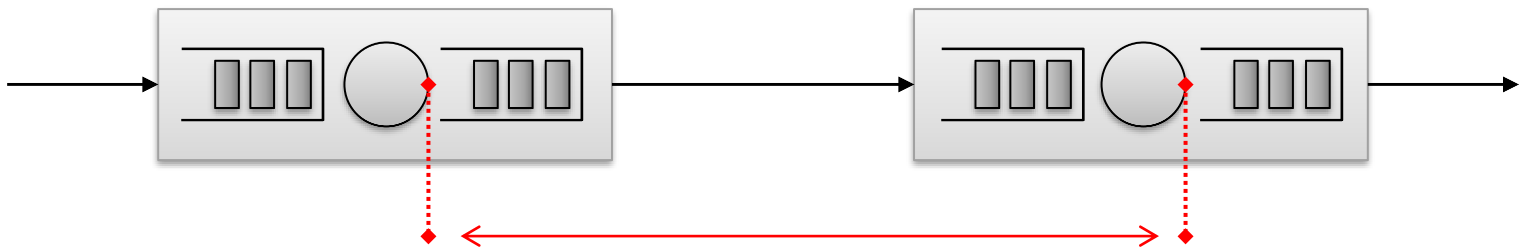
Credit-Based Shaper (CBS)

- One shaper for each priority
- Coarse-grained: Head of line blocking

Asynchronous Traffic Shaper (ATS)



ATS Latency Bound with Interleaved Queuing



Per-hop latency bound

- From shaper to shaper (not ingress to ingress)
- Delay bound:

Sum of all bursts of higher or equal traffic classes

Largest frame from Lower traffic classes

$$d_i^{PQ,SO} \leq \max_{j \in I} \left(\frac{\hat{b}_H + \hat{b}_{C(j)} + \hat{b}_j - \check{l}_j + \hat{l}_L}{r - \hat{r}_H} + \frac{\check{l}_j}{r} \right)$$

Remaining link speed, after subtracting higher priority rates

Transmission time of the smallest frame from this stream

J. Specht and S. Samii, “Urgency-based scheduler for time-sensitive switched ethernet networks”, in 2016 28th Euromicro Conference on Real-Time Systems (ECRTS), Jul. 2016, pp. 75–85.

ATS Latency Bound with Interleaved Queuing

Per-hop l

- From s
- Delay b

d_i

We learned (5):

ATS successfully prevents accumulating bursts.

(It does not need per-stream queuing,
but only per traffic class and per ingress port.)

Remaining link speed, after subtracting higher priority rates

Transmission time of the smallest frame from this stream

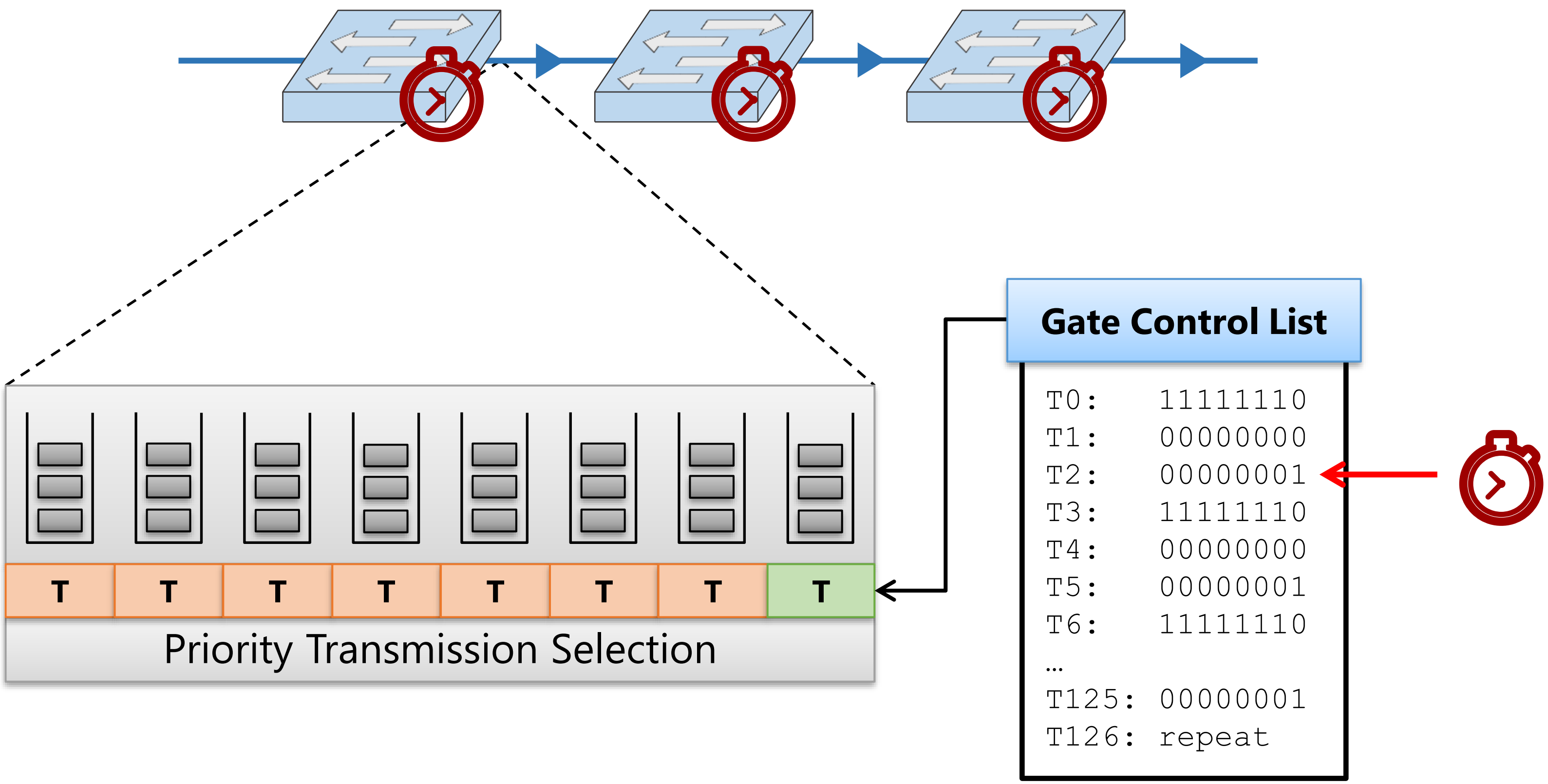
J. Specht and S. Samii, “Urgency-based scheduler for time-sensitive switched ethernet networks”, in 2016 28th Euromicro Conference on Real-Time Systems (ECRTS), Jul. 2016, pp. 75–85.

Section 3.3

TSN TRANSMISSION SELECTION – BOUNDED LATENCY MECHANISMS

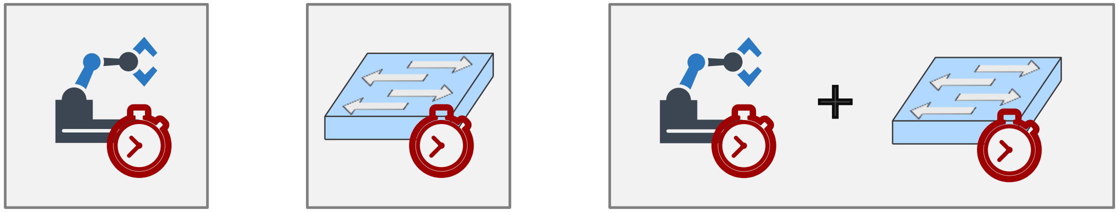
-- SYNCHRONIZED SHAPERS (TAS, CQF)

Time-Aware Shaper (TAS)

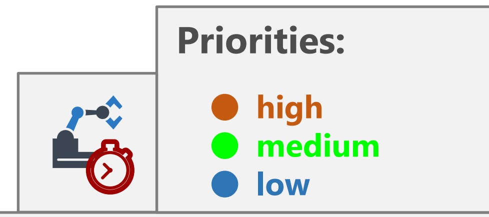


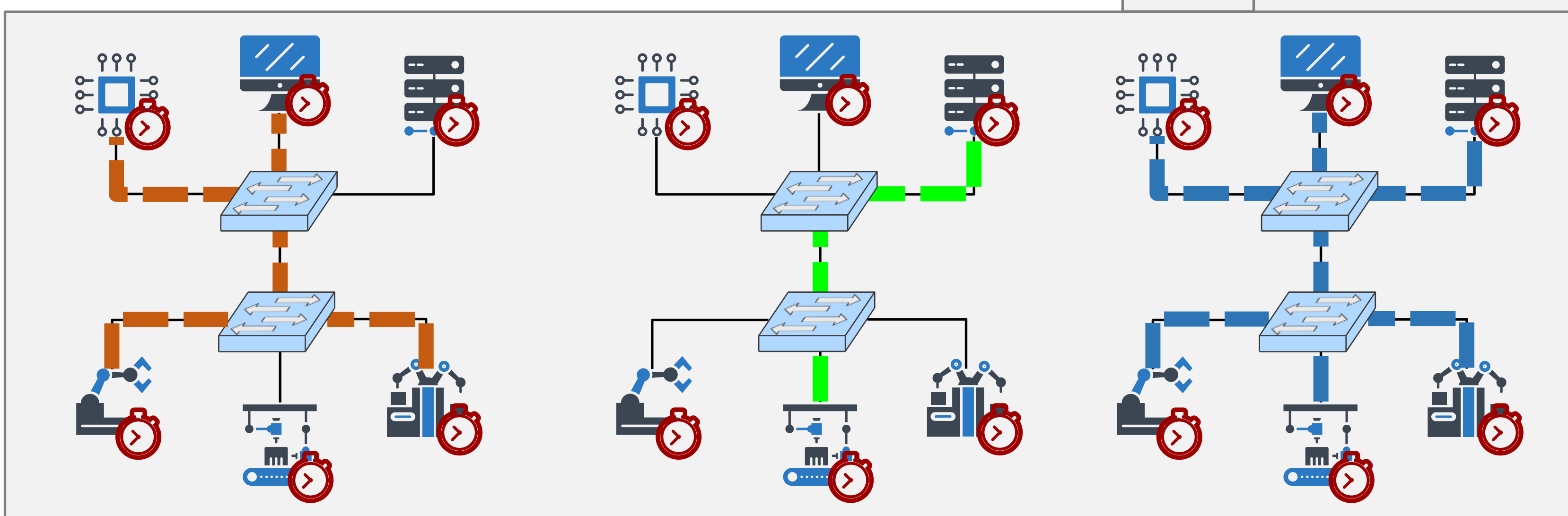
Levels of Isolation – Class Separation

Three setups possible:



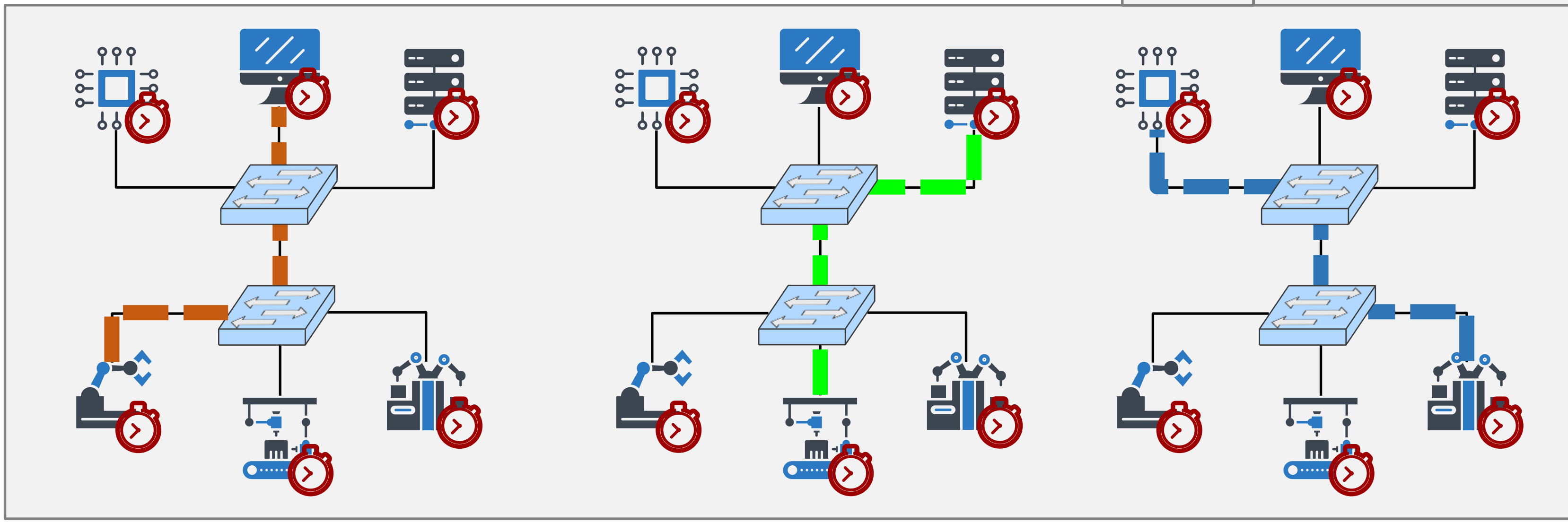
- Different isolation strategies possible
- **Strategy (1):** isolate traffic classes
- Streams still interfere with each other, but less





Levels of Isolation – Stream Separation

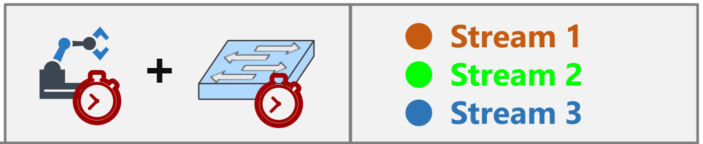
Strategy (2): isolate streams end-to-end

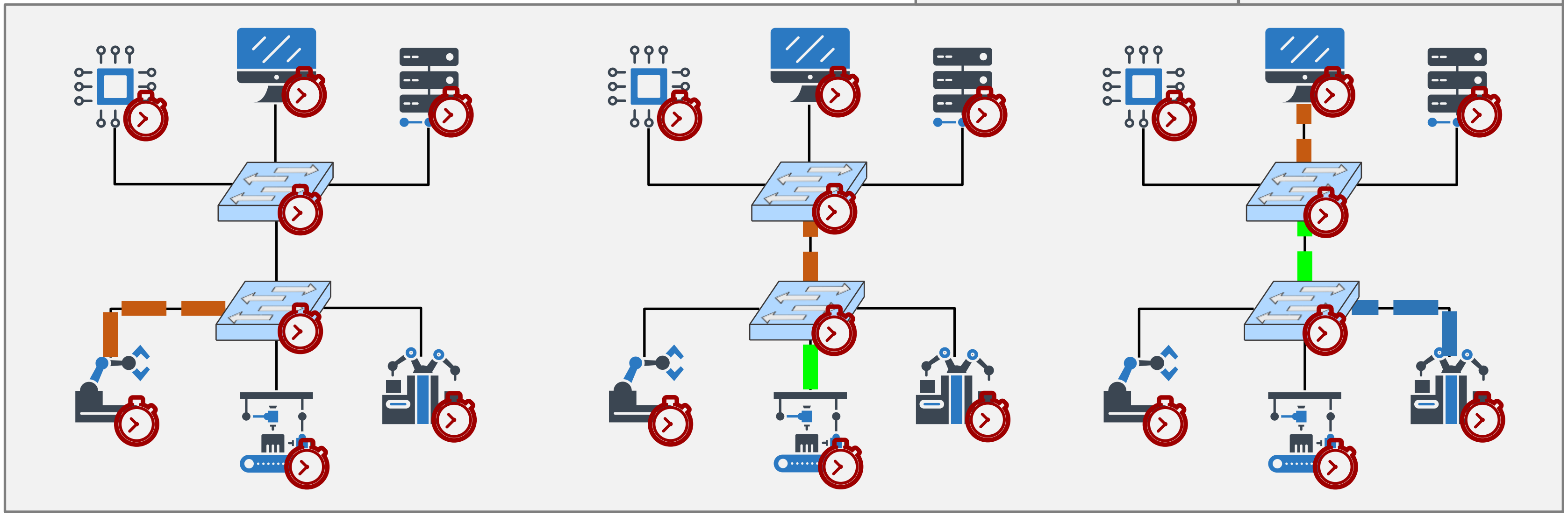


- Basically Time-Division Multiple Access (TDMA) for the entire network
- Similar to many PROFINET deployments
- Line depth makes planning more difficult
- Lot’s of wasted bandwidth / **links are often idle**

Levels of Isolation – Per-Hop Stream Separation

Strategy (3): isolate streams on each hop

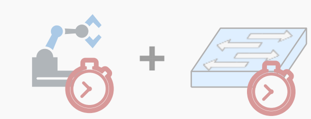




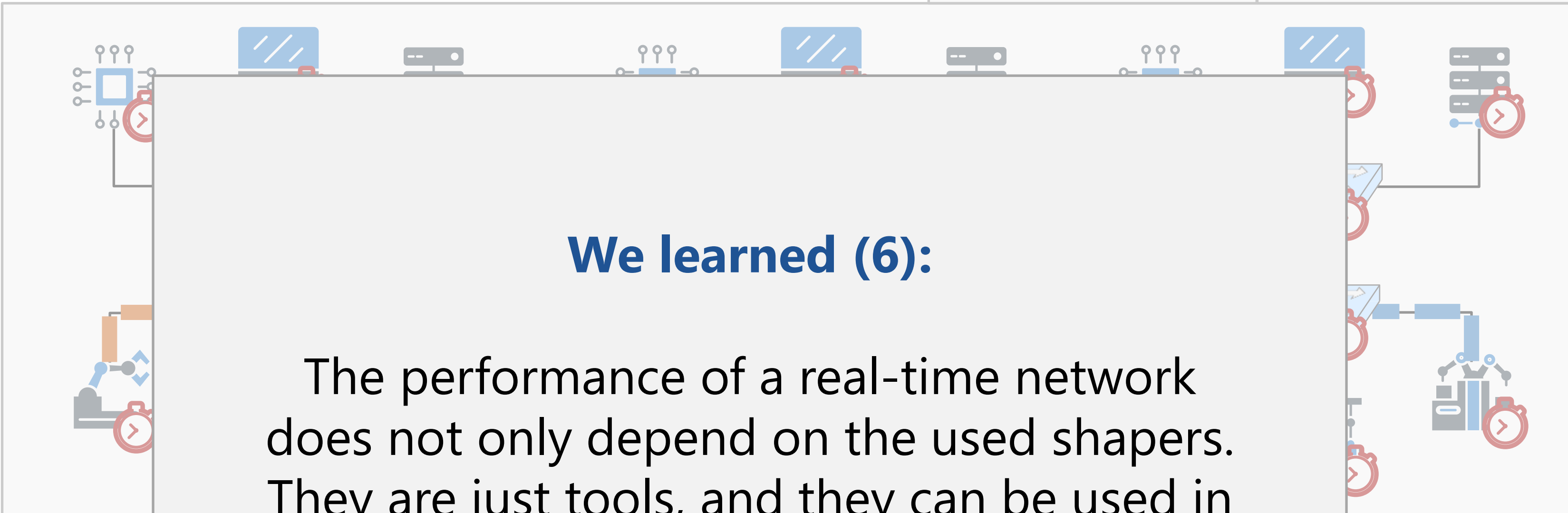
- Shorter cycles and ultra low latencies
- Bad efficiency: **links are still often idle**
- May require a **LOT** of engineering
  - Scheduling could change completely for every new stream
  - No simple plug & play

Levels of Isolation – Per-Hop Stream Separation

Strategy (3): isolate streams on each hop



Stream 1
Stream 2
Stream 3



We learned (6):

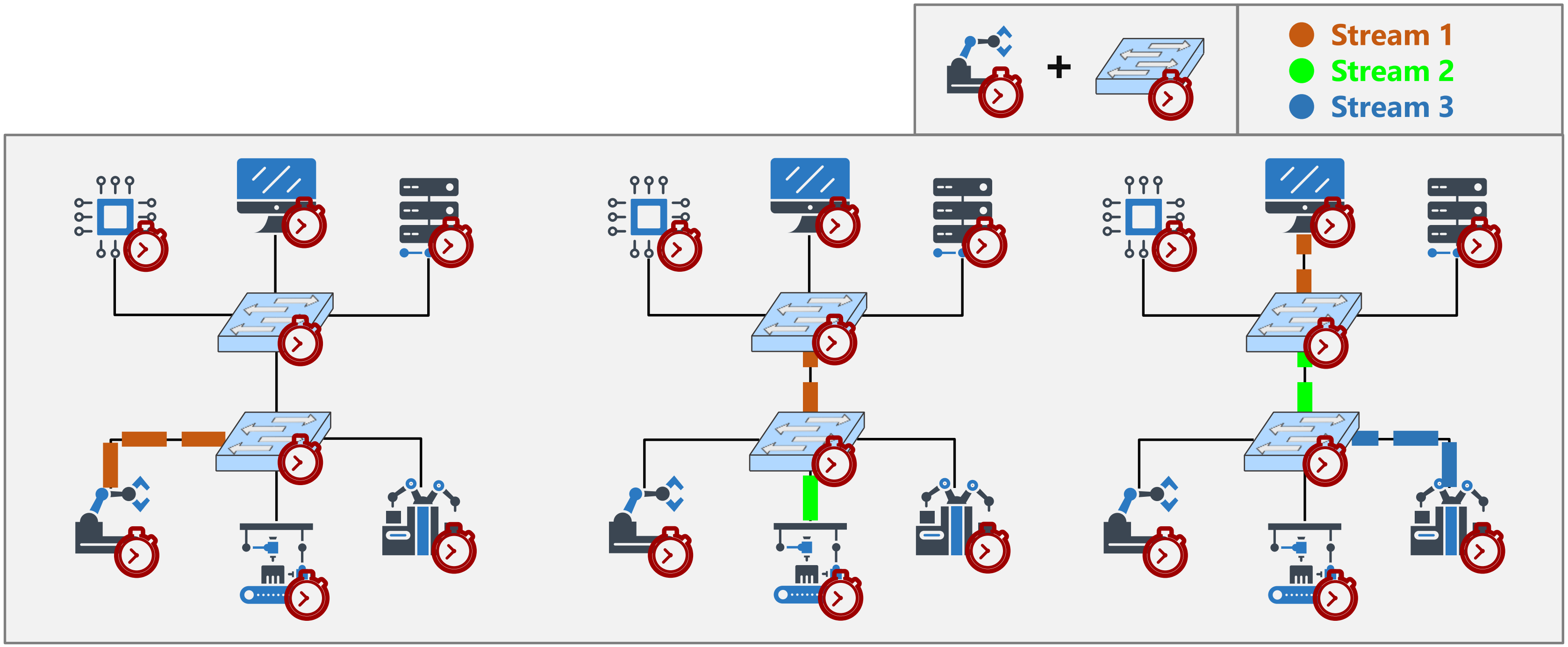
The performance of a real-time network does not only depend on the used shapers. They are just tools, and they can be used in different ways, with different results.

- Shorte
- Bad eff
- May require a **LOT** of engineering
  - Scheduling could change completely for every new stream
  - No simple plug & play

Another Plane of Isolation: Cycle Separation

Wanted:

- Predictable latency with low jitter
- No expensive scheduling
- More dynamic setup
- Better bandwidth efficiency

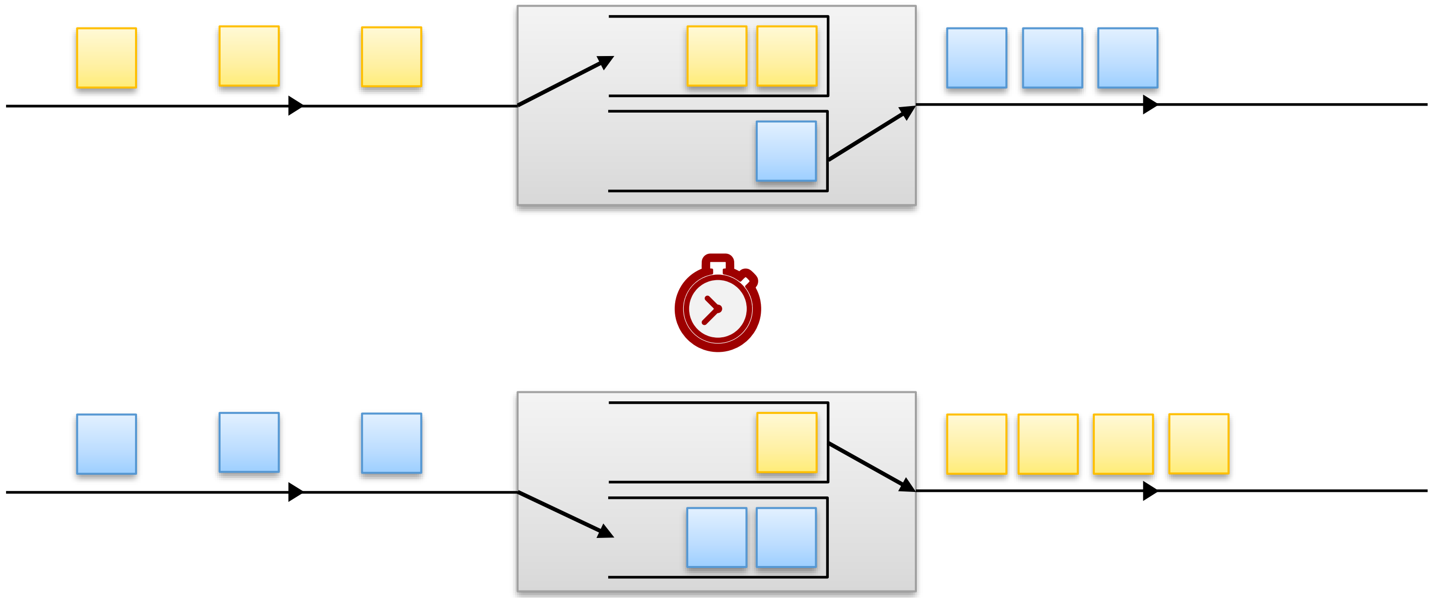


Cyclic Queuing and Forwarding (CQF)

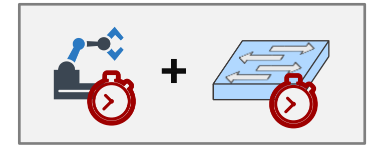
Wanted:

- Predictable latency with low jitter
- No expensive scheduling
- More dynamic setup
- Better bandwidth efficiency

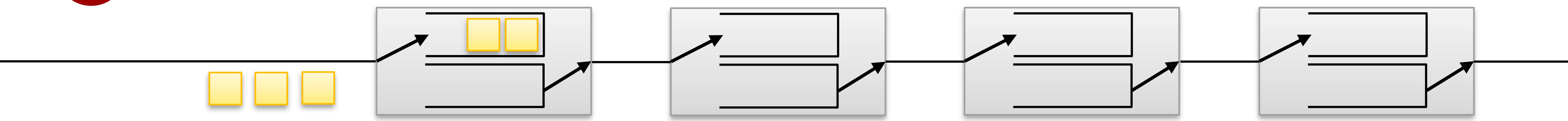
Solution: More Queues! (again)



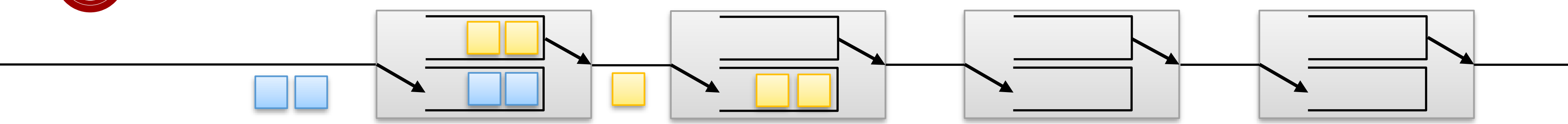
CQF Multi-Hop Example



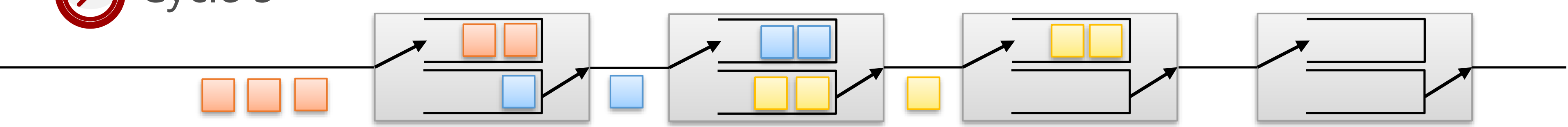
Cycle 1



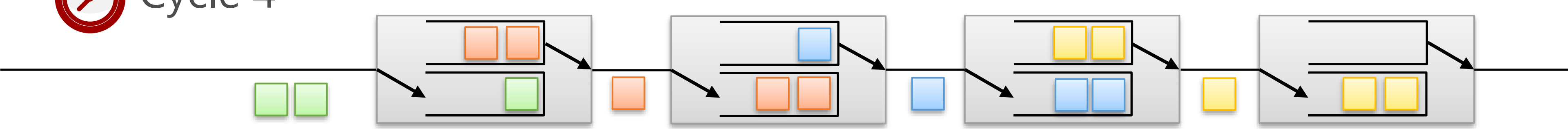
Cycle 2



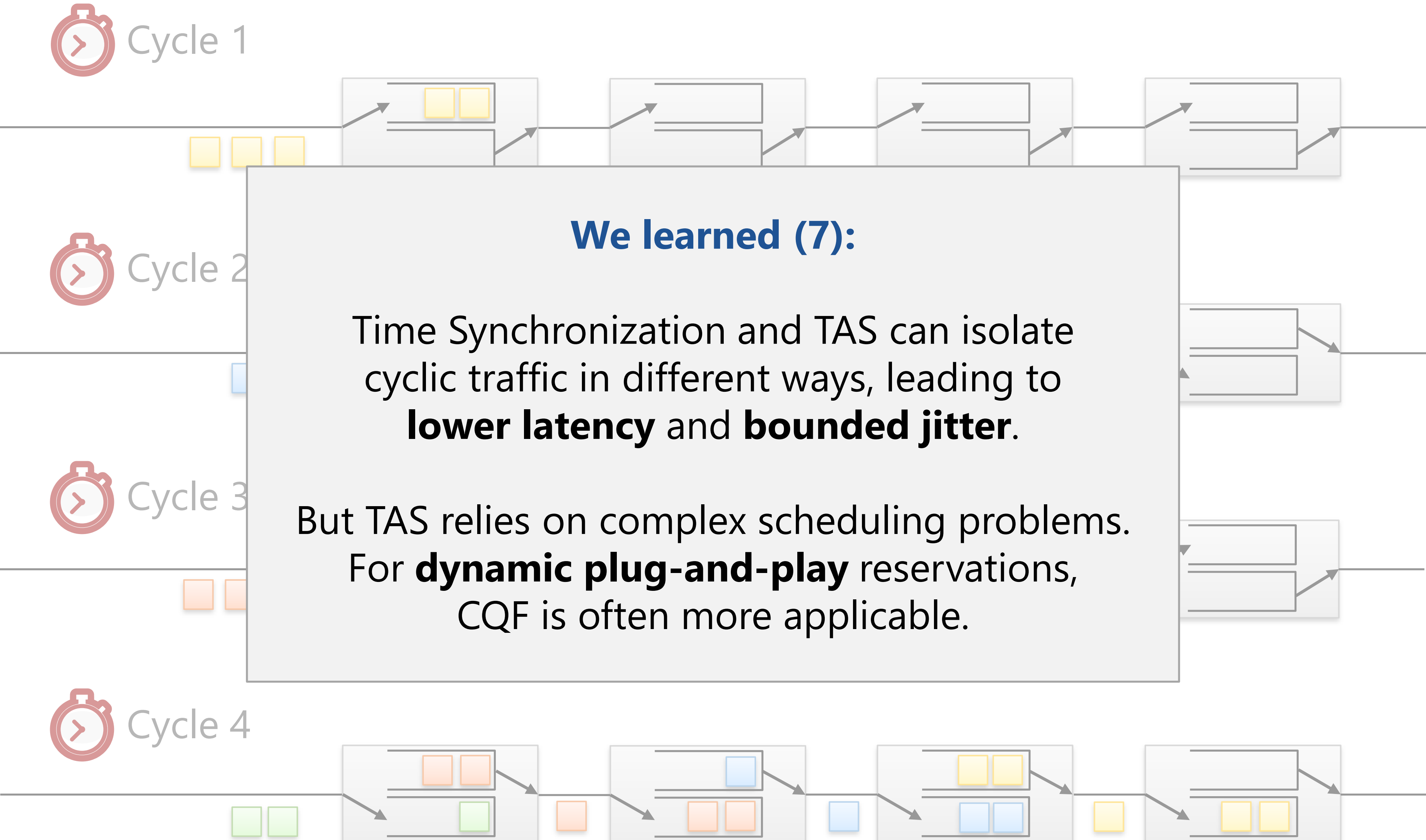
Cycle 3



Cycle 4



CQF Multi-Hop Example



We learned (7):

Time Synchronization and TAS can isolate cyclic traffic in different ways, leading to **lower latency** and **bounded jitter**.

But TAS relies on complex scheduling problems. For **dynamic plug-and-play** reservations, CQF is often more applicable.

Section 4

PRACTICAL EXAMPLE – LATENCY COMPUTATION IN PYTHON

https://github.com/lsinfo3/netsys21-tsn-tutorial

Section 5

BRIEF COMPARISON AND CONCLUSION

How Would You Compare The Tools?

Requirements

- Is time synchronization available?
- **Estimated** hardware complexity (e.g., number of queues)
- Cyclic packets vs. asynchronous, event-based communication
- Static networks vs. dynamic operation / plug & play

Measurable KPIs

- Delay bounds, jitter bounds
- **Number of supported streams** with pre-configured delay thresholds
- **Bandwidth efficiency:** usable bandwidth vs. link idle time
- Stream reservation performance (algorithm runtime)
- Network planning and provisioning complexity / runtime
  - TAS scheduling computation
  - Optimized delay threshold configuration

Brief Comparison: What does my Topology need?

- Based on opinion!
- Greatly simplified decisions!
- There is no general solution to your needs. Each mechanism comes with trade-offs!
- **Remember:** Not only the shaper is important, but also its application (isolation strategy / available information / delay equations)

- If **Strict Priority + Dynamic Reservation** is sufficient for your needs: go for it! (cheapest)
- Large topology / many hops?
  - **CBS:** isolates priorities / huge optimization potential
  - **ATS:** prevents accumulating bursts (snowball effect)
- Cyclic traffic? Very low latency or low jitter required?
  - **CQF:** simple configuration, dynamic application, low jitter
  - **TAS:** full traffic isolation in static networks

Summary

Determinism for Ethernet

- In particular: bounded latency
- TSN specified multiple mechanisms with different strategies

Challenges

- Accumulating bursts due to latency variance
- Limited traffic information available for latency computation in dynamic plug-and-play setups

Guaranteed latency can be simple

- Latency computation is based on information – not on shapers
- But shaping improves the latency guarantee
- Dynamic implementation with pre-configured thresholds

Trade-offs

- Different shapers improve different aspects
- There is no general solution for all needs
- Compare multiple KPIs for your specific scenario

Some Open Research Questions

- Network planning and provisioning
  - Configuration of thresholds for dynamic networks
  - Scheduling for static networks
  - Dynamic scheduling with central network management
- Comparison of KPIs with different shapers and isolation strategies
- Transfer of TSN concepts onto layer 3 Deterministic Networking
  - DetNet cooperates with TSN, and brings new challenges in layer 3
  - Heterogeneous deployments, i.e., switches use different shapers
  - Large number of streams becomes overwhelming → aggregation
- Ethernet is not just an improved bus and will change use cases
  - How do your future networks look like?
  - What latency do you need?
  - What are your other requirements? (dynamics, synchronization, ...)
- Explore hardware limitations and performance
  - How would central management units predict switch limitations?
- Prototyping and flexible, programmable data plane switches
  - What can already be deployed today? What functions do we need?

Thank You!

Icons: Rudez Studio (https://www.iconfinder.com/Ruslancorel), Shawn Rubel (https://www.iconfinder.com/Vecteezy), own creations

Literature

(1) TSN Task Group website: https://1.ieee802.org/tsn/
(2) J. Farkas, "IEEE 802.1 Time-Sensitive Networking (TSN) Task Group (TG) Overview", DetNet – TSN workshop, Nov. 2018 (and his other overview presentations on the TSN TG website)
(3) H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing", in IEEE INFOCOM'93 The Conference on Computer Communications, Proceedings, IEEE, 1993, pp. 227–236.
(4) J. Specht and S. Samii, "Urgency-based scheduler for time-sensitive switched ethernet networks", in 2016 28th Euromicro Conference on Real-Time Systems (ECRTS), Jul. 2016, pp. 75–85.
(5) A. Grigorjew, F. Metzger, T. Hoßfeld, J. Specht, F.-J. Götz, F. Chen, and J. Schmitt, "Bounded latency with bridge-local stream reservation and strict priority queuing," in 2020 11th International Conference on Networks of the Future (NoF). IEEE, 2020.
(6) A. Grigorjew, C. Baier, F. Metzger and T. Hoßfeld, "Distributed Implementation of Deterministic Networking in Existing Non-TSN Ethernet Switches," 2021 IEEE International Conference on Communications Workshops (ICC Workshops), 2021, pp. 1-6
(7) A. Grigorjew, N. Gray and T. Hoßfeld, "Dynamic Real-Time Stream Reservation with TAS and Shared Time Windows," 2021 IFIP Networking Conference (IFIP Networking), 2021, pp. 1-6
(8) A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, "Ultra-low latency (ull) networks: The ieee tsn and ietf detnet standards and related 5g ull research", IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp. 88–145, 2018.
(9) A. Nasrallah, V. Balasubramanian, A. Thyagaturu, M. Reisslein, and H. ElBakoury, "TSN algorithms for large scale networks: A survey and conceptual comparison", arXiv preprint arXiv:1905.08478, 2019.
(10) S. S. Craciunas, R. S. Oliver, M. Chmelik, and W. Steiner, "Scheduling real-time communication in ieee 802.1 qbv time sensitive networks", in Proceedings of the 24th International Conference on Real-Time Networks and Systems, 2016, pp. 183–192.
(11) E. Schweissguth, P. Danielis, D. Timmermann, H. Parzyjegla, and G. Mühl, "Ilp-based joint routing and scheduling for time-triggered networks", in Proceedings of the 25th International Conference on Real-Time Networks and Systems, 2017, pp. 8–17.
(12) S. Bondorf, P. Nikolaus, and J. B. Schmitt, "Quality and cost of deterministic network calculus: Design and evaluation of an accurate and fast analysis", Proceedings of the ACM on Measurement and Analysis of Computing Systems, vol. 1, no. 1, p. 16, 2017.
(13) K. Jang, J. Sherry, H. Ballani, and T. Moncaster, "Silo: Predictable message latency in the cloud," ACM SIGCOMM Computer Communication Review, vol. 45, no. 4, pp. 435–448, 2015.
(14) J. Y. Le Boudec, and P. Thiran, "Network calculus: a theory of deterministic queuing systems for the internet", Springer Science & Business Media, 2001