RPR Traffic Management

Frederic Thepot fthepot@dynarc.com

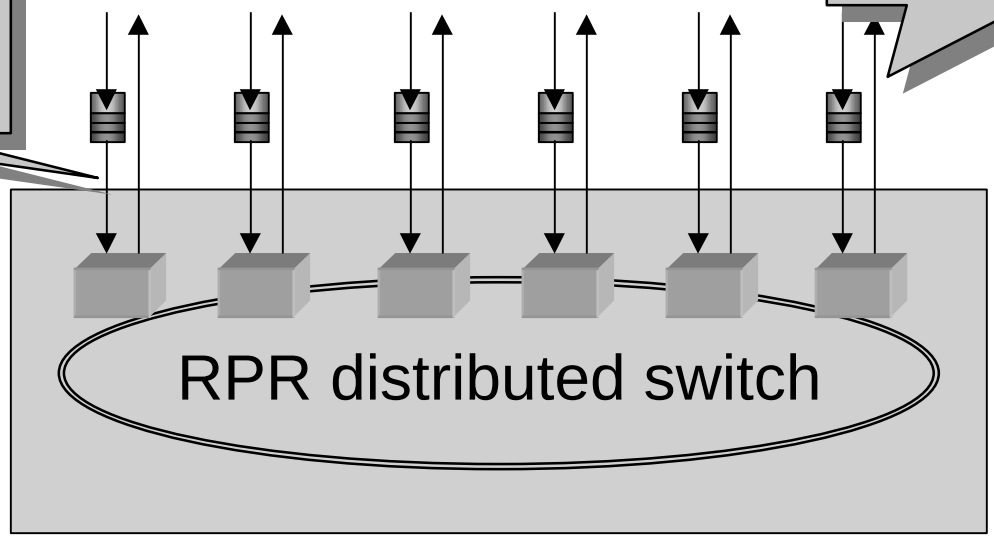
Lars Ramfelt larsh@dynarc.com

DYNARC

Introduction of RPR model

Ingress side:
- Ingress scheduling
- Customer traffic separation
- Congestion buffers (100's of MB)

Egress side:
- Flow control / back pressure



Distributed switch

- *The main issue is "control message **latency"** on the ring*
- *Performance model **should be the same** as for centralized switch!*
- *Base traffic case:*
  - *Equally distributed source and destination addresses*
  - *Hub environment, single and dual hub cases*
- *Diffserv model **should be the same** as for a centralized switch*

DYNARC

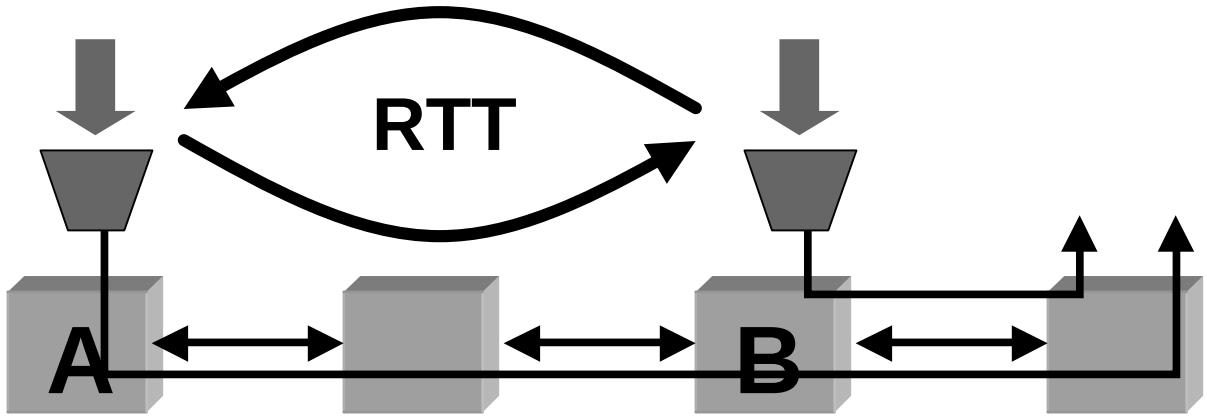
Goals of RPR resource management

- Fairness – if resources are scarce everybody will get the same service
- Non-fairness – all are equal but some are more equal (hub environment)
- Predictability (jitter,delay,…)
- Efficiency – high link utilization
- Scalability – increasing capacity and RTT
- QoS – Some traffic needs more deterministic service than others and are usually willing to spend more

Large number of requirements with complex trade-offs

DYNARC

Example: Fairness vs. high link utilization in RPR



1. A is utilizing the full link
2. Traffic arrives at B
3. Control information exchange between A and B
4. A is throttled one RTT later

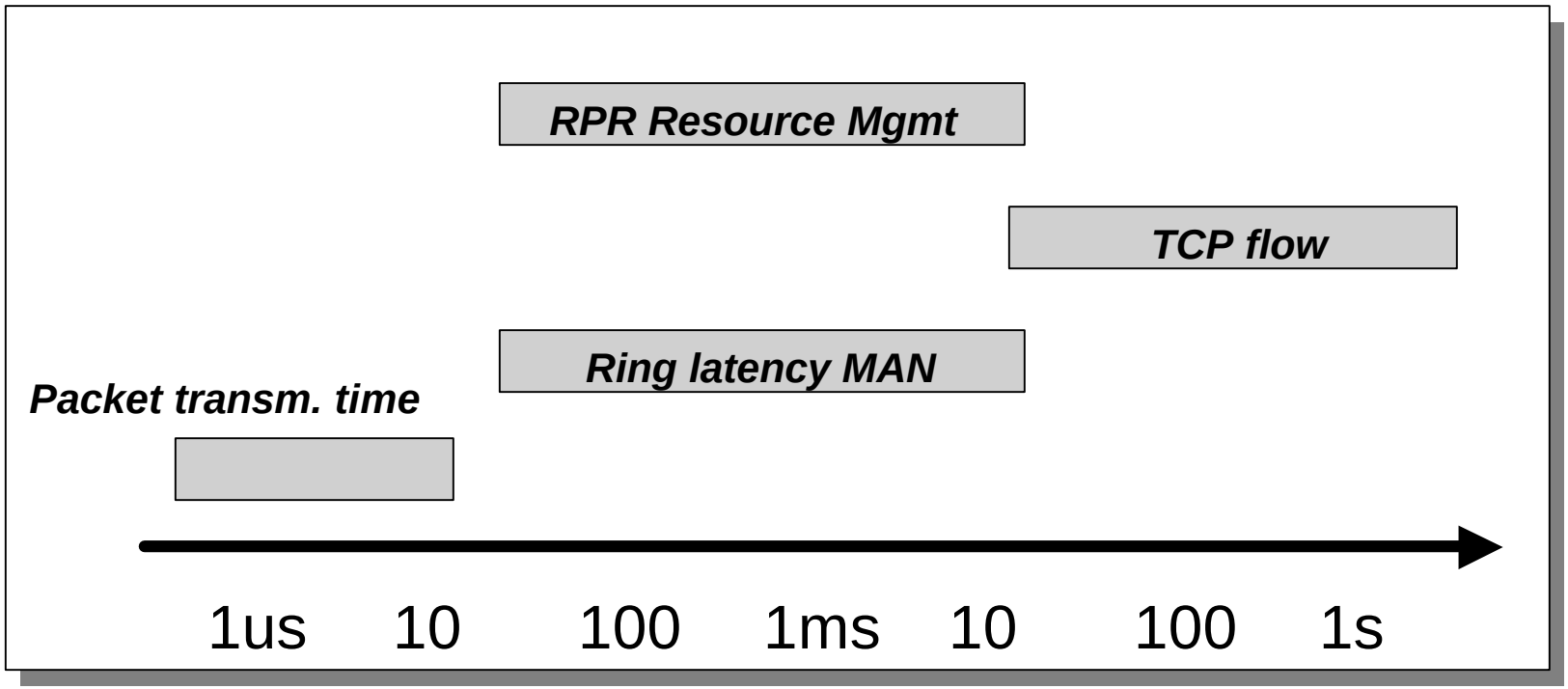
Interactions between nodes will be limited by RTT

DYNARC

The BW x Delay challenge

- The main challenge is the BW x Delay product in a distributed system and it will continue to be the main problem
  - This is a traditional problem from before, e.g…
    - CSMA/CD, DQDB, TCP, Token ring,…
- Networks with similar BW x Delay product will experience similar issues with latency
  - 200km@1Gbps = 20km@10Gbps = 2km@100Gbps

Network latency (RTT)



- *Limited value to operate resource management at a faster rate than RTT*
- *Packet transmission time decreases with capacity increase*

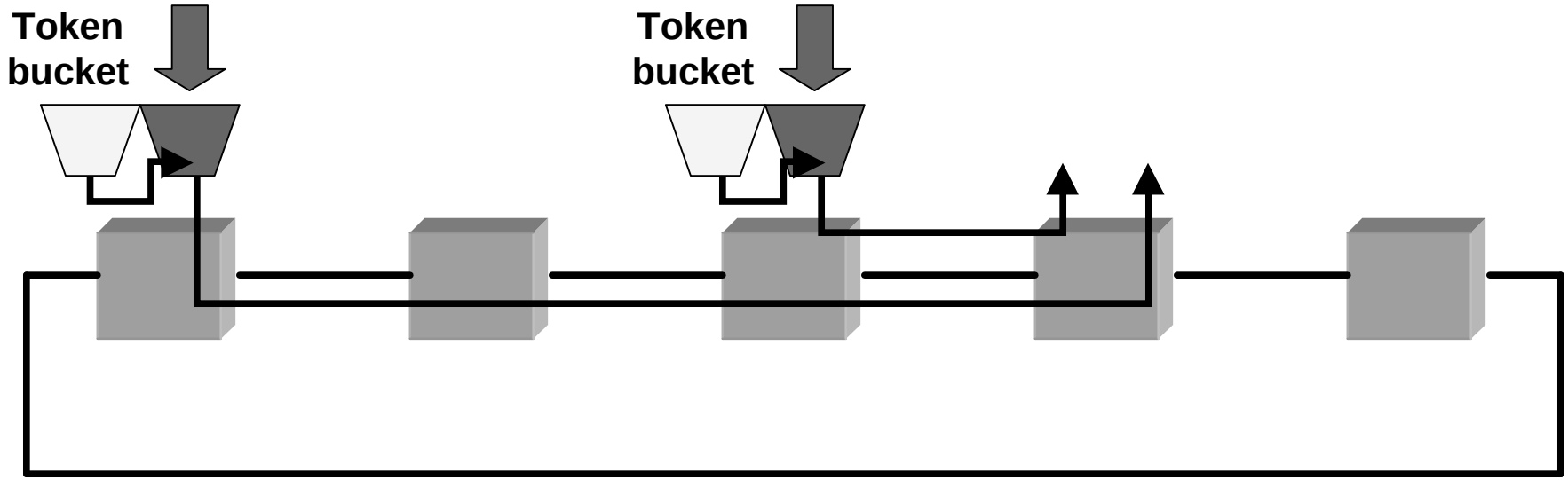
NO VALUE TO ADD COMPLEXITY IN HARDWARE FOR RESOURCE MANAGEMENT

DYNARC

How do we select a scalable solution?

If we want RPR to scale to an 100Gbps and 200km Metro RPR ring tomorrow it should be designed to work at 10Gbps in a 2000km today!

Example: DISTRIBUTED token bucket approach



- Scheduling is achieved by a token bucket
- Link tokens are re-allocated on demand
- Link tokens are used to configure token bucket
- Reallocation of full link capacity within one RTT

DYNARC

Conclusion

- Efficient use of resources and fulfilling of quality metrics (such as fairness,QoS..) is the goal of resource management
- There are many possible ways to implement such a scheme but they ALL suffer from scalability issues and operate on the timescale of RTT on the RPR ring
  - Fairness without efficient use of resources is simple (FIXED)
  - Efficient use of resources without fairness is also easy
- A software solution can be as efficient as a hardware implementation but is more flexible and leaves more room for future improvements and is external to the MAC

DYNARC

My Suggestion for RPR

- Implement a simple fairness scheme with a single token bucket that is independent of hardware and easy to extend later with more features.

- Easy to get interoperability in short time