Flow Aggregation in TSN

Tongtong Wang, Huawei
Norman Finn, Huawei
IEEE 802.1 TSN, November 2017, Orlando
Motivation

- After discussion of “stream aggregation for bridges” in Filling Detnet Needs in last meeting, this presentation provides more thoughts about TSN flow aggregation.
Motivation - Aggregation Scenario in Core Nodes

- Currently, core nodes need to keep all flow status in TSN network, and this will cause
  - Huge amount of flow table entries
  - More and more flow status refresh messages
- Flow aggregation can help simplify this problem, e.g. setup tunnels or dedicate paths etc., thus reduce queue number and scheduling cost.

- However, several issues come with aggregation
  - Multiple flows interfere each other in aggregation scenario.
  - Same latency for larger flows and smaller flows.
Motivation - Aggregation Scenario in Core Nodes

- Currently, core nodes need to keep all flow status in TSN network, and this will cause
  - Huge amount of flow table entries
  - More and more flow status refresh messages
- Flow aggregation can help simplify this problem, e.g. setup tunnels or dedicate paths etc., thus reduce queue number and scheduling cost.

- However, several issues come with aggregation
  - Multiple flows interfere each other in aggregation scenario.
  - Same latency for larger flows and smaller flows.
Aggregation Mode - 1

- Coarse-grained aggregation
  - Simply map a group of flows into one reservation (a set of queue and shaper)
  - Next bridge in path can forward multiple flows using single aggregation ID.
  - Need more jitter tolerance due to random arrangement in aggregation.
Aggregation Mode – 2

- Fine-grained aggregation / Hierarchy scheduling
  - Hierarchy mapping a group of flows onto the port, by precisely arrange its time window and sequence.
  - Small flows may have larger repeating interval (\(T_{outer}\)), while big flow uses smaller interval (\(T_{inner}\)). This is a very simple aggregation example, flows with other arbitrary measurable intervals may result in irregular aggregation pattern.
  - Latency for different flows depends on its own scheduling interval, that means larger flows can have smaller delay than smaller flows. This is the essential advantage over CQF.
  - This kind of aggregation is hard in time slice computation, but easier for data path. I.e. either TAS or ATS can support this feature.
Cycle Size Problem

- Coarse aggregation: lower speed links to higher speed links
  - If links have the same CQF cycle time, the data is packed OK, but the cycle requires more buffers on the high speed link, and has larger latency than necessary.

- Bridging device needs to store all packets for one cycle, quite large for output port if it is x10/x100 times higher speed than input port.
Cycle Size Problem

- Fine aggregation: lower speed links to higher speed links
  - Hierarchy mapping a group of Streams onto the port, using (say) Asynchronous Traffic Shaping on top of shorter CQF cycles.

  *Cycle sizes diagram*

- Buffer size in bridged is saved for shorter output cycle time;
- ATS (only one example) makes Streams very even.
- Note that Streams 2-5 share the same bandwidth in different (small) cycles.
- This may be difficult and complex, but can be made to work.
Coarse aggregation with shorter cycle?

- Coarse aggregation
  - Encapsulate a group of Streams into one Stream with one reservation.
    
    large CQF cycle  short CQF cycles
    ![Diagram showing coarse aggregation process]

    The packing that was a problem is OK, now, because all are the same Stream (Stream 8).

    Simple coarse aggregation solves the buffer size consumption problem, with some increasing gaps and bursts in merging flows.
Compatibility with Regular Bridge

- Core node (bridge 2 in this example) process all TSN flows just as a regular bridge defined in current TSN standard.
  - No knowledge of flow 1~5 is required.
  - Green flow and flow 6 are scheduled to same egress port.

- Detailed aggregation information is only required on bridges that modify/separate aggregation configuration, e.g. add or remove flows to the aggregated channel.
Deal with Packet Gaps and Bursts

- Coarse-gained aggregation does not care about gaps and bursts in flows.
- Fine-grained aggregation like to maintain characteristic for each individual flow in aggregating.
  - Need buffering to remove jitter
Aggregation Configuration

- Either centralized or distributed computation and configuration is possible, as long as it is all consistent.

- Packet encapsulation for aggregating
  - Add label on each packet in aggregation flow.
  - or use global sync-ed time line to setup aggregation. E.g. in TAS approach, aggregated flow are put into a certain time window.
Summary

- Aggregation:
  - Aggregation is definitely usefully in large scale TSN network with huge amount TSN flows.
  - May need to define control plane protocols to setup/remove aggregation?
    Aggregation is network level mechanism, not a decision for individual bridge.

- But coarse or fine aggregation is a choice to make
  - Timing performance will degrade within coarse grained aggregation channel, using fine-grained aggregation can help avoid interference at expense of hierarchy computation.
  - Maybe suitable for different scenarios?
Thank you