Simulation-Based Evaluation of a Delay-Based Forwarding Concept

Mehmet Cakir
Dept. Computer Science,Hamburg University of Applied Sciences, Germany
mehmet.cakir@haw-hamburg.de

1 Abstract

Quality-of-Service (QoS) mechanisms can prioritize a particular network flow with IntServ. Clemm and Eckert propose Latency-Based Forwarding (LBF) as a novel approach to provide support for high-precision latency objectives [1]. It prioritizes traffic with introducing packet metadata which carries latency objectives. With that metadata different actions will be taken at network nodes. A Proof-of-Concept has been developed using Big Packet Protocol (BPP) [2]. So in contrast to IntServ LBF supports prioritizing specific packets. The purpose is to provide fairness among different applications. For example packets that aren‘t urgent as others can be chosen sent later as the urgent ones. Clemm and Eckert contacted us to enable further investigations for the LBF mechanism with OMNeT++. We want to compare our simulation results to the emulation results [1]. The goal is to validate the results of Clemm and Eckert with ours. In the following the abbreviation DBF for Delay-Based Forwarding is used instead of Latency-Based Forwarding (LBF) because we use it often as a synonym.

DBF works with Service Level Objective (SLO) parameters as packet metadata. These parameters are carried within the packet. SLO parameters cover a time slot in which a packet should arrive and the experienced latency so far. The packet must not arrive earlier or later than the given time slot. In the following the role of the parameters will be explained. Figure 1 shows the algorithm of DBF:

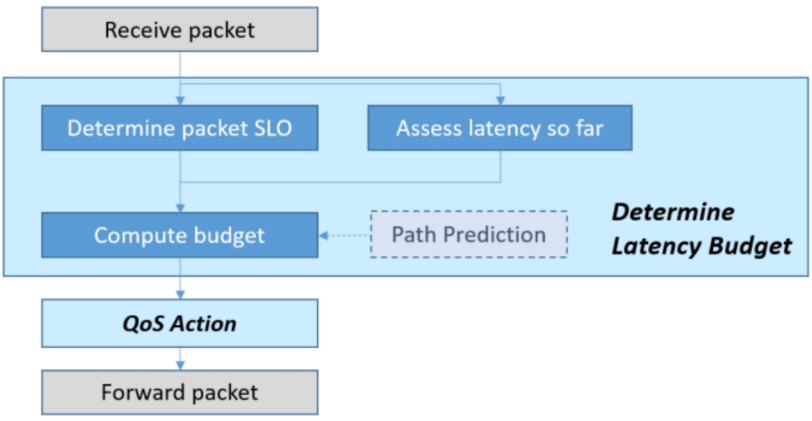

Figure 1: Delay-Based Forwarding - Node Algorithm [1]

When a node receives a packet the experienced delay so far will be determined. Also the local latency budget of the packet will be determined using the SLO parameters. The local latency budget is the time a packet can be queued on a node. Additionally the absolute time will be calculated when the packet can be sent at earliest or have to be sent at latest. If the earliest sending time isn‘t reached yet the packet will be enqueued. Thus a scheduler schedules a packet accordingly to forward it after the queuing time. If the latest sending time is already exceeded after dequeuing the packet it will be dropped.

Figure 2 shows a replica of the network which Clemm and Eckert used for their results [1]. The network consists of nine nodes and seven routers. Nodes starting with a *S* in their name send packets to their corresponding receiver nodes starting with a *R* in their name. Node *S1_R1* sends DBF packets to node *R1*. The traffic will be queued in the DBF manner by the routers *FWD1*, *FWD2*, *FWD3* and *FWD4*. All other senders send non DBF packets which will be handled as Best-Effort (BE) traffic. BE traffic has the lowest priority.

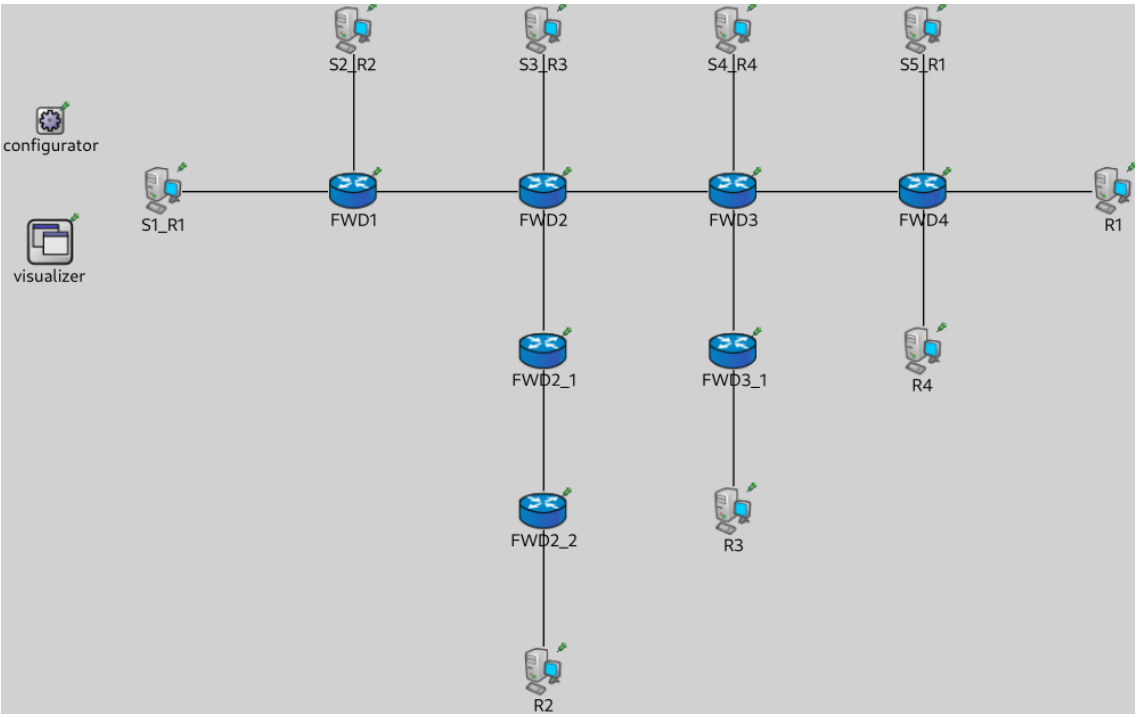

Figure 2: Delay-Based Forwarding Simulation Network

The implementation is realized in OMNeT++ version 5.6.2 with the INET framework version 4.2.0. SLO parameters will be carried as Type-Length-Value (TLV) options in the IPv4 header instead of introducing another header. This way it was less complex to implement because the INET framework expects the IPv4 header at a certain place in a frame at other layers. The IPv4 header carries only the parameters *Dmin*, *Dmax* and *eDelay* (Table 1):

Table 1: SLO Parameters

SLO Parameter	Description
Dmin	The lower bound of the time slot a packet can arrive at the receiver soonest.
Dmax	The upper bound of the time slot a packet can arrive at the receiver latest.
eDelay	The experienced delay of the packet so far.

In our nodes we modified Layer 3 and Layer 2. This way the DBF mechanism stays transparent to higher layers. Therefore we added three modules:

- The *Ingress* module extends Layer 2. Each DBF packet received from the network is timestamped by the *Ingress* module.
- The *Computer* module extends Layer 3. The *Computer* module computes further parameters to either drop the packet or determine the earliest/latest sending time and maximum/minimum local queuing budget. DBF packets are led through the

Computer module in two cases. One case are packets coming from Layer 4. SLO parameters are added to these packets. The other case is when the packet comes from layer 2 and must be forwarded to another node.

- The *Egress* module extends Layer 2. Each DBF packet coming from Layer 3 is classified, queued and possibly forwarded or dropped by the *Egress* module.

In the following the queuing in the *Egress* module is explained. Therefore two parameters calculated by the *Computer* module are used for the queuing process. These parameters are carried as packet tags on the actual node:

- *tMin*: The absolute time the packet may be forwarded to the next node at earliest.
- *tMax*: The absolute time the packet must be forwarded to the next node at latest.

The *Egress* module includes the following modules:

- *Prority Queue*: The *Prority Queue* prioritize enqueued DBF packets by their calculated *tMin* parameter. The lower *tMin* the higher the priority. DBF packets are preferred over non DBF packets.
- *Classifier*: The *Classifier* is the first module packets are led through when they are given to the *Egress* module. Packets will be classified if they should be handled in a DBF or Best-Effort (BE) manner. If a packet should be handled in a DBF manner first the delta of *tMax* and *tMin* will be calculated:

$$\Delta tDelta = tMax - tMin$$

  Using *tDelta* a lookup determines if a corresponding *Prority Queue* already exists for it. If not it will be dynamically created. The lower the *tDelta* the higher the priority of the *Prority Queue*. We foresee to delete queues if they get empty. Packets that are non DBF packets will be enqueued in a low priority infinite queue. A threshold for a high *tDelta* can be specified before the simulation starts. DBF packets with a high *tDelta* are also enqueued in the low priority queue
- *Scheduler*: The *Scheduler* dequeues packets from a *Prority Queue* starting from the queue with the highest priority. A DBF packet will be scheduled with its *tMin*. If its *tMin* is already exceeded the packet is sent immediately. BE packets are sent immediately. Before a packet will be dequeued, already expired packets in the queue will be dropped. The scheduler looks for packets in the queue when a new packet will be pushed in the queue and the Ethernet interface is idle or whenever the Ethernet interface becomes idle. If there is a packet with a higher priority then the possibly scheduled is canceled and the other packet will be scheduled or sent immediately.

We implemented the DBF mechanism using Layer 3 and Layer 2. This design makes DBF transparent to other layers. Additionally we compare our results with the emulation to show that our implementation matches the design.

References

[1] A. Clemm and T. Eckert. High-precision latency forwarding over packet-programmable networks. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–8, 2020.

[2] Richard Li, Kiran Makhijani, Hamed Yousefi, Cedric Westphal, Lijun Dong, Tim Wauters, and Filip De Turck. A framework for qualitative communications using big packet protocol. In *Proceedings of the ACM SIGCOMM 2019 Workshop on Networking for Emerging Applications and Technologies*, pages 22–28, 2019.