Evaluation of the Time-Aware Priority Queueing Discipline with Regard to Time-Sensitive Networking in Particular IEEE 802.1Qbv

Manish Kumar, Martin Boehm, Jannis Ohms, Oleksandr Shulha and Olaf Gebauer
Research Group Communication Systems, Ostfalia University of Applied Sciences, Salzdahlumer Str.46/48, D-38302, Wolfenbüttel, Germany
{m.kumar, ma.boehm, jannis.ohms2, o.shulha, ola.gebauer}@ostfalia.de

Keywords: TSN, Industry 4.0, Scheduling, Real-Time, QoS, M2M.

Abstract: Within the evolvement of Industry 4.0, the need for flexible real-time communication technologies emerges. Time-Sensitive Networking enables deterministic communication in IEEE 802 networks. The Time-Sensitive Networking Working Group published a set of standards whose implementation is in progress. IEEE 802.1Qbv standard introduces the concept of Time-Aware Shaping (TAS). TAS enables determinism by dividing traffic in different preconfigured time-slots configured in a Gate-Control-Lists (GCL). Intel recently released a time-aware scheduler based on the IEEE 802.1Qbv standard. This paper investigates Intel's implementation. In order to test the scheduler a testbed is created. The scheduler is configured to filter and group different types of incoming packets in queues. The packets are transmitted in accordance to the configured GCL. Ingress and Egress traffic of the scheduler are analyzed in accordance to the configured time-slots. The results show, that packets which are arriving outside of their respective time-slot are buffered and transmitted in the beginning of their time-slots. It shows that no traffic interfered with the incorrect time-slots.

1 INTRODUCTION

Industry 4.0 is rapidly evolving and raises the need for real-time M2M communication. However, there are already communication technologies for several decades, which provide determinism. PROFINET as an example offers the ability for deterministic networking. One disadvantage of these technologies is the vendor lock-in, which strictly limits the number of supported hardware. Furthermore, reconfigurations of these systems are costly and inefficient.

The Time-Sensitive Networking (TSN) Task Group, established by IEEE, targets these problems by developing standards for flexible deterministic communication in IEEE 802 networks. One of the most important standards for real-time communication besides precise time-synchronization is the scheduling of the network traffic for each device. The IEEE 802.1Qbv (Enhancements for Scheduled Traffic) standard, offers the ability to schedule traffic based on the traffic type [7]. So far, there are theoretical investigations about the scheduler [1].

This paper investigates an implementation of the IEEE 802.1Qbv standard. Chapter 2 gives an overview of the related work. Later, Chapter 3 gives an overview of TSN and its standards. Chapter 4 explains the IEEE 802.1Qbv standard in more detail. Chapter 5 presents a test-setup to test the timing-characteristics of the Ingress and Egress traffic for the scheduler. The discussion of the results takes place in Chapter 6. Chapter 7 wraps up the topic and exposes gaps and questions for future work.

2 RELATED WORK

J. Vila-Carbó et al. [2] show how to use queuing disciplines to provide bandwidth limitation for traffic classes in real-time communications. Their results show, that it is possible to avoid collisions between traffic classes and reduce delays about 30%. This reduction of delay does not provide the determinism required for industrial automation.

Craciunas et al. [3] analyze the algorithmic complexity of automated configuration synthesis for

TSN. Their results show, that the problem is NP complex. This opens the need for efficient heuristics.

In order to provide an auto configuration mechanism, all updates need to be timely synchronized to avoid unwanted network configuration states. This problem is addressed by Mizrahi et al. [4].

In [5], Gutiérrez et al. propose MQPRIO as a queueing discipline for the Linux operating system. Their work focuses on the use of a patched Linux kernel, called Real-time Preemption patch (PREEMPT-RT), for real-time communication in robotic applications.

In order to communicate between a TSN and other communication systems, Böhm et al. [6] propose an architectural design for a gateway which connects TSN with Software-Defined Networking resp. OpenFlow. The gateway forwards packets between the networks while preserving the real-time capabilities of the TSN network. The scheduler examined in this paper is one of the main components described in their requirements.

This paper investigates an implementation of the IEEE 802.1Qbv standard for "Enhancements for Scheduled Traffic" [7]. The functionalities and preciseness of the time-aware scheduler will be examined.

3 TIME SENSITIVE NETWORKING

In 2012, the Audio/Video Bridging Task Group which developed standards for time-synchronized low latency streaming services, renamed themselves to Time-Sensitive Networking Task Group. They focus on real-time communication through IEEE 802 networks and provide a set of standards. Depending on the case of application, multiple standards can be used together.

The combination of time synchronization (IEEE 802.1AS-Rev - Timing and Synchronization for Time-Sensitive Applications) and time-aware traffic shaping (IEEE 802.1Qbv - Enhancements for Scheduled Traffic) enables determinism, by processing different traffic types in their respective time-slots. Guard bands block time before each time slice to prevent conflicts of overlapping packets. Due to the waste of time for each guard band, where no traffic is allowed to be transmitted, they introduced Frame Preemption (IEEE802.1 Qbu - Frame Preemption) [10] to reduce the size of guard bands. This standard enables transmission of frames to be interrupted and later resumed.

TSN also offers the ability to reconfigure all network devices dynamically. As shown in Figure 1, end devices can request the Centralized User Configuration (CUC) for their specific deterministic communication flow including packet size, frequency etc. (IEEE 802.1Qcc - Stream Reservation Protocol (SRP)) [8]. The Centralized Network

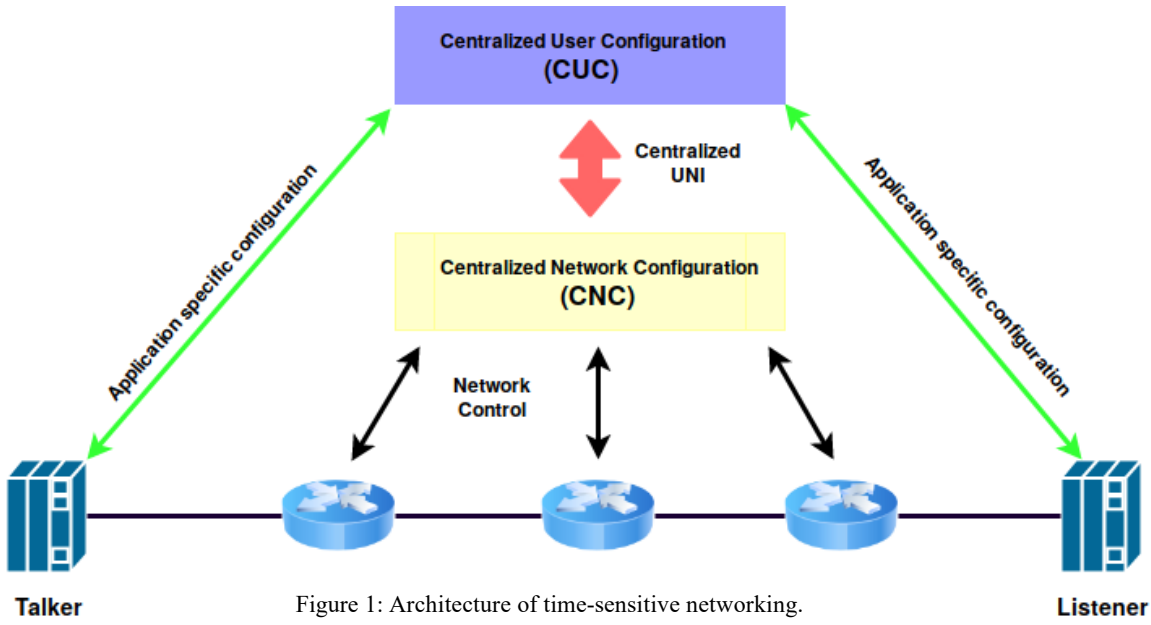

Figure 1: Architecture of time-sensitive networking.

Configuration (CNC) on the other side, which has a global view on the network, calculates configurations including time-slots size, VLAN to map traffic etc. Flow requests can also be rejected if resources are not sufficient. All devices get updated to the new configuration (IEEE 802.1Qcp - YANG Data Model) [9].

The majority of standards have been published. A few standards are not finalized yet. The next chapter introduces the IEEE 802.1Qbv TAS in detail.

4 TIME-AWARE SHAPING

The basic architecture of IEEE 802.1Qbv is visualized in Figure 2. The time-aware shaper consists of 1 to 8 queues. Each queue has a transmission selection algorithm which selects the next packet transmitted from the queue. This can be a queueing disciplines like first-in, first-out (FIFO) or token bucket filter (TBF). The state of a gate can either be open or close, only open gates transmit packets. A time-aware gate opens and closes according to its configured time. The schedule of the gate states is specified in the GCL. Each entry of the list consists of a set of gate states and their duration. The entries are repeated cyclically. Furthermore, the standard requires another parameter called base-time which specifies when the execution of the GCL starts. This parameter is used to assure that each device in a TSN network starts their schedule at the same time to avoid delays.

One open source implementation, developed by Intel, for the IEEE 802.1Qbv is called Time-Aware Priority (TAPRIO). It is a classful queueing discipline for the Linux Kernel. It timely opens and closes gates respective to the current entry of the GCL.

The next chapter shows the test-setup to test the functionality of the scheduler. Test-cases are presented.

5 TEST-SETUP AND TEST-CASES

This chapter introduces a test-setup as well as test-cases for TAPRIO, an implementation of the IEEE 802.1Qbv standard.

The test-setup is visualized in Figure . All systems are based on Ubuntu 18.04. TAPRIO is not part of the mainline Linux kernel. A custom version of the Kernel (GNU/Linux 4.19.0-rc5 x86_64) has been compiled. The bridge is equipped with one Intel i210 Ethernet controller which provides hardware offloading and precise timestamping for the TAPRIO queueing discipline. The i210 controller is used as an egress port. The ingress and egress traffic of the bridge is mirrored by two hardware network taps and recorded in a measurement system using Wireshark. Based on the recorded traffic the bridge delay and the time

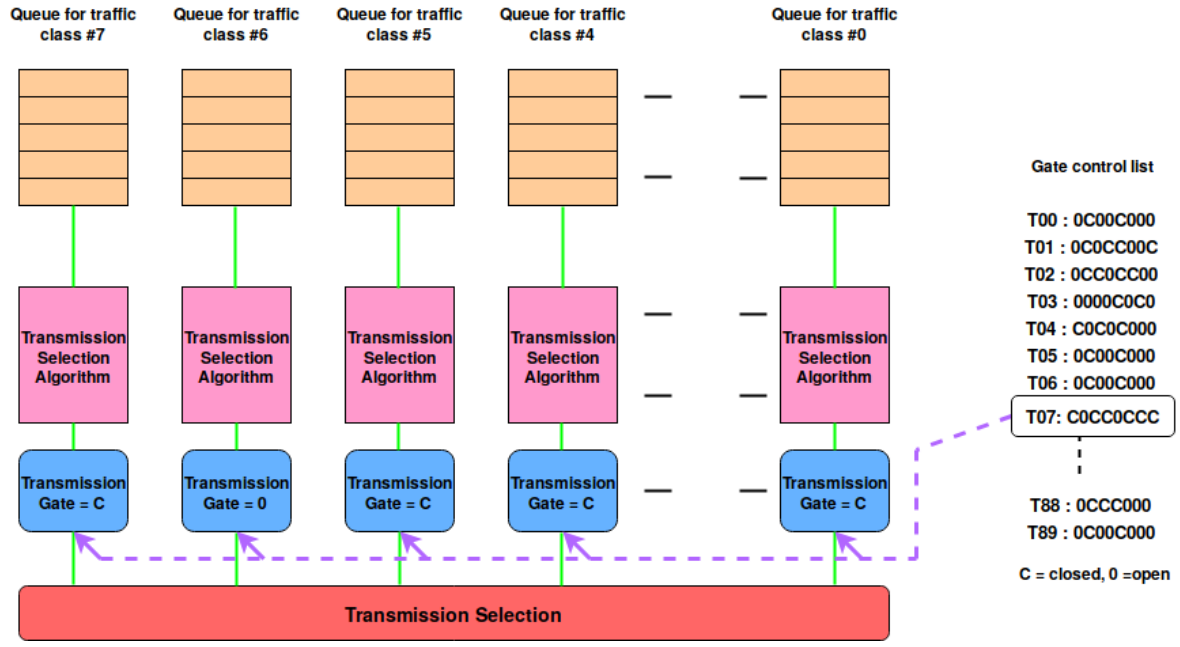

Figure 2: Selection for transmission-gates.

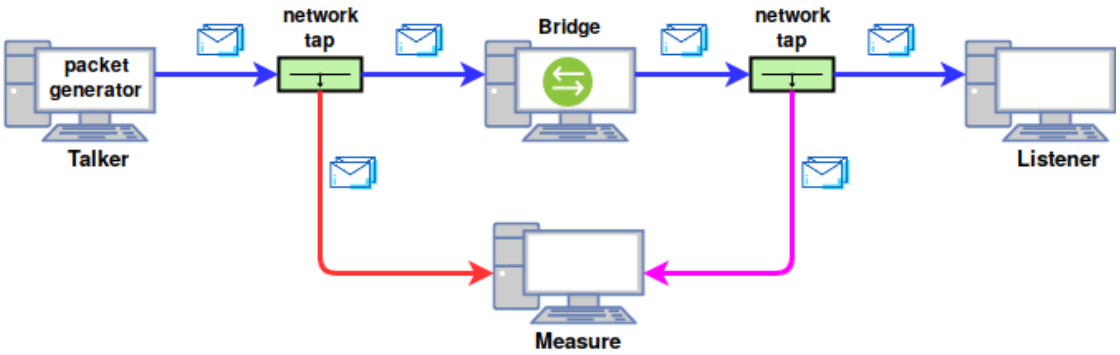

Figure 3: Test-setup for time-aware scheduling.

distribution of ingress/egress traffic is calculated. It has to be noted, that no real-time Linux like Linux RT or Industrial Linux has been used for the setup.

The Talker creates two UDP packets every 10 milliseconds. The packets use two different IP TOS values: 0x00 and 0x08. Each packet contains a unique number as payload for later mapping. The i210 controller supports four transmission queues while the test-cases use three time-slots for three different traffic classes. The configuration of TAPRIO, which is shown in Figure 4, consists of three queues Q_0 - Q_2. Each queue is opened for 30 milliseconds starting with Q_0. Traffic with a TOS value of 0x00 is assigned to Q_2 and traffic with a TOS value of 0x08 is assigned to Q_1. No traffic is assigned to Q_0, to make sure, that no traffic interferes with this queue and the respective slot always remains empty. The base-time parameter, which represents the starting point of the schedule, is configured to be a point in time in the near future.

```
qdisc replace dev eth0 parent root handle 100 taprio
    num_tc 3
    map 2 2 1 2 0 2 2 2 2 2 2 2 2 2 2 2
    queues 1@0 1@1 2@2
    base-time $BASE_TIME
    sched-entry S 01 30000000
    sched-entry S 02 30000000
    sched-entry S 04 30000000
    clockid CLOCK_REALTIME
```

Figure 4: Configuration of the time-aware priority queueing discipline.

The setup helps to determine if the timing characteristics of the GCL parameters of the TAPRIO queueing disciplines works properly. It measures packets before and after they are scheduled to calculate the delay between both points. This shows the distribution of processing plus waiting time for each packet. The captured egress packets show, if they are transmitted in their respective time-slots.

6 DISCUSSION OF THE RESULTS

This chapter discusses the results of the test-cases presented in the previous chapter.

The ingress time of each packet is visualized in Figure 5. Each colored dot represents a packet. Two packets enter the bridge every ten milliseconds with two different TOS fields. The x-axis shows the unique packet number of each packet while the y-axis shows the arrival time of the packets. It shows, that there is a linear rise of unscheduled incoming packets over time.

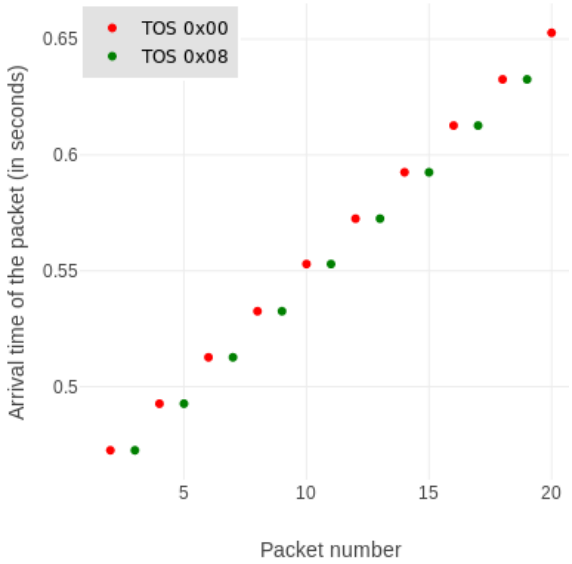

Figure 5: Ingress traffic with two TOS felds over time.

Figure 6 shows the egress time of each packet. The axes are the same as in Figure 5. In the graph, each horizontal line represents the start of a new time-slot. The grey highlighted region represents an unused time-slot.

Both, the green area as well as the red area, always start sending packet immediately after their time-slot starts. The traffic, which is not sent directly in the time-slot, is buffered and sent when the next allocated time-slot starts. In the graph, linear sequences of packets show the transmission of buffered packets. Packets which arrive in their respective time-slot are transferred directly, which can be seen between the horizontal lines. The graph also shows that no packet gets transmitted outside of the appropriate time-slots.

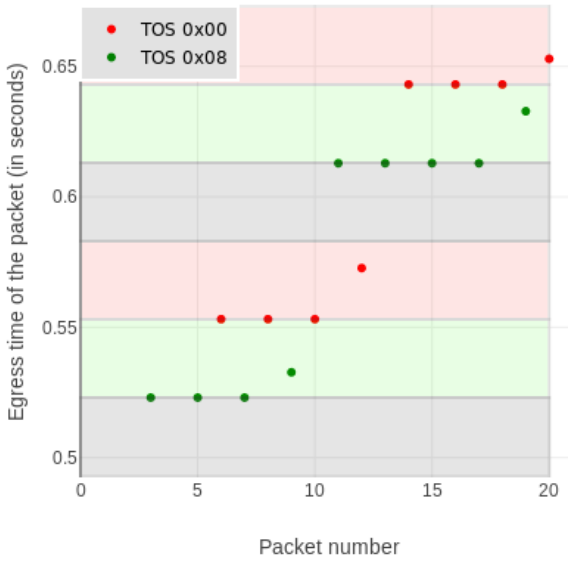

Figure 6: TOS based prioritized egress scheduled traffic with 3 time-slots.

The difference between egress and ingress time is visualized in Figure 7. It shows the amount of time each packet spends in the scheduled bridge. Since there are three slots with 30 milliseconds time-window each, the upper bound delay of the bridge should be maximum 60 milliseconds due to a maximum buffer time of two slots. This assumption can be validated by Figure 7.

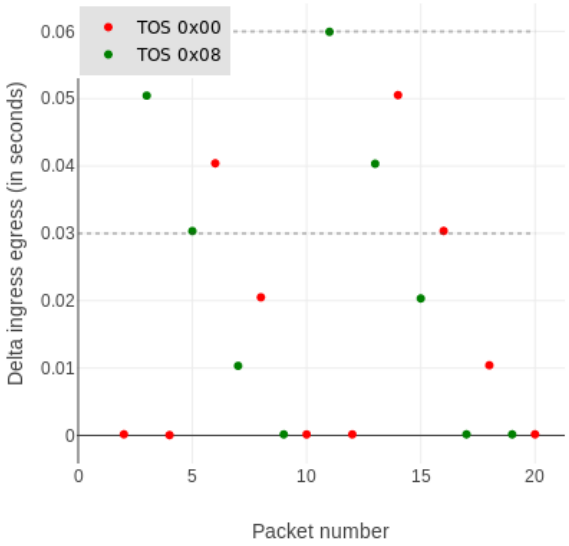

Figure 7: Distribution of the delta (difference between ingress- and egress-time of the traffic).

7 CONCLUSION AND FUTURE WORK

The results presented in this paper show, that the implementation of the IEEE 802.1Qbv scheduler works as expected. This has been validated with a test-setup by analyzing the traffic before and after is has been scheduled. It shows, that different traffic classes never interfere with the wrong time-slots. It also shows that packets, which arrived outside of their time-slots are buffered and processed as soon as their time-slots begin.

There are a few things which should be noted. Real-time traffic should be sent in their respective time-slot to avoid waiting time in the buffer. In this case, the Talker needs to be time-aware too. Due to the usage of a buffer, there is a possibility that the buffer is overfilled with packets. This can be a huge security risk.

Another topic, which is not mentioned in the standard is the assignment of queues for time-synchronization traffic. The impact of scheduled time-synchronization packets should be investigated in the future.

ACKNOWLEDGMENTS

This work was partly funded by the Ministry for Science and Culture of Lower Saxony as a part of the research project SecuRIn (VWZN3224) and the

Federal Ministry for Education and Research within the KMU-innovativ program as a part of MONAT (16KIS0782).

REFERENCES

[1] N. G. Nayak, F. Dürr, and K. Rothermel, "Routing algorithms for IEEE802.1Qbv networks," ACM SIGBED Review, no. 15(3), pp. 13-18, 2018.

[2] J. Vila-Carbó, J. Tur-Masanet, and E. Hernandez-Orallo, "An evaluation of switched ethernet and Linux traffic control for real-time transmission," IEEE International Conference on Emerging Technologies and Factory Automation, 2008, pp. 400-407.

[3] S. S. Craciunas, R. S. Oliver, M. Chmelík, 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, ACM, 2016, pp. 183-192.

[4] T. Mizrahi, E. Saat, and Y. Moses, "Timed consistent network updates in software-defined networks," IEEE/ACM Transactions on Networking, no. 24(6), pp. 3412-3425, 2016.

[5] C. S. V. Gutiérrez, L. U. S. Juan, I. Z. Ugarte, and V. M. Vilches, "Real-time Linux communications: an evaluation of the Linux communication stack for real-time robotic applications," arXiv preprint arXiv: 1808.10821, 2018.

[6] M. Böhm, J. Ohms, O. Gebauer, and D. Wermser, "Architectural design of a TSN to SDN gateway in the context of industry 4.0," 23. VDE/ITG Fachtagung Mobilkommunikation - Technologien und Anwendungen, 2018.

[7] IEEE 802.1Qbv, IEEE Standard for Local and metropolitan area networks - Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic, 2016.

[8] IEEE 802.1Qcc, IEEE Draft Standard for Local and metropolitan area networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks Amendment: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements, 2017.

[9] IEEE 802.1Qcp, IEEE Standard for Local and metropolitan area networks - Bridges and Bridged Networks - Amendment 30: YANG Data Model, 2018.

[10] IEEE 802.1Qbu, IEEE Standard for Local and metropolitan area networks - Bridges and Bridged Networks - Amendment 26: Frame Preemption, 2016.