http://dx.doi.org/10.25673/111737

Investigation in Automatic Fault Detection for Scheduled Traffic and Frame Preemption in Time-Sensitive Networks

Tobias Ferfers[1], Sebastian Schriegel[1] Jürgen Jasperneite[1]

Abstract: A thorough network diagnosis is essential to cutting down the cost of network downtime in heterogeneous, time-sensitive Ethernet networks. It appears that many Time-Sensitive Networking mechanisms do not provide sufficient information about possible error sources, error recognition, or error causes. This paper examines possible symptoms and error sources of Frame Preemption and how to detect them automatically. Moreover, it examines the limitations and functionality of the Scheduled Traffic Anomaly Detection algorithm (STADA) by utilizing a test network. This research provides assistance to manufacturers of industrial automation devices, experts, and network administrators in performing FDD and root-cause analysis for Scheduled Traffic and Frame Preemption faults in Time-Sensitive networks.

Keywords: Scheduled Traffic, Time-Sensitive Networking, Fault Detection and Diagnosis, Frame Preemption

1 Motivation

Time-Sensitive Networking (TSN) for Ethernet networks introduces the possibility of Quality of Service (QoS) in Ethernet networks like, deterministic and low-latency real-time communication for control application e.g., PROFINET over TSN [Pr23]. The key mechanisms of TSN in industrial communication networks are: Time Synchronization (IEEE 802.1AS), Enhancements for Scheduled Traffic (IEEE 802.1Q) and Frame Preemption (IEEE 802.1Q). During the lifetime of a TSN device, faults may occur. Possible faults of devices, products or production plants are physical, hardware, software, aging, design fault, operating error, configuration error or production error, but also faulty network cable or rough industrial environments, temperature, humidity and many more. The additional challenge in TSN networks is the consideration of the time behavior in the network, especially in case of a fault. The previously mentioned faults can lead to (network) downtime. The cost of the downtime heavily depends on the industry branch as well as the company's size and has a large variance, according to the Ponemon Institute the average cost of network downtime in data center is about $9000 per minute [Co16].

IEEE 61158-2017 "IEEE Standard for Industrial Real-Time Communication" considers possible error sources / error symptoms, error recognition and the error handling for some

[1] Fraunhofer IOSB-INA, Campusallee 1, 32657, Lemgo {tobias.ferfers, sebastian.schriegel, jürgen.jasperneite}@iosb-ina.fraunhofer.de

components of this communication technology e.g., data link layer and physical layer [Ie17]. In comparison, most TSN standards do not provide this kind of error recognition and error handling, hence expert knowledge and experience is necessary for troubleshooting. In case of a fault, the fault detection and diagnosis (FDD) [GDC15a, GDC15b] and troubleshooting can therefore take more time in TSN networks, extend the production downtime (planned or unplanned) and increase the revenue lost.

The primary objective of FDD and root-cause analysis is to facilitate the troubleshooting process for users in the event of a fault or failure. [FSJ23] et.al. presented a concept for the automatic root-cause analysis in time-sensitive networks based on fault models. Fault models connect the symptoms of faults. to their root causes. The current state of the physical network (netload, protocol alarms, runtime measurements) is compared to network models that contain the nominal state of the network (protocols, topology, netload, schedules, configuration). An anomaly detector uses FDD technologies to detect symptoms in the physical network, then a reasoner uses fault models to find possible root causes for troubleshooting and presents the possible root causes and their probability to the network operator. To create such a system, it is necessary to investigate TSN mechanism regarding possible faults, their symptoms and root-causes. What faults in TSN key technologies can occur? How to detect faults of TSN key technologies automatically during runtime? The aim of this paper is to investigate possible faults, their symptoms and automatic detection of Frame Preemption mechanism and to evaluate the functionality and limitations of the Scheduled Traffic Anomaly Detection Algorithm (STADA) [FSJ23].

The first section describes State of the Art Diagnosis in industrial communication. The second section describes the functionality of Frame Preemption and Scheduled Traffic. The third section investigates Frame Preemption mechanism and describes possible faults and how to detect them. The fourth section handles the evaluation of STADA including a description of the algorithm, the test setup and method as well as the results. The final section is conclusion and future work. This work will support vendors of industrial automation devices, experts and administrators of TSN networks during FDD and root-cause analysis for Scheduled Traffic and Frame Preemption faults.

2 State of the Art Diagnosis

The first section of the chapter, highlights the most important terms as well as general FDD techniques. In the second part of the chapter, three examples of diagnosis in industrial communication are explained in greater detail. In the 1990s, Isermann et al. defined terms in the field of Fault Detection and diagnosis (FDD), e.g., faults, fault diagnosis, fault management, and more [IB96]. *Fault detection* is the determination of the fault's presence in a system and the time of detection. *Fault isolation* is the determination of the kind, location, and time of the detection of a fault. *Fault identification* describes the determination of the size and time-variant behavior of a fault. After a fault diagnosis, the location, size, and type of fault are known, but the root cause and the actions to be taken

for rectification are unknown. Sometimes FDD and anomaly detection are used as synonyms; in principle, it is about being able to detect deviations from the normal state. The field of fault detection and diagnosis (FDD) is divided into four categories: signal-based, model-based, data driven and hybrid methods [GCD15a, GCD15b]. Signal-based methods rest upon signals that are somehow connected to the fault in time, frequency, or time-frequency domain and utilize, for example statistical information, e.g., mean value, variance, or kurtosis, for example [EM13]. Model-based approaches consider an exact (mathematical) representation of the system or process, commonly applied to physical processes, for example applied on LAN in [Fo02]. The data-driven methods are divided into two methods: statistical analysis and artificial intelligence, e.g. [An18]. Data-driven techniques use available information about the devices or network, and are often considered as an alternative to model-based approaches because there is no detailed modeling necessary.

The "IEEE61158 Standard for Industrial Hard Real-Time Communication" provides additional information and even recommendations regarding error management at both the data link layer and physical layer [Ie17]. The standard covers possible error sources, error recognition, error handling, and error registration. Loss of link, buffer overflows or underruns, timing violations for received frames, transmission errors, collisions, frame loss, incorrect physical Ethernet operating mode, and numerous other issues that are addressed in the standard. In the following, the “Loss of PollResponses” will be explained in more detail. This error indicates that no PollRepsonse frame was received in the current time slot. The standard describes multiple categories of possible error sources from physical errors e.g., loss of link Rx buffer overflow. Other possible sources of errors are, for example, defective components in the network or the use of devices whose latency does not meet the requirements. This fault is supposed to noticed in the management node cycle state machine and can be recognized if the sot timer expires and no frame was received in the slot. When a frame loss is detected, several actions are taken: Notification of other devices or components, exclusion from isochronous communication, or error logging.

PROFIENT IRT is a communication profile of PROFINET that outlines a highly synchronized (isochronous) and stringent communication protocol, meticulously engineered from the topology to the cable delays [Pr22]. Since, PROFINET IRT is highly engineered, the protocol implements mechanisms to detect changes in the topology or cable length during runtime. One part of error handling is done at the application level with the “SignOfLife” application, which has a counter that increases every cycle, and the mechanism checks through this counter if frames were lost in a cycle. A threshold is set by the user for the number of frames that are acceptable to lose. Normal errors are handled as in the non-isochronous PROFINET protocol: if a module or submodule detects a fault, an alarm is sent to the upper layer, e.g., if data processing is not finished when the next cycle starts. Since, PROFINET IRT depends very much on synchronization, details about sync errors are also given in the specification. More precisely, it describes how sync errors should be handled on the provider and consumer sides. For example, if an out-of-sync error occurs, access to data is refused, or it describes error codes, e.g., jitter out of bounds

or no sync telegram within rules received. PROFINET IRT diagnoses the communication at the application level and has an alarm system for notification, but in-depth diagnosis and providing the user with the root cause are not included.

"IEC/IEEE 60802 TSN Profile for Industrial Automation" describes a set of rules for time-sensitive networking in the industrial automation field [Ie23]. For diagnosis, the profile suggests observing the YANG data model representation in the local database of the component and observing the available objects. Furthermore, the profile defines a subscriber-based notification mechanism and corresponding events, e.g., loss of link, loss of sync or periodic statistics. As with other protocols, there are mechanisms to detect certain errors, but possible causes or actions to clarify them do not exist.

3 TSN-Mechanisms Scheduled Traffic and Frame Preemption

3.1 Scheduled Traffic

The Enhancements for Scheduled Traffic (IEEE 802.1Qbv) allows the transmission of each transmission queue to be scheduled to a relative time. Transmission gates are associated with each queue [Ie16a]. The state of the gate determines whether frames can be selected for transmission (open or closed). Every port has a gate control list with ordered gate operations, and for each entry in the gate control list, there is a traffic class assigned. Depending on whether frame preemption is used or not, the gate operation of each entry allows preemption of frames. Scheduled traffic leads to a slot-based communication where one or more traffic classes are assigned to the slots (see Fig. 1). The most important parameters for scheduled traffic are: base-time (start time of the schedule), cycle time, ControlList, ControlListLength and the CycleTimeExtension.

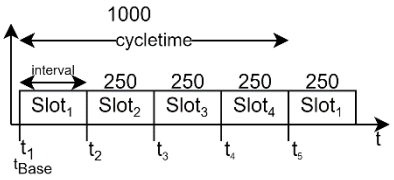


Fig. 1: Scheduled Traffic 802.1Qbv

3.2 Frame Preemption

IEEE 802.1Qbu – Enhancements for frame Preemption and IEEE 802.1br are a set of features that allows higher priority frames to interrupt the transmission of lower priority frames and resume it later [Ie16b]. Frame preemption is implemented at data link layer according to the ISO/OSI model. The MAC layer provides two services: the preemptable

MAC (pMAC) and the express MAC (eMAC). A MAC merge layer merges these two MAC services back together and preempts preemptable traffic currently being transmitted or prevents the start of the transmission of preemptable traffic. When the preemption capability is inactive, the MAC Merge sublayer does not allow express traffic to interrupt a frame provided by the pMAC service interface. In the MAC Merge sublayer, a special packet format is used called mPacket. (see Fig. 2).

a) format of express packet, complete preemptable packet or an initial fragment of a packet

b) format of a fragment of a packet

Fig. 2 : mPacket format MAC merge sublayer

The Preamble of an mPacket is identical to a MAC Preamble, the Start Frame Delimiter (SFD) is replaced by the Start mPacket Delimiter (SMD) value and identifies the type of mPacket frame e.g., verify, respond, express packet, preemptable packet start (SMD-S0 to SMD-S3), or a continuation fragment (SMD-C0 to SMD-C3). The frag_count in a fragment is a modulo-4 counter that increments each continuous fragment of a preempted mPacket. The frag_count is used to detect mPacket reassembly errors by enabling detection of the loss of up to three packets. As seen in Fig. 2 the frag_count is only included at a continuous fragment of a mPacket. The CRC field contains a cyclic redundancy check (CRC) and an indication of whether this is the final mPacket of a frame. In the final mPacket of a frame the CRC field contains the CRC of the MAC frame. For other mPackets the CRC field contains an mCRC (mPacket CRC) value for that specific mPacket. Generally, the preemption capability is enabled on the transmit direction only if it's ensured that the link partner also supports the frame preemption capability. The process of discovering the support on the link partner relies on the exchange of additional Ethernet capability TLV in the LLDP frame. The mechanism is only enabled if the support was announced before and the preemption mechanism is disabled in case of a link failure. Only if the frame preemption functionality has been made known beforehand the verification process will be triggered. In this process a verify mPacket is sent and a respond packet is expected from the link partner. If the frame preemption capability is enabled but has not been verified yet, the MAC merge sublayer indicates verification process. Verification can be disabled, this is useful for engineered networks.

4 Possible Faults and Symptoms Frame Preemption

In order to diagnose the previously outlined Frame Preemption mechanism, it is imperative to distinguish between two distinct stages: initialization during the ongoing verification process (static) and diagnosis subsequent to successful verification at runtime (dynamic). Static errors describe the fact that the verification process was not successful, which can have several causes. This phenomenon may manifest itself in the absence of LLDP frames or the absence of the additional Ethernet availability for frame preemption in the frame. Possible root causes are an incorrect implementation or configuration of the device or an increased network load could that leads to frame loss of LLD frames or verification mPackets. Therefore, devices or additional measuring equipment could check if the verification was successful, e.g., during startup or after a link failure. The second possible error category pertains to dynamic errors, which may arise in the event that the frame preemption verification was successful and the mechanism is operational and functioning. The standard already provides capabilities to indicate faults, like various counters and status variables to check if the mechanism is working correctly e.g., aMACMergeAssErrorCount, count of MAC frame reassembly errors on receiver side or the MACMergeFrameSmdErrorCount is a counter of the received MAC frames / frame fragments rejected due to unknown SMD value or arriving of SMD-C when no frame in progress. For detailed diagnosis access to these counters and the current device configuration is necessary. A faulty implementation could also be the reason for incorrectly sent or re-assembled fragments, which could be noticed by the link partner (missing frames etc.). Further on, it is possible to observe the jitter of the real-time network traffic, if frame preemption should be configured but is not and, enough best effort traffic is going through the network this faulty configuration could be noticed.

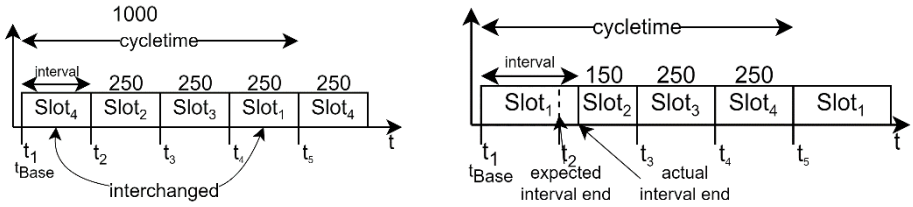
Some of the evaluation of the functionality can be done by observing counters or variables already defined in the frame preemption standard, especially for the runtime errors. Additional measuring equipment or an extension of the devices is necessary to detect faults in the verification process or the jitter of real-time traffic. Furthermore, access through an API must be granted for network administrators or central diagnosis systems to evaluate the quality of the network. This must be integrated into the driver or firmware of the devices to provide this information, if it is not already the case through Management Information Base (MIB).

5 Evaluation of STADA

5.1 Description STADA

The Scheduled Traffic Anomaly Detection Algorithm (STADA), presented in [Fe23], aims to validate the correct scheduled traffic configuration based on the transmit timestamp (tx_timestamp) of a frame at runtime, the desired scheduled traffic

configuration (base time of the schedule, traffic class for each slot). Based on the transmit timestamp the exceeded time in the current cycle can be calculated: *time_elapsed = (tx_timestamp – base_time) % cycle_time*. With the desired configuration and the elapsed time in the current cycle the current slot of the schedule can be determind and which frames are allowed. Then the algorithm compares whether the allowed and actual frame type match. If the frame types do not match, further comparison is done to determine if the frame type is even configured or if the interval is too short or too long or a wrong order of time slots is given. But how does this algorithm performs in a test setup and network?



(a) faulty slot order (b) faulty interval length

Figure 1: Possible faults scheduled traffic

5.2 Test setup

The test setup for the evaluation of STADA consists of a TSN controller, TSN switch, TSN device and additional measuring equipment. To analyze the traffic on the wire a network Test access point (TAP) is set between the TSN controller and the TSN switch. The TSN controller, TSN device and the measuring device is Linux based with Intel I225 network cards. Traffic on the controller is generated by a dummy application. Generally, it is possible to implement the STADA in the device driver or firmware of the network component, except of the API to the (Linux) kernel this is highly vendor specific and closed source. Therefore, STADA was implemented on additional measuring equipment and the transmit timestamp is calculated as *transmit_timestamp = receive_timestamp – delay*. The delay was determined by the delay of the TAP and previous measurements. The measuring equipment is connected to the TSN switch for synchronization and to the network TAP for diagnosis. The schedule for evaluation has three slots: network management (PTP and LLDP), second slot real-time traffic (PROFINET) and the third slot best effort (IPv4) as seen in Tab. *1*.

Slot number	Length [µs]	Traffic type
1	200	PTP, LLDP
2	250	PROFINET
3	550	Best effort (IPv4)

Tab. 1: nominal scheduled traffic configuration

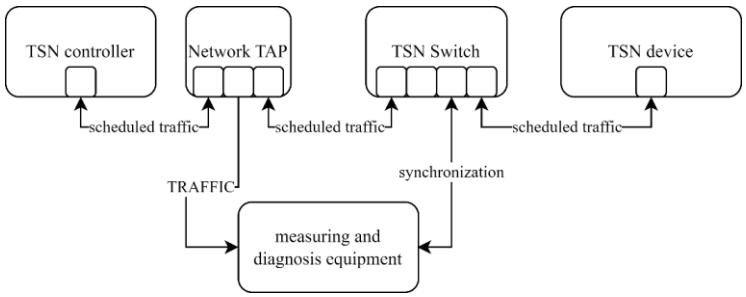


Fig. 3: Test setup for STADA

5.3 Method

As described in previous work [Fe23], possible faults for scheduled traffic are a wrong order of slots and an incorrect interval length (too short or too long). For the determination of the possible combinations for testing, it was specified that no repetitions may occur in the order. With the test schedule from the test setup this results in the following possible combination shown in Tab. 2. A short interval length is defined as half of the normal interval length, the long interval length is defined as double of the normal slot length.

Slot order	Slot 1 interval	Slot 2 interval	Slot interval 3
123	Short	Short	short
132	Normal	Normal	Normal
213	Long	Long	Long
231			
312			
321			

Tab. 2: scheduling configuration

The possible combinations based on Tab. 2 are 6x3x3x3 = 162 combinations. In the literature there are methods about reducing the test cases for example [Ho13]. In order to reduce the number of test cases, the approach of pairwise testing was selected. The idea behind pairwise testing is that some combinations of the parameters are responsible for faults, and most of the time the combination of two parameters. The construction of the possible test combinations is done via orthogonal arrays, as described in [Ho13]. After the reduction of test cases, the matrices show don't care entries, it was defined that every time an entry consists a don't care the normal slot interval length was chosen. With pairwise testing the combinations could be reduced to 33 test cases. The test sequence for all 33 test cases was as follows: Start synchronization, set the schedule, wait until synchronization is complete, start STADA for five minutes, procced with next test case. For all frames, the transmit time was determined based on the cycle time and logged.

5.4 Result

In all 33 executed test cases, the algorithm detected a faulty scheduled traffic configuration. However, the detailed statement (wrong order, interval length too short or too long) was not clear in most cases. The algorithm principle is responsible for this imprecise recognition. The algorithm uses the transmission time and only knows about the traffic class of the previous frame and the desired traffic class. Tab. 3 you can see some summarized measured values, e.g., mean, minimal, maximum value, from the two test cases. The first test case is the desired configuration, the second test case is the correct order but all intervals are too short. This shows that the mean value in the misconfigured case deviates significantly from the normal case, as does the standard deviation of the network management. These can be also parameters, which one puts into an extension of the algorithm. In the future, it may be possible to obtain a more precise diagnosis by closely scrutinizing multiple cycles, for instance.

Test case	traffic type	Count [µs]	Mean [µs]	Std [µs]	Min [µs]	Max [µs]
TC 1 desired	NM	6494	29	45	0,5	199
	Real-Time	326491	424	0,48	419	438
TC 4 short,123	NM	6646	269	251	0,63	599
	Real-Time	334101	601	0,11	600	604

Tab. 3 : Example Measurements results STADA evaluation

6 Conclusion

In this work possible errors and methods to detect frame preemption faults were investigated. Furthermore, the Scheduled Traffic Anomaly Detection Algorithm (STADA) was evaluated in a test setup. An example schedule with three time slots was chosen and the possible combinations determined. With pairwise testing the test cases could be reduced from 162 to 33. These 33 test cases were executed in the test setup. It could be shown that STADA generally detects a faulty configuration of the scheduled traffic. As of today, the algorithm cannot give a detailed analysis of what exactly is wrong with the schedule. Future work should concentrate on improvement of STADA with the mentioned ideas and the evaluation of the Frame Preemption fault detection ideas.

Literature

[Pr23] PROFINET over TSN, https://www.profibus.com/technology/industrie-40/profinet-over-tsn, 20.10.2023

[Co16] Cost of Data Center Outages, https://www.vertiv.com/globalassets/documents/reports/2016-cost-of-data-center-outages-11-11_51190_1.pdf, 2016

[Ie17] IEEE61158: IEEE Std 61158-2017 (Adoption of EPSG DS 301). IEEE Standard for Industrial Hard Real-Time Communication. IEEE, S.l., 2017.

[GCD15a] Gao, Z.; Cecati, C.; Ding, S. X.: A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches. IEEE Transactions on Industrial Electronics 6/62, pp. 3757–3767, 2015.

[GCD15b] Gao, Z.; Cecati, C.; Ding, S.: A Survey of Fault Diagnosis and Fault-Tolerant Techniques Part II: Fault Diagnosis with Knowledge-Based and Hybrid/Active Approaches. IEEE Transactions on Industrial Electronics, 2015.

[FSJ23] Ferfers, Tobias; S Schriegel, Sebastian; Jasperneite, Juergen: Automated Root Cause Analysis in Time-Sensitive Networks based on Fault Models, International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication ISPCS 2023, London, United Kingdom, 2023

[IB96] Isermann, Rolf; Ballé, Peter: Trends in the Application of Model Based Fault Detection and Diagnosis of Technical Processes, IFAC Proceedings Volumes, Volume 29, Issue 1, Pages 6325-6336, 1996

[EM13] Estima, J. O.; Marques Cardoso, A.J.; A New Algorithm for Real-Time Multiple Open-Circuit Fault Diagnosis in Voltage-Fed PWM Motor Drives by the Reference Current Errors, In IEEE Transactions on Industrial Electronics, vol. 60, no. 8, pp. 3496-3505, Aug. 2013

[Fo02] Fontanini, S. T.; Wainer J.; Bernal V.; Maragon S.: Model based diagnosis in LANs, IEEE Workshop on IP Operations and Management, Dallas, TX, USA, 2002, pp. 121-125

[An18] Anusasamornkul, Tanapat: A Network Root Cause Analysis and Repair System, 2018 6th International Symposium on Computational and Business Intelligence (ISCBI), 2018

[Pr22] Profibus Nutzerorganisation e.V.: Isochronous Mode – Guideline for PROFNET IO, Version 1.3, 2022

[Ie23] IEC/IEEE 60802 TSN Profile for Industrial Automation, https://1.ieee802.org/tsn/iec-ieee-60802/, draft 2.1, 20.10.2023

[Ie16a] IEEE Standard for Local and metropolitan area networks -- Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic, in IEEE Std 802.1Qbv-2015 , no., pp.1-57, 18 March 2016

[Ie16b] IEEE Standard for Local and metropolitan area networks -- Bridges and Bridged Networks -- Amendment 26: Frame Preemption, in IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014) , vol., no., pp.1-52, 30 Aug. 2016

[Ho13] Hoffmann, W. Dirk: Software-Qualität, 2. Auflage, Springer Vieweg, 2013