2ND WORKSHOP

“MACHINE LEARNING & NETWORKING” (MaLeNe) PROCEEDINGS

SEPTEMBER 4, 2023

CO-LOCATED WITH
THE 5TH INTERNATIONAL CONFERENCE ON NETWORKED SYSTEMS (NETSYS 2023)
POTSDAM, GERMANY

Towards Synthesizing Datasets for IEEE 802.1 Time-sensitive Networking

Doğanalp Ergenç*, Nurefşan Sertbaş Bülbül*, Lisa Maile†, Anna Arestova†, Mathias Fischer *
University of Hamburg* University of Erlangen-Nürnberg †
Email: *name.surname@uni-hamburg.de †name.surname@fau.de

***Abstract*—IEEE 802.1 Time-sensitive Networking (TSN) protocols have recently been proposed to replace legacy networking technologies across different mission-critical systems (MCSs). Design, configuration, and maintenance of TSN within MCSs require advanced methods to tackle the highly complex and interconnected nature of those systems. Accordingly, artificial intelligence (AI) and machine learning (ML) models are the most prominent enablers to develop such methods. However, they usually require a significant amount of data for model training, which is not easily accessible. This short paper aims to recapitulate the need for TSN datasets to flourish research on AI/ML-based techniques for TSN systems. Moreover, it analyzes the main requirements and alternative designs to build a TSN platform to synthesize realistic datasets.**

***Index Terms*—IEEE 802.1 TSN, machine learning, dataset**

I. Introduction

Modern mission-critical systems (MCSs) such as avionics and automobiles have evolved from static and close-loop networks of embedded devices to highly interconnected networks of different services. IEEE 802.1 Time-sensitive Networking (TSN) protocols have recently been proposed to satisfy the varying quality of service (QoS) and reliability requirements of such services over standard Ethernet equipment [1]. Replacing multiple domain-specific networking technologies in MCSs, TSN reduces their design and maintenance cost and offers additional configuration flexibility.

This flexibility enables us to (re)configure data streams for dynamically changing data traffic requirements due to the addition or removal of new devices, mobility of existing components, or any anomalies in case of potential failures and security incidents [2]. It requires capturing the complex system behavior to detect such changes and develop advanced reconfiguration strategies that can adapt MCSs in real-time to ensure end-to-end deterministic communication requirements [3].

Artificial intelligence (AI) and machine learning (ML) have been recently employed to model the complex nature of MCSs to enhance their safety, reliability, and efficiency [4]. In TSN-enabled MCSs, they can *autonomously classify different types of TSN traffic* such as video streaming, event- or time-triggered traffic. This classification capability aids in network management and QoS optimization to develop effective self-configuration mechanisms [3]. Moreover, AI/ML enables *prediction of future traffic behavior*, which is crucial for capacity planning, resource allocation, network optimization, and predictive maintenance. This leads to better resource efficiency and QoS in highly dynamic environments by rearranging resources based on estimated traffic patterns [5]. Besides, combined with an effective TSN monitoring tool [6], AI/ML models can accurately *detect anomalies in time-sensitive data traffic*, which could indicate network intrusions, security breaches, or performance issues.

However, AI/ML models usually require training with significant amounts of data that should accurately reflect network topology and communication. Since IEEE 802.1 TSN protocols have yet to be broadly deployed, it is challenging to find actual data. Besides, the lack of transparency in MCSs due to their safety and security obligations prevents even (potentially) existing data from being publicly available. Therefore, we need reliable sources and platforms to obtain extensive and realistic TSN datasets. Accordingly, in this short paper, our first goal was to recapitulate the need for TSN datasets to flourish research on AI/ML-based TSN design, configuration, and resilience methods. In the remainder, we explore the main requirements of an open and reusable platform to synthesize public TSN datasets (Section II). Then, we shortly review alternative designs to build such a platform (Section III).

II. Platform Requirements for Dataset Synthesis

The two primary reasons for the absence of public TSN datasets are the lack of widely-deployed TSN systems and the opaque nature of MCSs. Designing an accessible TSN platform, e.g., a TSN-based prototype, simulator, etc., should be the first step to synthesizing the required datasets, which should represent the overall behavior of the respective system in a reliable and verifiable manner. Accordingly, we outline the requirements of such a platform as follows.

- **Representation of a realistic MCS:** A TSN-based platform (and its respective dataset) should accurately reflect the design and operational principles of actual MCSs such as aircraft, automobiles, or industrial systems. This includes modeling a realistic network topology and different service classes with distinct QoS and reliability requirements.
- **Support for various TSN protocols:** This platform should support a broad set of TSN protocols to generate extensive datasets reflecting various MCS scenarios. For instance, P802.1Qav CBS or P802.1Qbv TAS could be alternatively used for scheduling mixed-criticality streams. P802.1CB FRER is required for redundant communication, and P802.1Qcc SRP could also be a prerequisite for the network-wide configuration of these protocols.

- **Verification of system behavior:** The design and configuration of the TSN platform should be verified to ensure a reliable dataset reflecting the desired network behavior. For instance, verifying the TAS scheduler guarantees that all streams in the synthesized dataset are realistically forwarded within their latency boundaries [7]. This requires an extensive analysis of the resulting dataset.
- **Visibility of individual components:** The platform should allow extracting local traffic from selected components alongside an extensive dataset with system-wide network communication. This enables the analysis of the behavior in target components more isolatedly.
- **Scalability:** A realistic topology size and connectivity of MCSs should also be considered for the platform design since they shape the interdependencies between TSN components. This mainly affects both amount and depth of a TSN dataset extracted from the respective platform.

III. Platform Design

We consider three potential designs for a TSN platform to synthesize datasets: a) a hardware-based prototype, b) a hybrid emulation, and c) a simulation platform. This section briefly analyzes them regarding the requirements mentioned earlier. Table I also summarizes this discussion.

TABLE I
Comparison of different platform designs.

	Hardware	**Hybrid**	**Simulation**
Representation	✓	✓	∼
Protocol Support	∼	∼	✓
Verification	✓	✓	✓
Visibility	∼	✓	✓
Scalability	✗	∼	✓

a) Hardware-based Prototype: A platform of actual off-the-shelf equipment, e.g., TSN bridges or more generic TSN-supporting Linux devices, represents an MCS the most realistically [8]. While generic equipment provides extensive visibility, TSN bridges require monitoring capabilities such as mirror ports, which may only exist in some commercial TSN bridges. Proportional to their visibility, the behavior of hardware-based components can be verified by manually accessing these components or processing the dataset derived from the platform. Unfortunately, such a platform potentially has a *partial* protocol support (marked as "∼" in Table I) since existing TSN equipment implements a limited set of protocols as it takes a significant engineering effort. Besides, it can only have a limited scalability that may not reflect the typical size and complexity of real MCSs [9].

b) Hybrid Emulation: When hardware capabilities and platform scalability are limited, it is possible to emulate certain parts of a platform by integrating software-based solutions, e.g., an emulator, into a basis hardware-based prototype. Mininet, for instance, could represent a network of generic Linux-based devices with TSN scheduling capabilities [10]. While this is partially scalable and provides more visibility and easier verification, integrating hardware- and software-based solutions is technically challenging and requires the synchronization of different environments.

c) Simulation: Simulation platforms have already been used in several TSN studies [3], [5], and it is the most flexible alternative regarding configurability, scalability, and visibility. They offer a wide range of TSN protocols with a convenient level of abstraction [11]. Besides, integrating (even run-time) verification mechanisms into the simulations is more straightforward. However, they can only approximate the actual dynamics of MCSs and potentially deviate from real-world network behavior.

IV. Conclusion

AI/ML models can significantly foster the development of advanced design, configuration, and maintenance techniques for emerging IEEE 802.1 TSN protocols. While these models typically require a significant amount of data for training, there does not exist any public TSN dataset due to the lack of broadly-deployed TSN systems. In this paper, we first recapitulate the need for an extensive TSN dataset to enable AI/ML research providing a standardized and reproducible basis for experimentation and validation. However, this first requires a realistic TSN platform that synthesizes and validates TSN datasets. Therefore, we next outline five major requirements and analyze alternative designs of such a platform. We aim to use this preliminary study as a road map for our ongoing effort to create extensive, realistic, and public TSN datasets.

References

[1] "Time-Sensitive Networking (TSN) Task Group." Available at https://1.ieee802.org/tsn/.

[2] L. Maile, K.-S. J. Hielscher, and R. German, "Delay-Guaranteeing Admission Control for Time-Sensitive Networking Using the Credit-Based Shaper," *IEEE Open Journal of the Comm. Soc.*, vol. 3, 2022.

[3] N. Sertbaş Bülbül, D. Ergenç, and M. Fischer, "SDN-based Self-Configuration for Time-Sensitive IoT Networks," *International Conference on Local Computer Networks (LCN)*, 2021.

[4] P. Laplante, D. Milojicic, S. Serebryakov, and D. Bennett, "Artificial Intelligence and Critical Systems: From Hype to Reality," *Computer*, vol. 53, no. 11, pp. 45–52, 2020.

[5] N. Sertbaş Bülbül, D. Ergenç, and M. Fischer, "Towards SDN-based Dynamic Path Reconfiguration for Time-sensitive Networking," *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2022.

[6] D. Ergenç, R. Schenderlein, and M. Fischer, "TSNZeek: An Open-Source IDS for IEEE 802.1 TSN," in *IFIP Networking, International Workshop on Time-Sensitive and Deterministic Networking (TENSOR)*, 2023.

[7] A. Arestova, K.-S. J. Hielscher, and R. German, "Design of a hybrid genetic algorithm for time-sensitive networking," in *20th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems*, 2020.

[8] M. Bosk, F. Rezabek, K. Holzinger, A. G. Marino, A. A. Kane, F. Fons, J. Ott, and G. Carle, "Methodology and infrastructure for tsn-based reproducible network experiments," *IEEE Access*, vol. 10, 2022.

[9] D. Ergenç, C. Brülhart, and M. Fischer, "Towards Developing Resilient and Service-oriented Mission-critical Systems," *9th IEEE International Conference on Network Softwarization (NetSoft)*, 2023.

[10] M. Ulbricht, S. Senk, H. K. Nazari, H.-H. Liu, M. Reisslein, G. T. Nguyen, and F. H. P. Fitzek, "TSN-FlexTest: Flexible TSN Measurement Testbed (Extended Version)," 2022. Preprint on arXiv:2211.10413.

[11] T. Steinbach, H. D. Kenfack, F. Korf, and T. C. Schmidt, "An Extension of the OMNeT++ INET Framework for Simulating Real-Time Ethernet with High Accuracy," in *4th International ICST Conference on Sim. Tools and Techniques*, 2011.