Enhancing Security in Time-Sensitive Networks: Simulation and Evaluation of PSFP

Peter Danielis, Willi Brekenfelder, Helge Parzyjegla, Florian Bayer and Gero Mühl
Institute of Computer Science, University of Rostock, Rostock, Germany
Email: {peter.danielis;willi.brekenfelder;helge.parzyjegla;gero.muehl}@uni-rostock.de

***Abstract*—Time-Sensitive Networking (TSN) standards provide Ethernet with real-time capabilities crucial for applications like industrial automation and autonomous systems. Among these, Per-Stream Filtering and Policing (PSFP) is designed to secure networks by mitigating unauthorized or malicious traffic.**

This paper outlines the security capabilities of PSFP, focusing on its ability to counter attacks such as frame injection. Using OMNeT++ simulations, we validate PSFP's effectiveness in filtering injected frames based on stream parameters, demonstrating its relevance for robust TSN deployment in critical environments.

***Index Terms*—Time-Sensitive Networking, Per-Stream Filtering and Policing, OMNeT++.**

I. Introduction

The demand for deterministic and reliable communication in industrial applications has grown with the rise of Industry 4.0. Modern smart factories and autonomous systems require seamless integration of real-time and traditional Ethernet traffic, making Time-Sensitive Networking (TSN) an essential technology. TSN extends Ethernet with capabilities such as deterministic latency, time synchronization, and traffic prioritization [1], [2].

Among the TSN standards, Per-Stream Filtering and Policing (PSFP) is pivotal for enhancing network security by mitigating threats like Denial of Service (DoS) attacks and unauthorized traffic. Operating at the ingress of switches, PSFP employs a three-stage filtering mechanism: verifying stream properties, checking timing constraints, and regulating data rates [3]. These features make PSFP indispensable for critical systems requiring both real-time communication and robust security.

Previous studies have explored various aspects of TSN security. In [4], for example, vulnerabilities in TSN standards were highlighted, with the manipulation of PSFP rule sets being emphasized as the main risk. [4] emphasized the need for application-specific security configurations but lacked a detailed methodology for PSFP deployment. Other works have analyzed the impact of attacks such as DoS and Man-in-the-Middle (MitM) on TSN scheduling using automata-based approaches [5]. In contrast, our research focuses on a simulation-based evaluation of PSFP's filtering and policing capabilities.

In this paper, we outline the security capabilities of PSFP to protect TSN networks from attacks and misconfigurations through OMNeT++ simulations.

The remainder of this paper is organized as follows: Section II outlines how to configure PSFP and introduces the simulation model. Section III presents the results of our case study. Finally, Section IV concludes the paper with directions for future work.

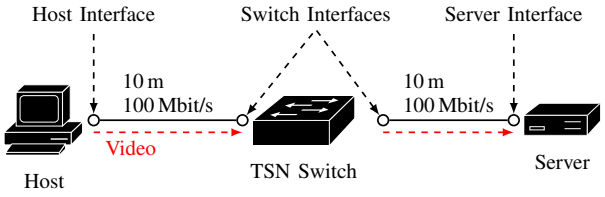

Fig. 1. TSN network with host (talker), TSN switch, and server (listener)

II. PSFP Configuration and Simulation Model

This section outlines how to configure PSFP to protect from attacks as well as malfunctions and integrating it into a simulation model. It also includes an explanation of the simulation setup, complemented by visual representations of the network layout and PSFP structure.

A. Network Planning and General Simulation Setup

The deployment of PSFP requires careful planning of network parameters, such as cable lengths, transmission rates, stream priorities, and frame size limitations. These aspects form the basis for simulations conducted using OMNeT++ (v6.0.1) with the OMNET++ framework INET (v4.5.0). Figure 1 illustrates a basic TSN network comprising a host (talker), a TSN switch, and a server (listener), with 10-meter cable lengths and a maximum transmission rate of 100 Mbps.

The simulation leverages the `Ieee8021qFilter` module, which effectively implements the PSFP standard. The modular architecture of the `Ieee8021qFilter`, including its adapted submodules, is shown in Figure 2. To enhance clarity, the figure is divided into two parts. It illustrates the three filter stages—*stream filter*, *stream gate*, and *flow meter*—each represented by multiple submodules. The first stage, `stream filter`, includes the `classifier` and `streamFilter` modules. The `classifier` directs incoming frames to the correct path according to a configurable mapping of streams to paths.

In order to assign frames to specific streams, a decoding module is required, which is located elsewhere in the switch. This module is not part of the PSFP standard but is crucial for functionality. It uses the source address, destination address, VLAN ID, Priority Code Point (PCP) value, and the incoming interface to perform the decoding. The `streamFilter`

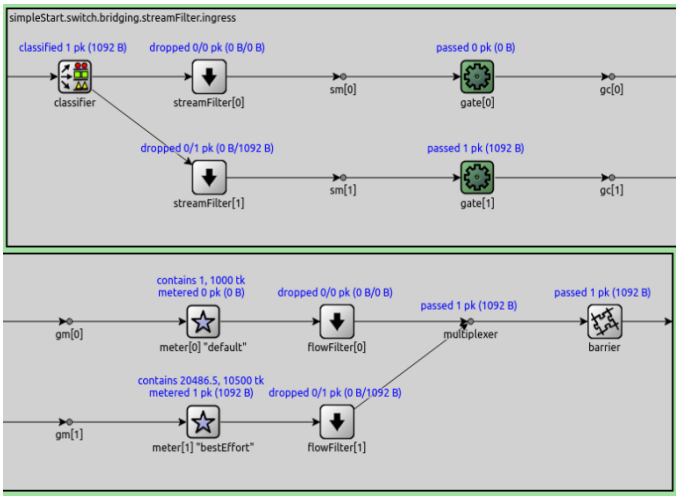

Fig. 2. Structure of PSFP in OMNeT++/INET

module then filters frames based on the maximum Service Data Unit (SDU).

The `stream gate` (labeled as `gate` in Figure 2) is built on the `PeriodicGate` module. The final stage of filtering involves the `meter` and `flowFilter` modules. The `meter` module assigns tokens to frames according to the token bucket parameters, while the `flowFilter` module determines which frames are discarded.

The `barrier` module is not relevant for PSFP's operation within INET and, as a result, is not further considered.

B. Mapping PSFP Parameters to Attack Scenarios

PSFP parameters, such as frame size constraints, stream priorities, and token bucket configurations, are mapped to specific attack scenarios. For example, unauthorized frame injection can be countered by strict stream identification in the classifier module, while excessive traffic from malicious sources is managed by the flow meter's token bucket mechanism. By simulating such scenarios, PSFP's ability to secure TSN environments against attacks as well as misconfigurations and ensure reliable traffic management is validated.

C. Simulation Model

The simulation model uses a simplified TSN setup with a talker, a switch, and a server. Streams are configured with realistic parameters, including maximum SDU sizes, periodic gate timings, and token bucket parameters, such as Committed Information Rate (CIR), Excess Information Rate (EIR), Excess Burst Size (EBS), and Committed Burst Size (CBS), for traffic shaping. Frames are monitored as they pass through the PSFP filter stages described in Figure 2.

In addition to handling normal traffic, the simulation replicates attack scenarios by injecting unauthorized or oversized frames. For example, the stream gate blocks frames attempting to pass through closed paths, while the flow meter prevents bandwidth overuse by throttling excessive traffic.

TABLE I
OVERVIEW OF STREAMS FOR HOST TO SERVER COMMUNICATION

Stream	Best effort	Video	Control
Talker	Host	Host	Host
Bridge	Switch	Switch	Switch
Listener	Server	Server	Server
Priority	0	4	7
Data rate	42 Mbps	30 Mbps	2 Mbps
Trans. time	unknown	unknown	500 µs
Dest. port	1001	1001	1001
VLAN ID	1	1	1
Frame size	unknown	300-500 B	130 B
Protocol	UDP	UDP	UDP

III. EVALUATION

For the evaluation, a case study incorperating an injection attack with three stream classes is conducted in OMNeT++ and its INET framework.

A. Simulation Setup

The simulation scenario examines communication between a host (talker) and a server (listener). The network is illustrated in Figure 1. The host sends a stream to the server, considering three different streams with distinct characteristics. These streams include a best-effort stream with varying data rates, a video stream with relatively stable data rates, and a control stream with fixed transmission times. The specific stream characteristics are listed in Table I. Each stream is examined separately in a distinct simulation, allowing the configured PSFP standard to be evaluated in response. Since the injection attack can affect all stream parameters, it is addressed in the case study. The paragraph describing the best-effort stream focuses on flow control using the flow meter. Additionally, part of the stream filter functionality is examined. The video stream is also protected by the stream filter. The control stream can be managed using the gate control. All scenarios have a simulation duration of 1 second. A secondary application on the host is implemented to simulate injected packets, sending additional frames starting at 0.5 seconds. These frames are precisely timed to be sent between the healthy frames, requiring fixed transmission timestamps for the healthy stream.

B. Results

Best-Effort Stream: The best-effort stream is characterized in Table I. The attacker stream consists of frames 600 bytes in length, injected between the original frames, with a data rate of 24 Mbps. PSFP parameters are configured assuming that neither transmission timestamps nor frame sizes are known. Thus, filtering based on frame size or timestamps is not feasible. Instead, the stream can only be controlled via the stream handle value and data rate. A second stream representing the attacker is injected, with the frame size set to 600 bytes. The injected frames have identical stream properties to the original stream.

To regulate the data rate, the token bucket values are configured in the flow meter. The best-effort stream has an expected data rate of 42 Mbps, so the CIR is set accordingly. The bucket size is configured to 21 KB, allowing for a temporary 10% increase in the data rate for up to 5 ms. The

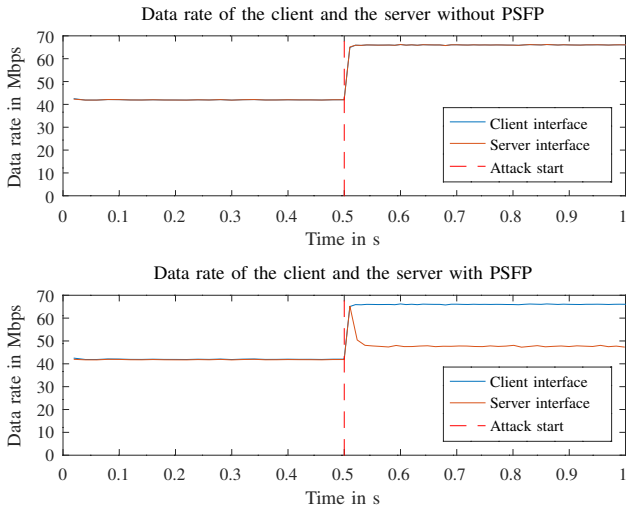

Fig. 3. Impact of PSFP on the data rate of injected frames in the best-effort stream

EIR is set to 10% of the CIR value, with the EBS being half of the CBS. From 0.5 seconds onwards, the data rate of the Best-Effort Stream is expected to increase due to the injected frames. However, PSFP is expected to throttle the data rate, stabilizing it at CIR + EIR (46.2 Mbps).

Figure 3 depicts the data rates at the client and server interfaces. The top graph shows the data rate without PSFP, while the bottom graph illustrates the rate with PSFP enabled. In the top graph, data rates overlap at both interfaces, as all packets are forwarded by the switch without exception, and the expected throttling to 46.2 Mbps does not occur. In the bottom graph, the server interface data rate deviates from the client data rate after the attack begins. Initially, a peak is observed at the server interface, after which the rate stabilizes around 48 Mbps. The flow meter initially attempts to sustain the increased data rate by using all available tokens. Once all tokens are exhausted, additional frames are marked and discarded. This behavior results in a peak in the graph, with new tokens being added to the buckets at CIR and EIR rates. Since incoming frames continue to exceed the token availability, the data rate eventually stabilizes at 48 Mbps, slightly higher than the expected 46.2 Mbps due to the omission of the Ethernet header in data rate calculations. It is important to note that while limiting the data rate helps against clogging the entire link, it does not help against a DoS that aims to limit the connection of the sender of the original data. Because the interfering traffic cannot be distinguished from the original traffic, correct packets are also discarded. This leads to an impairment of the intended connection.

Video Stream: The characteristics of the video stream are shown in Table I. Unlike the best-effort stream, the frame size is known and considered in PSFP configuration. The stream filter is set to a maximum allowable frame size of 500 bytes. In OMNeT++, filtering is applied to the entire frame length instead of the standard's SDU specification. As expected, injected frames, with a length of 600 bytes, are discarded due to size violations.

Control Stream: The same network is now considered with the control stream. This stream is characterized by the fact that, in addition to the data rate and frame size, the exact transmission time is also known. The relevant parameters are summarized in Table I. This allows the PSFP configuration of the switch to account for the specific timing of the stream. In this case, the gates are configured to be open during the transmission window and closed during the remaining time. This ensures that frames transmitted at incorrect times are prevented from reaching the server.

The injected stream is configured with a frame size of 100 bytes and a packet length of 70 bytes. The frame size is smaller than that of the control stream, which prevents the frames from being filtered out by the stream filter based solely on their frame size. Since the gates are closed when the frames of the injected stream arrive, these frames are dropped as expected.

IV. Conclusion

TSN extends Ethernet for real-time applications, with the PSFP standard offering protection against attacks and malfunctions. This paper outlines the security capabilities of PSFP within real-time Ethernet networks, using its three-stage filtering mechanism: the stream filter (which handles size and stream parameters), the stream gate (managing arrival times), and the flow meter (regulating data rate). Simulations conducted in OMNeT++ demonstrate that PSFP can safeguard TSN networks from injection attacks by filtering frames according to predefined stream characteristics.

Future work should focus on completing the PSFP implementation in OMNeT++, validating the findings through hardware-based setups, and investigating automated parameter configuration to further improve network security.

References

[1] TSN Task Group, "IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks," *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)*, pp. 1–1993, 2018.

[2] J. L. Messenger, "Time-sensitive networking: An introduction," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 29–33, 2018.

[3] TSN Task Group, "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing," *IEEE Std 802.1Qci-2017*, pp. 1–65, 2017.

[4] F. Fischer and D. Merli, "Security Considerations for IEEE 802.1 Time-Sensitive Networking," *ICECCME*, pp. 1–7, 2022.

[5] H. Wang *et al.*, "A Vulnerability Mining Method for IEEE802.1Qbv in TSN Systems," *China Automation Congress*, pp. 6644–6649, 2022.