

An east-westbound control architecture for multi-segment deterministic networking

Jakob Miserez , Didier Colle , Mario Pickavet , Wouter Tavernier 
{jakob.miserez,didier.colle,mario.pickavet,wouter.tavernier}@ugent.be

Abstract—Time-Sensitive Networking (TSN) and Deterministic Networking (DetNet) provide standardized solutions for reliable real-time communication over respectively L2 and L3 networks. Interconnecting heterogeneous TSN segments and L3 DetNet segments is one of the main open challenges to enable wide area DetNet applications. It requires mapping diverse latency guarantee models, as well as adequate interaction between the control systems of different segments. Current state-of-the-art solutions typically focus on a single network segment, or they don't consider the complex interworking between heterogeneous deterministic network segments. This paper proposes an architecture for routing and signaling end-to-end traffic flows in heterogeneous multi-segment deterministic networks. The proposed East-Westbound interaction architecture between the control systems of DetNet network segments enables a divide-and-conquer strategy that significantly reduces the inherent complexity of provisioning end-to-end routes in these networks. The potential of the architecture is validated through a proof-of-concept in an emulated multi-segment network. Notably, interactions contained within a segment retain constant overhead, while the overhead of actions spanning multiple segments tends to increase when the number of involved segments rises.

Index Terms—Time-Sensitive Networking (TSN), Deterministic Networking (DetNet), routing, Software-Defined Networking (SDN), multi-segment deterministic networks, east-westbound

I. INTRODUCTION

Driven by the industry demand for reliable real-time communication, the IEEE802.1 Time-Sensitive Networking (TSN) standards emerged to provide Ethernet-based solutions for deterministic L2 LANs [1]. TSN provides reliable data transmission through mechanisms such as Time-Aware Shaping (TAS) and Cyclic Queueing and Forwarding (CQF) [2]. Relying heavily on time synchronization, TAS and CQF are traffic shapers in support of achieving low-latency, low-jitter, and zero packet loss for periodic traffic flows. The goal of the IETF Deterministic Networking (DetNet) Working Group is to extend the TSN standards to larger-scale L3 networks. One of the main challenges of DetNet is the integration between TSN network segments and L3 network segments [3] [4]. This is necessary for real-time L3 applications (using e.g., RTP [5]), and to stitch TSN islands together. Example applications for such scenarios include professional audio and video (ProAV), industrial machine-to-machine (IM2M), and electrical utilities [3]. A specific industrial use case for interconnected TSN islands is given in [6]. While both TSN and DetNet share the same goal of providing deterministic transmission, there are significant differences between them.

For example, DetNet networks are not bound to Ethernet, and they typically are deployed in wide area environments involving multiple L2 and L3 network segments with potentially a high number of routers/switches. Different network segments likely involve different data plane and control technologies, such that achieving high time synchronization accuracy is hard [4]. This limitation makes TAS and CQF infeasible in DetNet. Therefore, several new packet scheduling techniques have been proposed within DetNet, such as tagged Cyclic Queueing and Forwarding (tCQF) [7]. These disparities pose significant challenges. Firstly, the heterogeneity of technologies across different segments requires mapping diverse latency guarantee models, complicating the provisioning of end-to-end routes. Moreover, different control mechanisms and protocols add additional complexities, requiring sophisticated coordination and translation mechanisms. Lastly, the unavailability of end-to-end time synchronization across segments and the wide range of involved delays, stemming from e.g., longer links and varying transmission mediums, introduce additional challenges in ensuring determinism across the network. While the interworking between TSN and DetNet is an integral part of the DetNet architecture [8], no control plane solutions are proposed as of yet at IETF. This paper proposes a solution for routing traffic flows over heterogeneous multi-segment deterministic networks to address these challenges. Here, every network segment (e.g., a TAS-based TSN) is managed by a Centralized Network Controller (CNC). An East-Westbound (EW) architecture enables a divide-and-conquer approach that significantly reduces the end-to-end routing complexity by hiding internal details. This also enables plug-and-play between various technologies, as opposed to a complex one-fits-all solution. While focusing on the combination of L3 DetNet segments using strict priority queueing (SPQ) and TAS-based TSN segments, the solution can be extended to other combinations as well. This paper focuses on the control aspects, but also provides a short overview on network calculus (NC) principles for interconnecting traffic flows.

II. RELATED WORK

A. Single-domain/segment solutions

Within TSN, TAS and CQF are the main drivers for deterministic transmission of periodic traffic. Relying heavily on time synchronization, TAS and CQF operate on a fixed traffic schedule in cycles, therefore eliminating non-determinism. Gate Control Lists (GCLs) determine which queues are open for transmission at which time during a cycle. Plenty of GCL

synthesis algorithms exist for computing GCLs in a single-domain TSN network [9]. The routing solutions in [10], [11] exploit SPQ to bound queueing delays using NC. They do not require time synchronization and can be deployed in a wide-area network. However, it is hard to provide jitter guarantees as the experienced queueing delays can vary greatly depending on other traffic in the network, as opposed to TAS and CQF.

B. Multi-domain/segment solutions

In [12], a distributed SDN control plane was proposed for heterogeneous time-sensitive networks, but it left out routing and configuring multi-domain flows. An idea for a CNC to CNC Protocol based on the Link Registration Protocol (LRP) was proposed in [13]. A controller for mixed wired and wireless TSN networks was introduced in [14], where domains refer to wireless and wired domains that are both handled by the controller. Domain Management Functions (DMFs) for domain discovery and multi-domain stream configuration were proposed in [15], but no details on the communication between DMFs were given. The work in [16] presents a GCL synthesis algorithm for TSN networks with heterogeneous end stations (i.e., without TSN capabilities). A hierarchical SDN framework for TSN that enables transmission across non-TSN domains is given in [17]. A path selection method with QoS support in multi-domain and hierarchical SDN networks was presented in [18]. The authors of [19] propose a hierarchical approach for provisioning end-to-end routes over a multi-domain TSN network, where a top-level controller communicates with all domain controllers to build a full view of the network. Using this full view, the top-level controller employs a global GCL synthesis algorithm that computes GCLs in all domains. Although it exploits parallel signaling, the top-level controller is a single point of failure, and the complexity increases per added domain, resulting in high runtimes. An EW protocol for dynamic inter-domain stream configuration in multi-domain TAS-based TSN networks was proposed in [20]. This EW protocol allows communication between neighboring domain controllers, resulting in a peer-to-peer architecture. The domains exchange abstracted views of their local topology to build an abstracted multi-domain topology, which is used to determine a multi-domain path. The abstracted views only contain reachability information of neighboring domains and end stations. As the end-to-end stream configuration is distributed over each domain, every involved domain configures an internal stream such that the end-to-end delay needs to be divided among the domains. A simple heuristic approach is used, splitting the maximum end-to-end delay equally among the remaining domains. This work focuses on the mechanisms and interfaces to setup inter-domain streams, but does not consider multi-domain routing algorithms or advanced delay splitting strategies.

C. Proposed solution

This paper proposes an SDN-based solution focusing on routing traffic flows in heterogeneous multi-segment determin-

istic networks, building on topology abstraction and end-to-end delay budget division. It has the following features:

- Local topology discovery using the Link-Layer Discovery Protocol (LLDP) [21].
- A topology abstraction technique for representing complex and heterogeneous deterministic network segments.
- An EW manager for multi-segment topology discovery and coordination of end-to-end traffic flows across multiple segments.
- A multi-segment routing algorithm that takes network capabilities into account.
- A diameter-based delay budget division technique for efficient and flexible intra-segment routing.

Note that this paper considers multi-segment deterministic networks, and not multi-domain networks, meaning that all network segments are assumed to be under the same administrative control. Therefore, specific privacy or security constraints regarding this are not considered. The remainder of this paper is organized as follows. Sec. III provides an in-depth overview of the control plane mechanisms. Sec. IV gives a brief overview of NC principles for TSN and DetNet interworking. Then, Sec. V discusses the experimental setup and results. Finally, this paper is concluded in Sec. VI.

III. CONTROL PLANE MECHANISMS

This section describes the control plane mechanisms for configuring network nodes, discovering the local topology, neighboring segments and the multi-segment topology, and the provisioning and signaling of end-to-end routes. In the proposed architecture, every CNC is responsible for a specific network segment, e.g., a TAS-based TSN segment. At the core of every CNC lies the EW Manager, which is responsible for communicating with neighboring CNCs to discover the multi-segment topology and to provision end-to-end routes. A high-level overview of the control plane mechanisms is depicted in Fig. 1. The following high-level interactions (shown as labeled arrows), which will be detailed later on, are present:

- 1) Node (agent) and CNC communication. This includes the configuration of the node by the CNC and the node providing event notifications, such as neighbor changes.
- 2) Local topology discovery (interaction 2a) and neighbor discovery (interaction 2b) using LLDP.
- 3) Propagation of neighbor changes coming from the network nodes to update the local topology (interaction 3a). A change to the local topology is also notified to the EW manager (interaction 3b).
- 4) Neighboring EW managers communicate with each other to obtain information about other segments and to coordinate provisioning end-to-end flows.
- 5) The EW manager can start the routing/scheduling of an internal traffic flow (interaction 5a). Once a new path/schedule is computed, network nodes are configured via the southbound interface (SBI) (interaction 5b).

The following subsections provide detailed information about the various mechanisms.

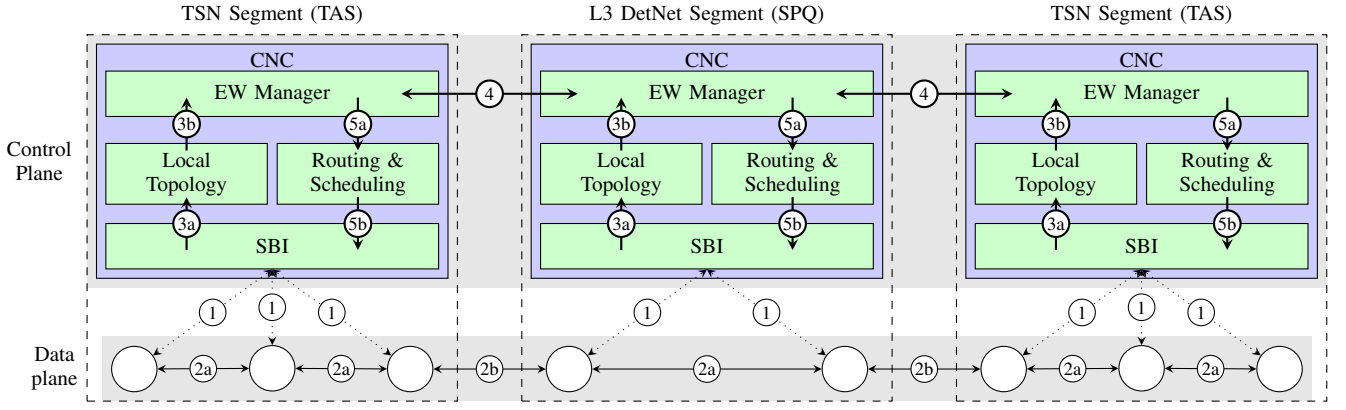


Fig. 1: Control plane mechanisms and interactions for multi-segment heterogeneous deterministic networks

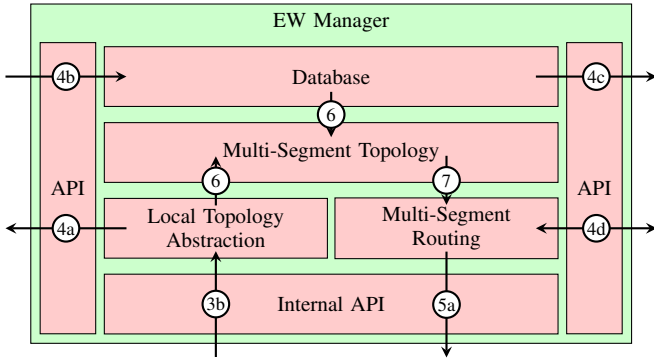


Fig. 2: East-Westbound manager interactions

A. Node discovery and configuration

Every node (including end stations) runs an agent that handles the communication with the CNC. Every CNC is preconfigured with a unique ID, and upon a new node connection, the CNC provides the node with IP and LLDP configuration, and assigns a globally unique agent ID to the node by concatenating the CNC ID with a locally unique ID. Locally unique IDs are managed by the CNC, and they can be implemented using a simple counter.

B. LLDP-based local topology discovery

LLDP [21] is an Ethernet-based protocol to detect neighboring devices, such as routers, switches, and end stations. Assuming all network nodes (including end stations) run LLDP, every agent can detect neighboring devices from both the same segment and from neighboring segments by incorporating the agent ID within the LLDP update frames. Since the CNC ID is incorporated into the global agent ID, neighbors from other segments can be detected easily. Neighbor updates are communicated to the CNC.

C. Intra-segment routing & scheduling

The proposed solution is agnostic to the routing/scheduling of a flow within the same segment. For TAS-based TSN networks, any existing solution can be used [9]. For L3 DetNet

segments employing SPQ, approaches based on NC can be used [10] [11]. However, a mapping between the traffic models used by the algorithms is required (cf. Sec. IV).

D. East-westbound (EW) manager

The main goal of the EW manager is to coordinate the provisioning of end-to-end paths spanning multiple segments by breaking down the end-to-end flow into multiple internal flows, one per involved segment. The key idea here is that the multi-segment route is computed using an abstracted multi-segment topology, which hides complex internal details such as internal links, GCL configurations, reserved resources at every link, etc. Then, the provisioning of each internal flow can be offloaded to the particular segment. The result is a 'glued' end-to-end path over multiple segments. The different components and detailed interactions (labeled arrows) of the EW manager are depicted in Fig. 2. Notice the similarities with the high-level interactions in Fig. 1.

1) *Topology abstractions*: Topology abstractions are the building blocks for constructing an abstracted multi-segment topology. Importantly, they hide internal details as much as possible, but provide enough information to construct a suitable multi-segment path. The EW manager therefore maintains an abstracted view of the local topology. Changes to the local topology, such as new neighbor from another segment, are notified to the EW manager, which updates the abstraction accordingly (interaction 3b). Topology abstractions in the current solution contain the following information:

- The ID of the network segment.
- Reachable end stations within this network segment.
- The IDs of neighboring segments and border router/switch information.
- Network segment diameter (using e.g., Floyd Warshall).
- Network segment capabilities (cf. next subsection).

It could be that an update to the local topology does not affect the abstracted representation (internal detail). However, when it does (e.g., when a new neighboring segment is discovered), the updated abstraction is distributed to neighboring EW managers (interaction 4a). Every EW manager maintains a database of the latest topology abstractions that it received

from other CNCs. When receiving an updated abstraction (interaction 4b), the EW manager forwards this abstraction to its other neighbors (interaction 4c), such that every EW manager's database will eventually be up-to-date. The database containing all received abstractions and the local topology abstraction suffice to build the multi-segment topology (interaction 6). An example of a multi-segment topology based on topology abstractions is shown in Fig. 3. Note that the abstracted multi-segment topology hides internal links, but maintains reachability information about the end stations. Border information is included such that the start and end points of an internal flow can be decided.

2) *Multi-segment routing*: Upon receiving a flow request spanning multiple segments, the multi-segment routing component performs two steps. First, it computes a multi-segment path for the multi-segment flow. The result is a list of 'glued' internal flows, one per involved segment. Then, a delay budget division technique assigns a portion of the end-to-end delay requirement to each internal flow. Both of these steps allow to provision an end-to-end path that spans multiple segments and that respects the total end-to-end delay requirements, assuming that each internal flow respects its given delay budget.

This paper proposes a simple multi-segment routing algorithm based on Dijkstra's shortest-path algorithm, meaning that the resulting multi-segment path utilizes a minimal number of segments. This algorithm takes the multi-segment topology as input (interaction 7). However, network segments can differ greatly. For example, a TAS-based TSN segment can provide bounded delay and jitter within its segment, similar to an L3 DetNet segment using e.g., tCQF. However, when using SPQ, this might be impossible. If this is not taken into account, a traffic flow with strong jitter requirements could be forced to be routed over a segment that can never meet these requirements, introducing costly additional signaling and rerouting. To counter this, the basic algorithm is extended to consider the capabilities of the segments. These capabilities, which are present in the topology abstractions (and therefore in the multi-segment topology), refer to which type of traffic these network segments can handle (e.g., periodic, asynchronous) and what guarantees they can give. For example, if a flow needs to be established with strong jitter guarantees, then network segments that cannot provide this are first removed from the multi-segment topology. This ensures that the resulting multi-segment path can meet all requirements.

Given the possible differences between the network segments both in number as hops and geographical scale, the end-to-end delay needs to be divided carefully among the internal flows such that each segment can provide an internal route and simultaneously has enough flexibility to do this when resources are scarce. Therefore, it makes sense to take network segment characteristics into account. A split based on network diameter is proposed here. In this case, network segments with a larger diameter receive a proportionally higher delay budget compared to segments with a lower diameter. The rationale behind this method is that in bigger segments, the internal flow will typically cross a larger number of hops.

This means that more queueing delay can be accumulated in those networks, and therefore they need a larger budget. As an example, consider the network in Fig. 3 and a flow between the end stations of S1 and S4 with end-to-end delay requirement of 24ms. Say the multi-domain routing algorithm has decided to route this flow over S1, S5, and S4, with diameters of respectively 2, 1, and 2. To account for single-node segments with a diameter of 0, the network diameters are increased with 1. The sum of these adjusted diameters is now 8. S1 has an adjusted diameter of 3, and contributes $\frac{3}{8}$ to this sum. Therefore, this domain receives a delay budget of $\frac{3}{8} \times 24\text{ms} = 9\text{ms}$. Similarly, S5 and S4 receive a delay budget of respectively 6ms and 9ms.

3) *Path provisioning and signaling*: Once the multi-segment routing component has determined the internal flows for a multi-segment flow, the internal flows need to be provisioned at every involved segment. A split multi-segment flow request contains both the traffic model and internal flows. First, the source CNC initiates the local routing/scheduling algorithm for its own internal flow (interaction 5a). If a suitable route is found, the EW manager proceeds to translate the traffic model parameters for the subsequent segment (cf. Sec. IV). Then, the split multi-segment flow request travels along the involved CNCs through the EW managers (interaction 4d). Upon reception of the request (interaction 4d), the local routing/scheduling algorithm is invoked again (interaction 5a), after which the traffic model is again translated and the request is forwarded. This process continues until the destination segment is reached. Success is signaled back from the destination segment to the source segment, confirming the establishment of the requested end-to-end flow. However, in case of intermediate failures, due to e.g., insufficient resources within a segment, a failure signal is signaled back to the previous segment until the source segment is reached. Upon receiving a failure signal, all previously allocated resources associated with the failed route are freed.

IV. TSN AND DETNET INTERWORKING

While the proposed solution takes advantage of a divide-and-conquer approach, the routing and scheduling algorithms (based on their delay guarantee model) can differ greatly per segment. An important aspect here is the traffic flow model. An L3 solution based on SPQ and NC such as [10] or [11] models traffic flows by an arrival curve, which differs from the periodic traffic flows in TAS-based TSN networks. Therefore, when routing traffic coming from the L3 segment to the TAS-based TSN network and vice versa, traffic models must be translated from one segment to the other in order to be routable within each segment. The NC models in [22] prove the co-existence of SPQ and TAS in TSN by deriving service curves for TAS queues. This can serve as foundation for integrating asynchronous traffic coming from the L3 segment into a TAS-based TSN. Furthermore, arrival curves have been derived for periodic traffic flows [23], which can be used to route flows coming from a TSN segment through the L3 DetNet segment.

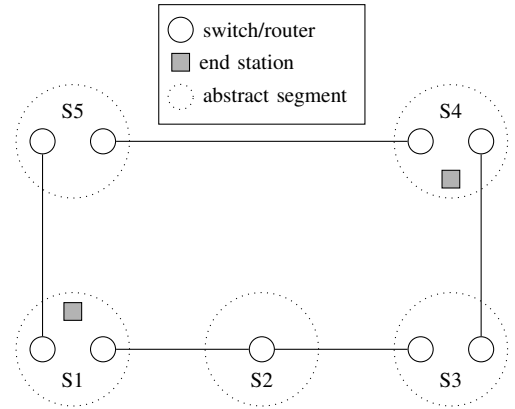
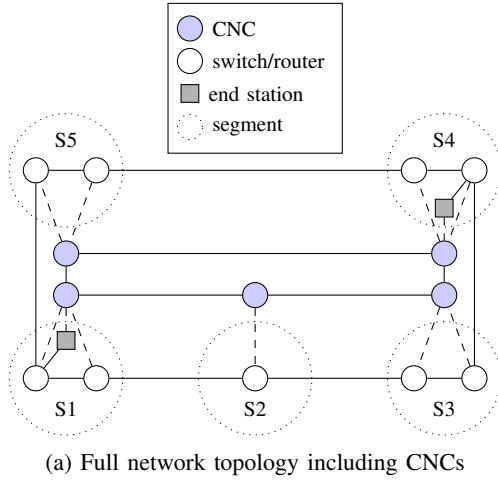


Fig. 3: Multi-segment topology example

V. EXPERIMENTATION

A. Experimental setup

A total of 5 scenarios were evaluated w.r.t. topology discovery time, control overhead and path signaling time. The *complex* scenario (denoted as '5-complex') uses the topology as shown in Fig. 3a. The other 4 scenarios have a topology with respectively 2, 3, 4 and 5 segments which are interconnected in a linear way (using segments S1, S2, S3, etc.). Note that in the *5-linear* scenario, all segments are used, but the links between S1 and S5 are disabled. All scenarios were conducted on an emulated multi-segment network deployed on the Virtual Wall of the local ilab.t testbed¹ consisting of Linux machines using Ubuntu 22.04, with an Intel® Core™ i5-9500 CPU, and disposing of 64GB RAM memory. The link delays in this network average 65 μ s, which means the segments span tens of kilometers. All experiments were repeated 10 times.

B. Topology discovery time

This experiment measures the time it takes for each CNC to discover both their local topology and the multi-segment topology once this local topology is obtained. The local topology discovery time includes the network nodes connecting with their CNC, receiving configuration, and neighbor discovery using LLDP. The multi-segment topology discovery time refers to the time it takes to distribute the network abstractions via the EW manager, such that every CNC can build the multi-segment topology. As presented in Fig. 4a, the local topology discovery time is typically in the order of seconds. This result is largely attributed to the LLDP update frequency, which is configured to 1s. Moreover, the local topology discovery time remains approximately constant in all scenarios. This is because the local topology discovery requires only LLDP interactions with local nodes and direct neighbors. The multi-segment topology discovery time increases in a linear fashion when more segments are added. This trend can be attributed to the distribution of topology abstractions over more segments.

¹<https://doc.ilabt.imec.be/ilabt/virtualwall/>

C. Control overhead

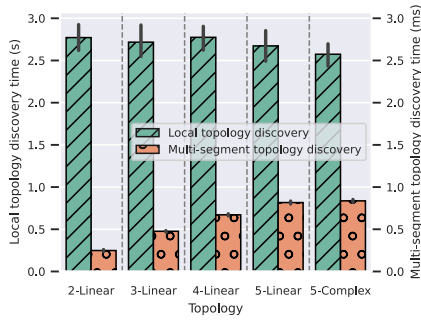
The control overhead for discovering both the local topology and the multi-segment topology is depicted in Fig. 4b. It's observed that the average bandwidth on the CNC-Agent links remains approximately constant in all scenarios. This is because the data that is exchanged between the agent and the CNC typically contains local information only. On the other hand, the bandwidth used by control traffic on the CNC-CNC links increases when the number of interconnected segments increase. This is due to the fact that more network segments introduce additional abstraction flooding, and every abstraction will contain more information (e.g., more neighbors). Still, the control overhead remains small in absolute terms.

D. Path signaling time

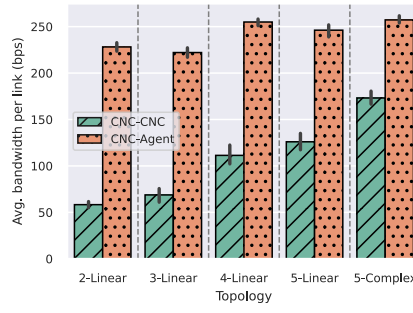
The path signaling time is defined as the time that it takes for a path to be successfully established over multiple segments. This includes computing the multi-segment route at the source segment, and forwarding the multi-segment request along the route and signaling back a confirmation via the EW manager. Note that the internal routing is not considered here. As depicted in Fig. 4c, the path signaling time increases linearly when the path spans an increasing number of segments. This can be explained by the fact that the multi-segment flow request needs to travel over more links to setup the end-to-end path when the multi-segment path becomes longer.

VI. CONCLUSION

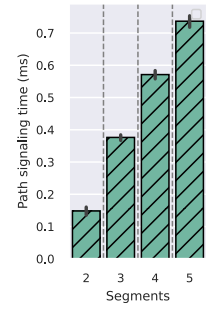
This paper introduced an East-Westbound control architecture enabling interaction between heterogeneous DetNet segments to set up multi-segment paths with guaranteed delay. By abstracting each individual network segment, the protocol provides a divide-and-conquer approach that significantly simplifies provisioning end-to-end paths spanning multiple heterogeneous deterministic network segments. The experiments highlight that interactions contained within a segment retain constant overhead, while the overhead of actions spanning multiple segments tends to increase when the number of



(a) Topology discovery time



(b) Control overhead



(c) Path signaling time

Fig. 4: Experimental results

involved segments rises. However, the overhead is quite small in absolute terms. The work presented in this paper also opens many doors to interesting future research directions, such as advanced and dynamic delay splitting algorithms, multi-segment routing algorithms, and extensive evaluations.

ACKNOWLEDGMENT

This research is partially funded by the Flemish FWO SBO S003921N VERI-END.com (Verifiable and elastic end-to-end communication infrastructures for private professional environments) project, and by the European Commission through the Horizon Europe/JU SNS project Hexa-X-II (Grant Agreement no. 101095759).

REFERENCES

- [1] "Ieee standard for local and metropolitan area networks—bridges and bridged networks," *IEEE Std 802.1Q-2022 (Revision of IEEE Std 802.1Q-2018)*, pp. 1–2163, 2022.
- [2] "Iso/iec/ieee international standard – information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements – part 1q: Bridges and bridged networks amendment 3: Enhancements for scheduled traffic," *ISO/IEC/IEEE 8802-1Q:2016/Amd.3:2017(E)*, pp. 1–62, 2018.
- [3] E. Grossman, "Deterministic Networking Use Cases," RFC 8578, May 2019. [Online]. Available: <https://www.rfc-editor.org/info/rfc8578>
- [4] P. Liu, Y. Li, T. Eckert, Q. Xiong, J. dong Ryoo, zhushiyin, and X. Geng, "Requirements for Scaling Deterministic Networks," Internet Engineering Task Force, Internet-Draft draft-ietf-detnet-scaling-requirements-05, Nov. 2023, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-ietf-detnet-scaling-requirements/05/>
- [5] H. Schulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550, Jul. 2003. [Online]. Available: <https://www.rfc-editor.org/info/rfc3550>
- [6] IEC CD/IEEE 802.1 TSN TG Ballot. Use Cases IEC/IEEE 60802. Accessed: 2024-04-01. [Online]. Available: <https://www.ieee802.org/1/files/public/docs2018/60802-industrial-use-cases-0918-v13.pdf>
- [7] T. Eckert, Y. Li, S. Bryant, A. G. Malis, J. dong Ryoo, P. Liu, G. Li, S. Ren, and F. Yang, "Deterministic Networking (DetNet) Data Plane - Tagged Cyclic Queuing and Forwarding (TCQF) for bounded latency with low jitter in large scale DetNets," Internet Engineering Task Force, Internet-Draft draft-eckert-detnet-tcqf-05, Jan. 2024, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-eckert-detnet-tcqf/05/>
- [8] N. Finn, P. Thubert, B. Varga, and J. Farkas, "Deterministic Networking Architecture," RFC 8655, Oct. 2019. [Online]. Available: <https://www.rfc-editor.org/info/rfc8655>
- [9] T. Stüber, L. Osswald, S. Lindner, and M. Menth, "A survey of scheduling algorithms for the time-aware shaper in time-sensitive networking (tsn)," *IEEE Access*, 2023.
- [10] A. Van Bemten, N. erić, A. Varasteh, S. Schmid, C. Mas-Machuca, A. Blenk, and W. Kellerer, "Chameleon: predictable latency and high utilization with queue-aware and adaptive source routing," in *Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies*, 2020, pp. 451–465.
- [11] J. Miserez, G. P. Sharma, and W. Tavernier, "Routing protocols exploiting queue information for deterministic networks," in *2023 19th International Conference on the Design of Reliable Communication Networks (DRCN)*. IEEE, 2023, pp. 1–8.
- [12] S. Schriegel, T. Kobzan, and J. Jasperneite, "Investigation on a distributed sdn control plane architecture for heterogeneous time sensitive networks," in *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, 2018, pp. 1–10.
- [13] L. Chen. (2019) TSN configuration interaction. Accessed: 2024-04-01. [Online]. Available: <https://www.ieee802.org/1/files/public/docs2019/new-chen-TSN-Configuration-Interaction-0719-v01.pdf>
- [14] G. Miranda, E. Municio, J. Haxhibeqiri, J. Hoebeke, I. Moerman, and J. M. Marquez-Barja, "Enabling time-sensitive network management over multi-domain wired/wi-fi networks," *IEEE Transactions on Network and Service Management*, 2023.
- [15] S. Höme, S. Kerschbaum, G. Steindl, and J. Dorr. (2020) Inter TSN domain communication concept. Accessed: 2024-04-01. [Online]. Available: <https://www.ieee802.org/1/files/public/docs2020/60802-Steindl-et-al-InterTsnDomainCommunication-0620-v3.pdf>
- [16] N. Reusch, M. Barzegaran, L. Zhao, S. S. Craciunas, and P. Pop, "Configuration optimization for heterogeneous time-sensitive networks," *Real-Time Systems*, vol. 59, no. 4, pp. 705–747, 2023.
- [17] M. Guo, G. Shou, Y. Liu, and Y. Hu, "Software-defined time-sensitive networking for cross-domain deterministic transmission," *Electronics*, vol. 13, no. 7, p. 1246, 2024.
- [18] G.-m. Lee, C.-w. Lee, and B.-h. Roh, "Qos support path selection for inter-domain flows using effective delay and directed acyclic graph in multi-domain sdn," *Electronics*, vol. 11, no. 14, p. 2245, 2022.
- [19] S. Bhattacharjee, K. Alexandris, and T. Bauschert, "Hierarchical control plane framework for multi-domain tsn orchestration," in *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*, 2023, pp. 26–34.
- [20] M. Böhm and D. Wermser, "Multi-domain time-sensitive networks—control plane mechanisms for dynamic inter-domain stream configuration," *Electronics*, vol. 10, no. 20, p. 2477, 2021.
- [21] "Ieee standard for local and metropolitan area networks - station and media access control connectivity discovery," *IEEE Std 802.1AB-2016 (Revision of IEEE Std 802.1AB-2009)*, pp. 1–146, 2016.
- [22] L. Zhao, P. Pop, and S. Steinhorst, "Quantitative performance comparison of various traffic shapers in time-sensitive networking," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2899–2928, 2022.
- [23] L. Maile, K.-S. Hielscher, and R. German, "Network calculus results for TSN: An introduction," in *2020 Information Communication Technologies Conference (ICTC)*. IEEE, 2020, pp. 131–140.