Header Proposal for the DetNet Application Layer

Marta Blanco Caamaño*, Luis M. Contreras*, Rafael A. López da Silva*
** Telefónica Innovación Digital, Madrid, Spain*

ABSTRACT
The new generation of real-time mission-critical applications requires high resilience and low-latency coordinated actions, surpassing the specifications outlined for 5G URLLC services. Examples include high-precision robot control and autonomous vehicles that cannot tolerate millisecond latency, and factory automation over wireless links that demand sub-millisecond end-to-end latency. To support the next generation of URLLC use cases, referred to as eXtreme URLLC (xURLLC), 6G systems will need to make faster and more reliable decisions at the network edge. One essential requirement in terms of connectivity is determinism. Currently, the solution to incorporate determinism at the IP layer is defined by IETF DetNet specifications [1]. In a DetNet domain, a DetNet stack is employed, consisting of two sub-layers: the Service sub-layer and the Forwarding sub-layer. This stack is documented in RFC 8655 [2]. What remains open is the definition of header fields in the application layer to incorporate determinism into the network. This paper presents a proposal for a DetNet header for the application layer.

Keywords: 6G, URLLC, DetNet, deterministic communication.

1. INTRODUCTION

DetNet is born as an extension to layer 3 for bringing at that level the functional concept of Time-Sensitive Networking (TSN) [3]. It operates at the IP layer and provides services through lower-layer technologies, such as MPLS or TSN, as defined by IEEE 802.1, or OTN (Optical Transport Network) [2]. This standard proposition opens the possibility to transport unicast or multicast flows of real-time applications that require an extremely low data loss rate (the reliability of one transmission of a 32-byte packet is expected to be at least 99.999%) and limited latency, such as <1 millisecond, within a domain.

1.1 DetNet Stack Model

DetNet has its own protocol stack consisting of two layers: the Service sub-layer and the Forwarding sub-layer, see Figure 1. DetNet functionalities will be situated within one of these two sub-layers, such as service protection in the Service sub-layer, or providing explicit routes and resource allocation to DetNet flows in the Forwarding sub-layer. Depending on the DetNet functionalities present in a network or application, all sub-layers may be required, or only one of them.

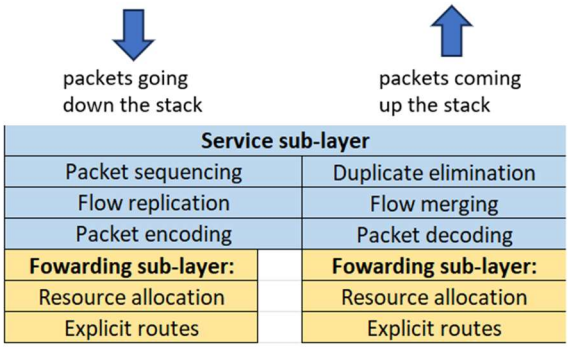

Figure 1. DetNet Stack Model.

Examples of technologies that can be used in the Service sub-layer to add functionality include PW, UDP, or GRE. On the other hand, examples of technologies that can be used in the Forwarding sub-layer to add functionality include IPv6, IPv4, MPLS TE LSPs, or MPLS SR [2].

1.2 Architecture

A deterministic network is composed of end systems, edge nodes, relay nodes, and transit nodes.

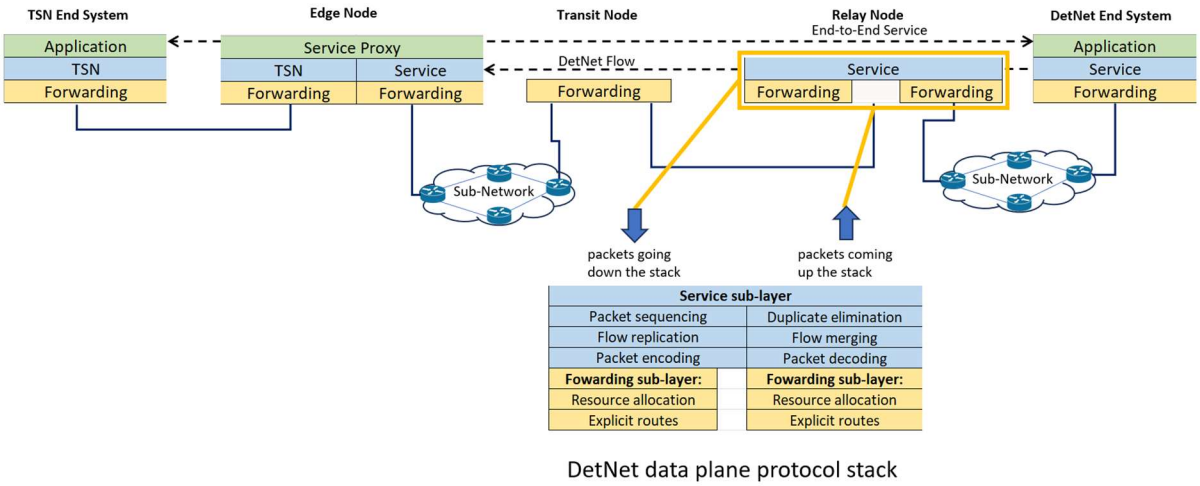

Figure 2. A simple DetNet-enabled network.

End-system

An end-system may or may not have specific DetNet functionality, meaning it may contain the Service and Forwarding sublayers, only one of them, or none. In case of not containing any of the sublayers, service proxies are required. Additionally, end-systems may or may not have the same Forwarding sublayer as the DetNet domain to which they are connected.

DetNet Edge, Relay, and Transit Nodes

As shown in the previous Figure 2, the edge nodes and relay nodes are DetNet aware because they contain both sub-layers (Service and Forwarding). Additionally, the edge nodes provide proxy service. In contrast, transit nodes only need to be aware of the DetNet Forwarding sub-layer.

1.3 Proposal of the paper

If we turn our attention back to Figure 2, the DetNet End Systems feature an application layer. The IETF leaves open the definition of the fields in the Application layer to incorporate determinism into the network. This paper introduces a proposal for a DetNet header at the Application layer.

2. HEADER PROPOSAL FOR THE DETNET APPLICATION LAYER

The Application layer corresponds to the IP layer. Our proposal is based on using an adaptation of the IPv6 Extended Fragment Header for IPv4 to identify traffic types and, based on this identification, provide different services accordingly [4]. The IPv4 sources would insert an IPv6 Destination Option with an Extended Fragment Header in an IPv6 extension header chain that would begin immediately after the end of the IPv4 header and would end immediately before the upper layer protocol header. The Extended Fragment Header option for the IPv6 Destination Options header is formatted as shown in Figure 3. In the field of *Identification* of this proposed new header, we will include information regarding the type of traffic being processed through a hexadecimal code, enabling the differentiation of various traffic types. In section 2.1, we introduce a potential traffic classification drawn from the TSN-Working Group [5].

2.1 Industrial traffic types

The main features for distinguishing types of traffic in an industrial environment are three:

- Cyclic. Traffic types consist of frames that can either be transmitted on a reoccurring time period (cyclic) or at no set period (acyclic). Available selections are:
  - Required. Traffic frames are transmitted cyclically.
  - Optional
- Data delivery requirements. Four options are specified:
  - Frame Latency. Exists a bounded timespan.
  - Flow Latency. Data delivery up to a certain number of frames or data size occurring over a defined period.
  - Deadline. Data delivery before a specific point in time.
  - No. No special data delivery requirements.
- Time-triggered transmission. Data transmission occurs at a specific point in time based upon the Working Clock. Available selections are:

- Required
- Optional. Implementation of time-triggered transmission is at the discretion of the user.

Table 1 summarizes relevant industrial automation traffic types and their associated characteristics. In the column labelled *Traffic type hexadecimal code*, we propose a coding scheme to be added in the Identification field of the proposed header to reflect the type of delivered traffic.

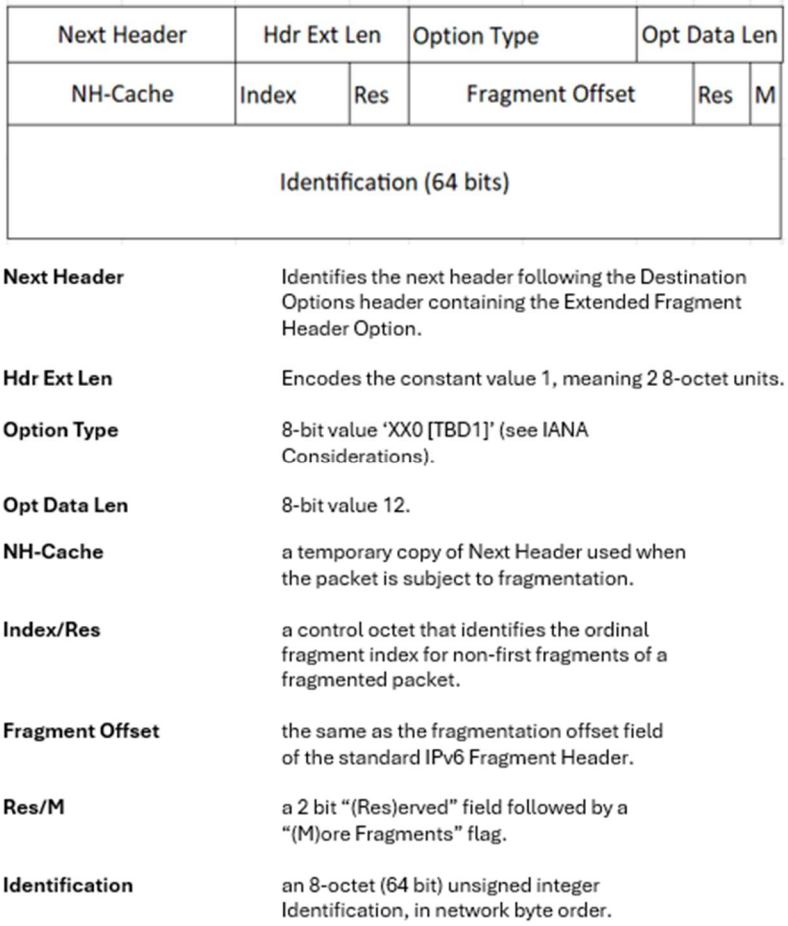

Figure 3. IPv6 Extended Fragment Header.

3. SIMULATION USING THE PROPOSED HEADER

We have executed an implementation utilizing the P4 programming language on the BMv2 software switch, showcasing the structure of a DetNet packet with the newly proposed header. The objective of this implementation is to transmit an Ethernet-IPv4-TCP packet to the BMv2 switch and have it egress the switch with the DetNet stack fields, as detailed in the Introduction section, along with the newly proposed header. The resulting packet would comprise Ethernet-Forwarding sublayer-Service sublayer-IPv4-Extension header-TCP. For packet transmission, we will employ the *scapy* tool, while *Wireshark* will be utilized for visualizing the packet at the BMv2 egress. The following Figure 4 illustrates an example of the output of a DetNet packet for Best Effort Low traffic.

4. CONCLUSIONS

In this paper, we have proposed a header to provide determinism at the application layer, as the IETF leaves this matter open-ended. We see it necessary to define this header as a means to identifying the various types of traffic entering the network and thereby be able to offer deterministic behaviour. As a future line of work, we will conduct tests using this new header to assess its feasibility, comparing its advantages and disadvantages. Additionally, to visualize each newly defined field, we will rely on an implementation based on the *Lua* language provided by *Wireshark* for representing protocols not inherent to the tool.

ACKNOWLEDGEMENTS

This work has been partially funded by the European Commission Horizon Europe SNS JU project DESIRE6G (GA 101096466).

Traffic type name	Traffic type code	**Traffic type hexadecimal code**	Traffic-type-category	Cyclic	Data delivery requirements	Time-triggered transmission
Isochronous	H	**0x08**	IA time-aware-stream	Required	Deadline	Required
Cyclic-synchronous	G	**0x07**	IA time-aware-stream	Required	Frame Latency	Required
Cyclic-asynchronous	F	**0x06**	IA stream	Required	Frame Latency	Optional
Alarms & Events	E	**0x05**	IA traffic engineered non-stream	Optional	Flow Latency	Optional
Configuration & Diagnostics	D	**0x04**	IA traffic engineered non-stream	Optional	Flow Latency	Optional
Network Control	C	**0x03**	IA traffic engineered non-stream	Optional	Flow Latency	Optional
Best Effort High	B	**0x02**	IA non-stream	Optional	No	Optional
Best Effort Low	A	**0x01**	IA non-stream	Optional	No	Optional

Table 1. Industrial automation traffic types summary.

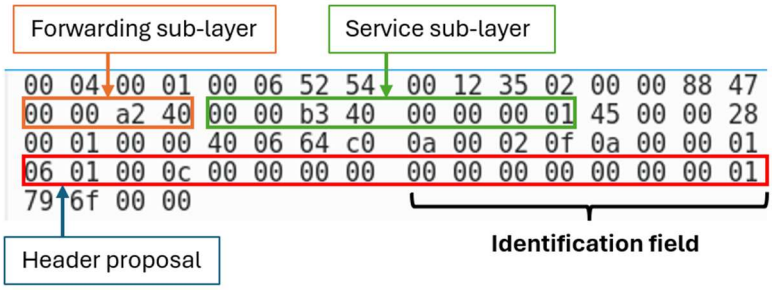

Figure 4. A DetNet packet example with the proposal application header.

REFERENCES

[1] J. H. Wang, Y. Zhou, and Y. Xu, 'Deterministic Network', in *Fundamentals of 6G Communications and Networking*, X. Lin, J. Zhang, Y. Liu, and J. Kim, Eds., Cham: Springer International Publishing, 2024, pp. 633–665. doi: 10.1007/978-3-031-37920-8_25.

[2] N. Finn, P. Thubert, B. Varga, and J. Farkas, 'Deterministic Networking Architecture', Internet Engineering Task Force, Request for Comments RFC 8655, Oct. 2019. doi: 10.17487/RFC8655.

[3] 'Time-Sensitive Networking (TSN) Task Group |'. Accessed: Apr. 12, 2024. [Online]. Available: https://1.ieee802.org/tsn/

[4] F. Templin, 'IPv6 Extended Fragment Header', Internet Engineering Task Force, Internet Draft draft-templin-6man-ipid-ext2-01, Feb. 2024. Accessed: Mar. 21, 2024. [Online]. Available: https://datatracker.ietf.org/doc/draft-templin-6man-ipid-ext2-01

[5] 'IEC/IEEE 60802 TSN Profile for Industrial Automation |'. Accessed: Apr. 12, 2024. [Online]. Available: https://1.ieee802.org/tsn/iec-ieee-60802/