DetNet
Internet-Draft
Intended status: Standards Track
Expires: 15 August 2026

Shaofu. Peng
ZTE
Peng. Liu
China Mobile
Kashinath. Basu
Oxford Brookes University

11 February 2026

Mechanism to control jitter caused by policing in Detnet
draft-peng-detnet-policing-jitter-control-02

Abstract

This document presents a noble mechanism to eliminate jitter caused by policing delay in a network. It needs to be used in combination with a queueing mechanism that provides low jitter for the DetNet path, and ultimately provides a low jitter guarantee for the DetNet flow.

Table of Contents

1. Introduction

A policing function, as defined in [RFC2216] differentiates those packets in a traffic flow which conform to a particular token bucket specification from those packets which do not. A Token Bucket is a particular form of traffic specification consisting of a "token rate" r and a "bucket size" b. More specifically, the traffic must obey the rule that over all time periods, the amount of data sent cannot exceed r*T+b, where T is the length of the time period. The common treatment accorded to nonconforming packets may be relegating the packet to best effort service, discarding the packet, or marking the packet in some fashion.

A DetNet [RFC8655] flow with conforming packets released to the network is the condition that must be followed to obtain the DetNet services. According to DetNet architecture [RFC8655], rate limiting and shaping functions at the ingress of the DetNet domain must be applied. This is also consistent with the problem statement of flow characterization in [RFC8557]. Assuming that there is enough buffer space at the network entrance to store nonconforming packets, then that is important for application flows that expect zero packet loss. A conforming packet may experience zero policing delay, while a

nonconforming packet may experience non-zero policing delay. After policing at the network entrance, each packet of the application flow will be guaranteed the bounded delay and jitter by the applied queueing mechanisms in the DetNet domain. Generally, the applied queueing mechanism is only responsible for the delay performance of the DetNet path, but not the runtime policing delay at the network entrance.

Although some application flows may declare that they will accept additional jitter caused by policing delay for nonconforming packets at the edges, other application flows, especially those that are extremely sensitive to jitter, hope not to get larger jitter due to any reason including the policing delay.

This document describes a mechanism to eliminate jitter caused by policing delay. It needs to be used in combination with a queueing mechanism that provides low jitter for the DetNet path, and ultimately provides a low jitter guarantee for the application flow.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Overview of the Solution

The end-to-end delay experienced by the application flow can be considered to consist of two parts: edge-to-edge policing delay, and the DetNet path delay. According to flow’s end-to-end delay requirement and edge-to-edge policing delay budget, a DetNet path with expected delay performance (i.e., end-to-end delay requirement minus edge-to-edge policing delay budget) can be calculated and setup. An appropriate edge-to-edge policing delay budget can be configured for the application flow according to its TSpec and actual possible arrival pattern, and generally be the maximum policing delay that may be possibly experienced at the network entrance.

The edge-to-edge policing delay budget is consumed by the headend and endpoint of the DetNet path.

```
edge-to-edge policing delay budget = headend policing delay +
endpoint damping delay
```

The headend policing delay is the runtime policing delay experienced at the network entrance. The endpoint damping delay is obtained by subtracting the headend policing delay from the edge-to-edge policing delay budget.

Figure 1 shows that the relationship between these delay components.

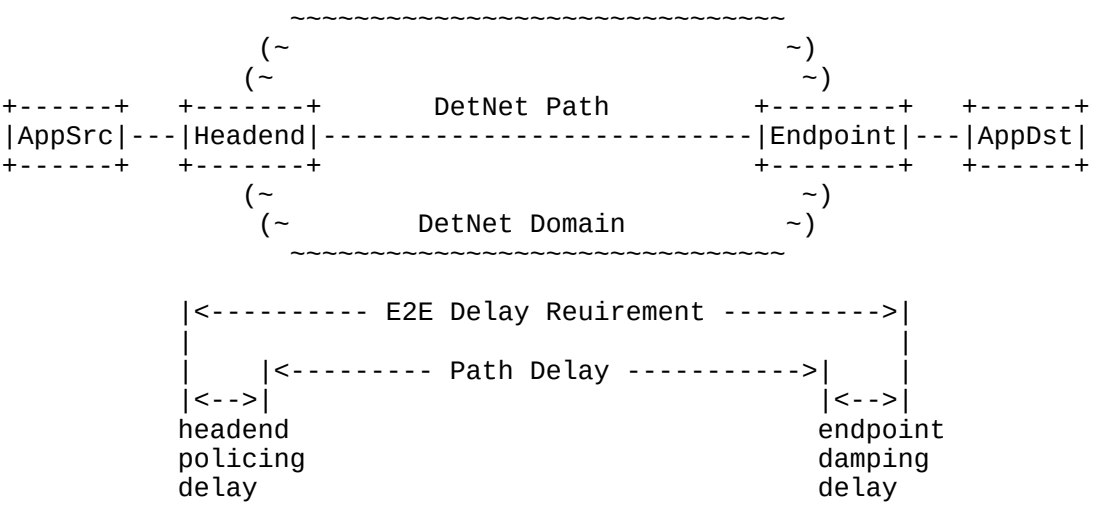

Figure 1: Relationship Between Delay Elements

When the headend of the DetNet path receives the packet from the application source, it may carry the endpoint damping delay value in the encoded packet sent to the endpoint, and the endpoint damping delay will be used as the holding time imposed on the endpoint before the packet is delivered to the application destination.

Note that some queueing mechanisms, such as [I-D.ietf-detnet-deadline-based-forwarding] and [I-D.ietf-detnet-packet-timeslot-mechanism], also provide the latency deviation (E) information of the DetNet path. Depending on the implementation, the endpoint can use different buffers to firstly implement jitter control based on path latency deviation (E) and secondly jitter control based on endpoint damping delay, or use the same buffer to uniformly implement jitter control based on the sum of path latency deviation (E) and endpoint damping delay. However, it must use separate fields in the packet to carry these two values.

Therefore, a flow with possible nonconforming packets will be regulated and changed to be conforming at the network entrance, then get the DetNet service within the network, and finally reverted to the nonconforming pattern at the network exit, and then delivered to the application destination.

4. Set Edge-to-edge Policing Delay Budget

The edge-to-edge policing delay budget can be configured for the application flow according to its TSpec and actual possible arrival pattern.

For example, an applicaiton flow has service burst interval (SBI) 100 us, and three packets P1, P2, P3 per SBI. Assuming the maximum packet size is 1000 bits. It can be seen that the bandwidth required by the application flow is 30 Mbps. The packet size 1000 bits and the required bandwidth 30 Mbps are also the leaky bucket policing parameters at the network entrance.

In the ideal case, a conforming pattern of this application flow is that the three packets arrive evenly at the network entrance, i.e., with packet interval 33 us. However, an extremely case of nonconforming pattern may be that P1, P2, P3 arrived back-to-back. After leaky bucket based regulation, P1, P2 and P3 will get different policing delays, of which P3 has the largest policing delay, which may be 2/3 of SBI and can be used as edge-to-edge policing delay budget.

There may be other settings of edge-to-edge policing delay budget that are not based on the above extreme case, such as sampling the most likely actual arrival pattern of flow to set a smaller edge-to-edge policing delay budget.

The network entrance node should maintain the edge-to-edge policing delay budget for each application flow, and when it receives a packet from the application source, it identifies the flow and applies the corresponding edge-to-edge policing delay budget. If the edge-to-edge policing delay budget is M, and the runtime policing delay of the packet is S, then the endpoint damping delay for that packet equals to M - S.

5. Multi-domain considerations

In the case of multi-domain, all domains may apply the same or different queueing machanisms. For each transit domain and egress domain, the input traffic should be conforming and then get the DetNet services. The output traffic from the upstream domain must be conforming, that can be achieved based on path latency deviation (E).

There are two options to implement policing jitter control.

5.1. Separate Policing for Each Domain

This option is to implement policing jitter control at the entrance and exit of each domain independently. It is only applicable to scenarios where transit domains maintain flow states.

At each domain entrance, it maintain the edge-to-edge policing delay budget for the application flow, regulate the nonconforming arrived packets, and calculate the endpoint damping delay for each packet. Then, at each domain exit, packets are held based on the endpoint damping delay. In this option, each domain contribute a separate edge-to-edge policing delay budget to the end-to-end delay.

When the packet leaves the upstream domain, the scheduling metadata related to the queueing mechanism of the upstream domain and the endpoint damping delay information are removed, and the application header is exposed. The current domain entrance will re-encapsulate the scheduling metadata related to the queueing mechanism of the current domain and the endpoint damping delay information.

The limitation of this option is the states per flow maintained at each domain entrance.

5.2. Single Policing for All Domains

This option is to implement policing jitter control only at the ingress domain entrance and the egress domain exit. It is applicable to scenarios where transit domains do not maintain flow states.

At the ingress domain entrance, it maintain the edge-to-edge policing delay budget for the application flow, regulate the nonconforming arrived packets, and calculate the endpoint damping delay for each packet. Then, at the egress domain exit, packets are held based on the endpoint damping delay. In this option, only a single edge-to-edge policing delay budget is contributed to the end-to-end delay.

When the packet leaves the upstream domain, the scheduling metadata related to the queueing mechanism of the upstream domain is removed, but the endpoint damping delay information remains unchanged. The current domain entrance will re-encapsulate the scheduling metadata related to the queueing mechanism of the current domain. However, if all domains use the same queueing mechanism, they may optionally share a single metadata to avoid removing and re-encapsulating.

This option has less cost than the first option, i.e., no states per flow maintained on each domain entrance.

6. Encoding Considerations

A new IPv6 option for DOH Options header ([RFC8200]) to carry endpoint damping delay is defined in [I-D.peng-6man-delay-options].

A new ancillary data for MPLS MNA header ([I-D.ietf-mpls-mna-hdr]) to carry endpoint damping delay is for further definition.

7. IANA Considerations

This document need not require IANA allocations.

8. Security Considerations

TBD.

9. Acknowledgements

TBD.

10. References

10.1. Normative References

[I-D.ietf-detnet-deadline-based-forwarding]
Peng, S., Du, Z., Basu, K., cheng, C., Yang, D., and C. Liu, "Deadline Based Deterministic Forwarding", Work in Progress, Internet-Draft, draft-ietf-detnet-deadline-based-forwarding-00, 16 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-detnet-deadline-based-forwarding-00>.

[I-D.ietf-detnet-packet-timeslot-mechanism]
Peng, S., Liu, P., Basu, K., Liu, A., Yang, D., Peng, G., and J. Zhao, "Timeslot Queueing and Forwarding Mechanism", Work in Progress, Internet-Draft, draft-ietf-detnet-packet-timeslot-mechanism-00, 16 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-detnet-packet-timeslot-mechanism-00>.

[I-D.ietf-mpls-mna-hdr]
Rajamanickam, J., Gandhi, R., Zigler, R., Song, H., and K. Kompella, "MPLS Network Action (MNA) Sub-Stack Specification including In-Stack Network Actions and Data", Work in Progress, Internet-Draft, draft-ietf-mpls-mna-hdr-20, 6 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-mpls-mna-hdr-20>.

[I-D.peng-6man-delay-options]
Peng, S., "Delay Options", Work in Progress, Internet-Draft, draft-peng-6man-delay-options-01, 26 January 2026, <https://datatracker.ietf.org/doc/html/draft-peng-6man-delay-options-01>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2216] Shenker, S. and J. Wroclawski, "Network Element Service Specification Template", RFC 2216, DOI 10.17487/RFC2216, September 1997, <https://www.rfc-editor.org/info/rfc2216>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8557] Finn, N. and P. Thubert, "Deterministic Networking Problem Statement", RFC 8557, DOI 10.17487/RFC8557, May 2019, <https://www.rfc-editor.org/info/rfc8557>.

[RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <https://www.rfc-editor.org/info/rfc8655>.

Authors' Addresses

Shaofu Peng
ZTE
China
Email: peng.shaofu@zte.com.cn

Peng Liu
China Mobile
China
Email: liupengyjy@chinamobile.com

Kashinath Basu
Oxford Brookes University
United Kingdom
Email: kbasu@brookes.ac.uk