Stream: Internet Engineering Task Force (IETF)
RFC: 9550
Category: Informational
Published: March 2024
ISSN: 2070-1721
Authors: B. Varga, Ed. *Ericsson*
J. Farkas *Ericsson*
S. Kehrer *Hirschmann Automation and Control GmbH*
T. Heer *Hirschmann Automation and Control GmbH*

RFC 9550
Deterministic Networking (DetNet): Packet Ordering Function

Abstract

The replication and elimination functions of the Deterministic Networking (DetNet) architecture can result in out-of-order packets, which is not acceptable for some time-sensitive applications. The Packet Ordering Function (POF) algorithm described in this document enables restoration of the correct packet order when the replication and elimination functions are used in DetNet networks. POF only provides ordering within the latency bound of a DetNet flow; it does not provide any additional reliability.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9550.

Copyright Notice

Table of Contents

1. Introduction

The DetNet Working Group has defined the Packet Replication Function (PRF) and Packet Elimination Function (PEF) for achieving extremely low packet loss. PRF and PEF are described in [RFC8655] and provide service protection for DetNet flows. This service protection method relies on copies of the same packet sent over multiple maximally disjoint paths and uses sequencing information to eliminate duplicates. A possible implementation of PRF and PEF functions is described in [IEEE8021CB], and the related YANG model is defined in [IEEEP8021CBcv].

In general, use of per-packet replication and elimination functions can result in out-of-order delivery of packets, which is not acceptable for some deterministic applications. Correcting packet order is not a trivial task; therefore, details of a Packet Ordering Function (POF) are specified in this document. In [RFC8655], the IETF DetNet Working Group defined the external observable result of a POF function (i.e., that packets are reordered), but without any implementation details.

So far in packet networks, out-of-order delivery situations have been handled at higher OSI layers at the endpoints/hosts (e.g., in the TCP stack when packets are sent to the application layer) and not within a network in nodes acting at the Layer 2 or Layer 3 OSI layers.

Figure 1 shows a DetNet flow on which Packet Replication, Elimination, and Ordering Functions (PREOF) are applied during forwarding from source to destination.

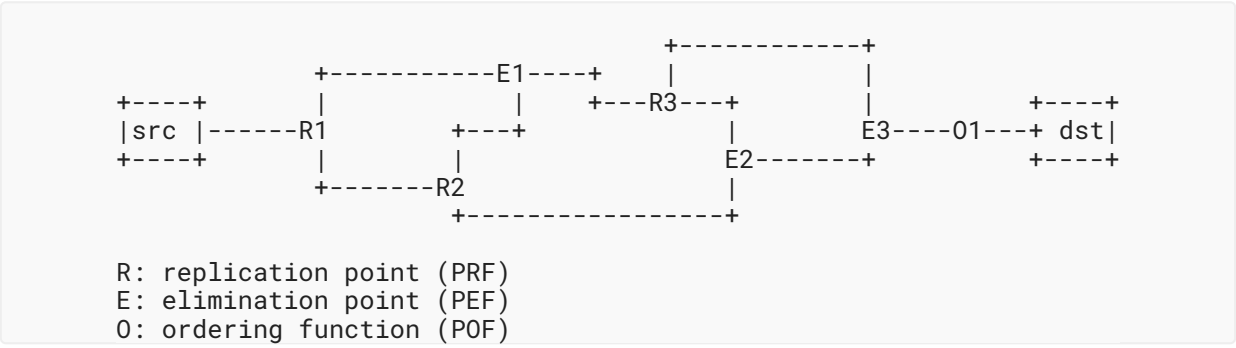

Figure 1: PREOF Scenario in a DetNet Network

In general, the use of PREOF functions requires sequencing information to be included in the packets of a DetNet compound flow. This can be done by adding a sequence number as part of DetNet encapsulation [RFC8655]. Sequencing information is typically added once, at or close to the source.

It is important to note that different applications can react differently to out-of-order delivery. A single out-of-order packet (e.g., packet order #1, #3, #2, #4, #5) is interpreted by some application as a single error, but other applications treat it as three errors in a row. For example, in industrial scenarios, three errors in a row is a typical error threshold and can cause the application to stop (e.g., go to a fail-safe state).

POF ensures in-order delivery for packets within the latency bound of the (DetNet) flow. POF does not correct errors in the packet flow (e.g., duplicate packets or packets that are too late).

2. Terminology

2.1. Terms Used in This Document

This document uses the terminology established in the DetNet architecture [RFC8655]; the reader is assumed to be familiar with that document and its terminology.

2.2. Abbreviations

The following abbreviations are used in this document:

DetNet	Deterministic Networking
PEF	Packet Elimination Function
POF	Packet Ordering Function
PREOF	Packet Replication, Elimination, and Ordering Functions
PRF	Packet Replication Function

3. Requirements for POF Implementations

The requirements for a POF function are:

- To solve the out-of-order delivery problem of the replication and elimination functions of DetNet networks.
- To consider the delay bound requirement of a DetNet flow.
- To be simple and to require, in network nodes, only a minimum set of state/configuration parameters and resources per DetNet flow.
- To add minimal or no delay to the forwarding process of packets.
- To not require synchronization between PREOF nodes.

Some aspects are explicitly out of scope for a POF function:

- To eliminate the delay variation caused by the packet ordering. Dealing with delay variation is a DetNet forwarding sub-layer target, and it can be achieved, for example, by placing a de-jitter buffer or flow regulator (e.g., shaping) function after the POF functionality.

4. POF Algorithms

4.1. Prerequisites and Assumptions

The POF algorithm discussed in this document makes some assumptions and trade-offs regarding the characteristics of the sequence of received packets. In particular, the algorithm assumes that a PEF is performed on the incoming packets before they are handed to the POF function. Hence, the sequence of incoming packets can be out-of-order or incomplete but cannot contain duplicate packets. However, the PREOF functions run independently without any state exchange required between the PEF and the POF or the PRF and the POF. Error cases in which duplicate packets are presented to the POF can lead to out-of-order delivery of duplicate packets and to increased delays.

The algorithm further requires that the delay difference between two replicated packets that arrive at the PEF before the POF is bounded and known. Error cases that violate this condition (e.g., a packet that arrives later than this bound) will result in out-of-order packets.

The algorithm also makes some trade-offs. For simplicity, it is designed to allow for some out-of-order packets directly after initialization. If this is not acceptable, Section 4.5 provides an alternative initialization scheme that prevents out-of-order packets in the initialization phase.

4.2. POF Building Blocks

The method described in this document provides POF for DetNet networks. The configuration parameters of POF can be derived when engineering the DetNet flow through the network.

The POF method is provided via the following:

Delay calculator: Calculates buffering time for out-of-order packets. Buffering time considers (i) the delay difference of paths used for forwarding the replicated packets and (ii) the bounded delay requirement of the given DetNet flow.

Conditional buffer: Used for buffering the out-of-order packets of a DetNet flow for a given time.

Note: The conditional buffer of POF increases the burstiness of the traffic as it only adds delay for some of the packets.

Figure 2 shows the building blocks of a possible POF implementation.

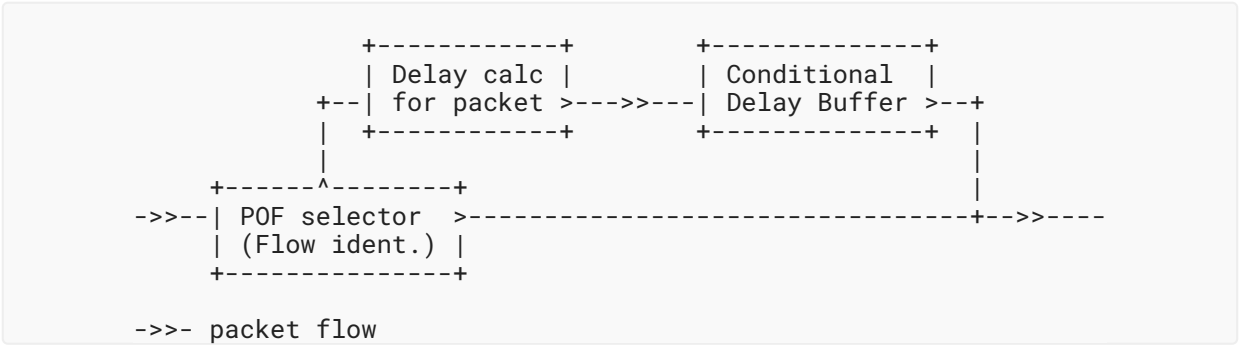

Figure 2: POF Building Blocks

4.3. The Basic POF Algorithm

The basic POF algorithm delays all out-of-order packets until all previous packets arrive or a given time (POFMaxDelay) elapses. The basic POF algorithm works as follows:

- The sequence number of the last forwarded packet (POFLastSent) is stored for each DetNet flow.
- The sequence number (seq_num) of a received packet is compared to that of the last forwarded one (POFLastSent).
- If (seq_num <= POFLastSent + 1)
  - Then the packet is forwarded without buffering, and "POFLastSent" is updated (POFLastSent = seq_num).
  - Else, the received packet is buffered.
- A buffered packet is forwarded from the buffer when its seq_num becomes equal to "POFLastSent + 1" OR a predefined time ("POFMaxDelay") elapses.
- When a packet is forwarded from the buffer, "POFLastSent" is updated with its seq_num (POFLastSent = seq_num).

Notes:

- The difference of sequence number in consecutive packets is bounded due to the history window of the elimination function before the POF. Therefore, "<=" can be evaluated despite the circular sequence number space. A possible implementation of the PEF function and the impact of the history window are described in [IEEE8021CB].
- The algorithm can be extended to cope with multiple failure scenarios (i.e., simultaneous packet loss and out-of-order packets) when the expiration of the timer for a packet with sequence number N triggers the release of some packets with a sequence number smaller than N.

The state used by the basic POF algorithm (i.e., "POFLastSent") needs initialization and maintenance. This works as follows:

- The next received packet is forwarded and the POFLastSent updated when the POF function is reset OR no packet is received for a predefined time ("POFTakeAnyTime").
- The reset of POF erases all packets from the time-based buffer used by POF.

The basic POF algorithm has two parameters to engineer:

- "POFMaxDelay", which cannot be smaller than the delay difference of the paths used by the flow.
- "POFTakeAnyTime", which is calculated based on several factors, for example, the settings of the elimination function(s) relating to RECOVERY_TIMEOUT before the POF, the flow characteristics (e.g., inter-packet time), and the delay difference of the paths used by the flow.

Design of these parameters is out of scope for this document.

Note: Multiple network failures can impact the POF function (e.g., complete outage of all redundant paths).

The basic POF algorithm increases the delay of packets with maximum "POFMaxDelay" time. In-order packets are not delayed. This basic POF method can be applied in all network scenarios where the remaining delay budget of a flow at the POF point is larger than "POFMaxDelay" time.

Figure 3 shows the delay budget relations at the POF point.

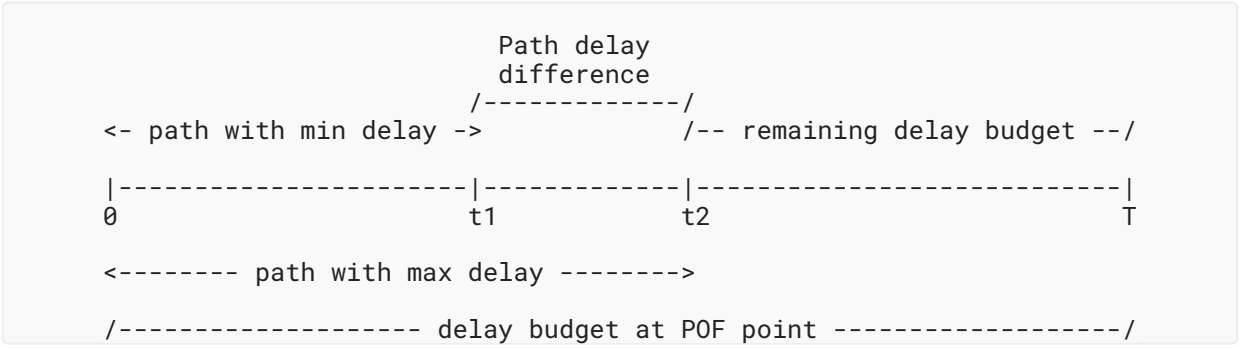

Figure 3: Delay Budget Relations at the POF Point

4.4. The Advanced POF Algorithm

In network scenarios where the remaining delay budget of a flow at the POF point is smaller than "POFMaxDelay" time, the basic method needs extensions.

The issue is that packets on the longest path cannot be buffered in order to keep the delay budget of the flow. It must be noted that such a packet (i.e., forwarded over the longest path) needs no buffering as it is the last chance to deliver a packet with a given sequence number. This is because all replicas already arrived via a shorter path(s).

The advanced POF algorithm needs two extensions of the basic POF algorithm:

- to identify the received packet's path at the POF location and
- to make the value of "POFMaxDelay" for buffered packets path dependent ("POFMaxDelay_i", where "i" notes the path the packet has used).

By identifying the path of a given packet, the POF algorithm can use this information to select what predefined time "POFMaxDelay_i" to apply for the buffered packet. So, in the advanced POF algorithm, "POFMaxDelay" is an array that contains the predefined and path-specific buffering time for each redundant path of a flow. Values in the "POFMaxDelay" array are engineered to fulfill the delay budget requirement.

Design of these parameters is out of scope for this document.

Note: For the advanced POF algorithm, the path-dependent delays might result in multiple packets triggering the "maximum delay reached" at exactly the same time. The transmission order of these packets should be done in their seq_num order.

The method for identifying the packet's path at the POF location depends on the network scenario. It can be implemented via various techniques, for example, using ingress interface information, encoding the path in the packet itself (e.g., replication functions can set different FlowID per egress what can be used as a PathID), or other means. Methods for identifying the packet's path are out of scope for this document.

Note: When using the advanced POF algorithm, it might be advantageous to combine PEF and POF locations in the DetNet network, as this can simplify the method used for identifying the packet's path at the POF location.

4.5. Further Enhancements of POF Algorithms

POF algorithms can be further enhanced by distinguishing the case of initialization from normal operation at the price of more states and more sophisticated implementation. Such enhancements could, for example, react better after some failure scenarios (e.g., complete outage of all paths of a DetNet flow) and can be dependent on the PEF implementation.

The challenge for POF initialization is that, for example, after a reset, it is not known whether the first received packet is in-order or out-of-order. The original initialization (see before) considers the first packet as in-order, so out-of-order packet(s) during "POFMaxTime"/"POFMaxTime_path_i" time -- after the first packet is received -- cannot be corrected. The motivation behind such an initialization is simplicity of POF implementation.

A possible enhancement of POF initialization works as follows:

- After a reset, all received packets are buffered with their predefined timer ("POFMaxTime"/"POFMaxTime_path_i").
- No basic/advanced POF rules are applied until the first timer expires.
- When the first timer expires, the packet with lowest seq_num in the buffer is selected and forwarded, and "POFLastSent" is set with its seq_num.

- The basic/advanced POF rules are applied for the packet(s) in the buffer and the subsequently received packets.

4.6. Selecting and Using the POF Algorithm

The selection of the POF algorithm depends on the network scenario and the remaining delay budget of a flow. Using POF and calculating its parameters require proper design. Knowing the path delay difference is essential for the POF algorithms described here. Failure scenarios breaking the design assumptions can impact the result of POF (e.g., packet received out of the expected worst-case delay window -- calculated based on the path delay difference -- can result in unwanted out-of-order delivery).

In DetNet scenarios, there is always an elimination function before the POF (therefore, duplicates are not considered by the POF). Implementing them together in the same node allows POF to consider PEF events/states during the reordering. For example, under normal circumstances, the difference of sequence number in consecutive packets is bounded due to the history window of PEF. However, in some scenarios (e.g., reset of sequence number), the difference can be much larger than the size of the history window.

5. Control and Management Plane Parameters for POF

POF algorithms need setting of the following parameters:

- Basic POF
  - "POFMaxDelay"
  - "POFTakeAnyTime"
- Advanced POF
  - "POFMaxDelay_i" for each possible path i
  - "POFTakeAnyTime"
  - Configuration(s) related to network path identification

Note: In a proper design, "POFTakeAnyTime" is always larger than "POFMaxDelay".

6. Security Considerations

PREOF-related security considerations (including POF) are described in Section 3.3 of [RFC9055]. There are no additional POF-related security considerations originating from this document.

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[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>.

[RFC9055] Grossman, E., Ed., Mizrahi, T., and A. Hacker, "Deterministic Networking (DetNet) Security Considerations", RFC 9055, DOI 10.17487/RFC9055, June 2021, <https://www.rfc-editor.org/info/rfc9055>.

8.2. Informative References

[IEEE8021CB] IEEE, "IEEE Standard for Local and metropolitan area networks -- Frame Replication and Elimination for Reliability", IEEE Std 802.1CB-2017, DOI 10.1109/IEEESTD.2017.8091139, October 2017, <https://standards.ieee.org/standard/802_1CB-2017.html>.

[IEEEP8021CBcv] Kehrer, S., "FRER YANG Data Model and Management Information Base Module", IEEE P802.1CBcv /D1.2 P802.1CBcv, March 2021, <https://www.ieee802.org/1/files/private/cv-drafts/d1/802-1CBcv-d1-2.pdf>.

Acknowledgements

Authors extend their appreciation to Gyorgy Miklos, Ehsan Mohammadpour, Ludovic Thomas, Greg Mirsky, Jeong-dong Ryoo, Fan Yang, Toerless Eckert, Norman Finn, and Ethan Grossman for their insightful comments and productive discussion that helped to improve the document.

Authors' Addresses

Balazs Varga (EDITOR)
Ericsson
Budapest
Magyar Tudosok krt. 11.
1117
Hungary
Email: balazs.a.varga@ericsson.com

Janos Farkas
Ericsson
Budapest
Magyar Tudosok krt. 11.
1117
Hungary
Email: janos.farkas@ericsson.com

Stephan Kehrer
Hirschmann Automation and Control GmbH
Stuttgarter Strasse 45-51.
72654 Neckartenzlingen
Germany
Email: Stephan.Kehrer@belden.com

Tobias Heer
Hirschmann Automation and Control GmbH
Stuttgarter Strasse 45-51.
72654 Neckartenzlingen
Germany
Email: Tobias.Heer@belden.com