DetNet
Internet-Draft
Intended status: Informational
Expires: 25 April 2022

B. Varga, Ed.
J. Farkas
Ericsson
S. Kehrer
T. Heer
Hirschmann Automation and Control GmbH
22 October 2021

Deterministic Networking (DetNet): Packet Ordering Function
draft-varga-detnet-pof-02

Abstract

Replication and Elimination functions of DetNet [RFC8655] may result in out-of-order packets, which may not be acceptable for some time-sensitive applications. The Packet Ordering Function (POF) algorithm described herein enables to restore the correct packet order when replication and elimination functions are used in DetNet networks.

Table of Contents

1. Introduction

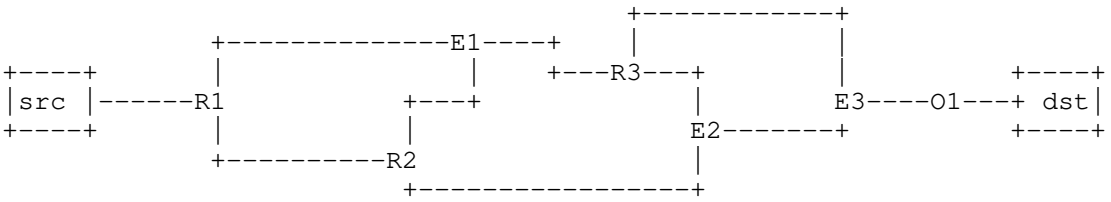
The DetNet Working Group has defined packet replication (PRF) and packet elimination (PEF) functions for achieving extremely low packet loss. PRF and PEF are described in [RFC8655] and provide service protection for DetNet flows. This service protection method relies on copies of the same packet sent 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 may result in out-of-order delivery of packets, which may not be acceptable for some deterministic applications. Correcting packet

order is not a trivial task, therefore details of a Packet Ordering Function (POF) are specified herein. The IETF DetNet WG has defined in [RFC8655] 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 were handled at higher OSI layers at the end-points/hosts (e.g., in the TCP stack when packets are sent to 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 PREOF functions are applied during forwarding from source to destination.



R: replication point (PRF)
E: elimination point (PEF)
O: ordering function (POF)

Figure 1: PREOF scenario in a DetNet network

Important to note, that application may react differently on out-of-order delivery. A single out-of-order packet (E.g., packet order: #1, #3, #2, #4, #5) may be interpreted by some applications as a single error, but some other applications may treat it as a 3 errors in-a-row situation. 3 errors in-a-row is a usual error threshold and may cause the application to stop (e.g., to tranistion to a fail safe state).

2. Terminology

2.1. Terms Used in This Document

This document uses the terminology established in the DetNet architecture [RFC8655], and 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.

2.3. 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. Requirements on POF Implementations

The requirements on 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 states/configuration parameters and resources per DetNet Flow.

* to add only minimal or no delay to the forwarding process of packets.

* not to 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 tradeoffs regarding the characteristics of the sequence of received packets. In particular, the algorithm assumes that a Packet Elimination Function (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 the POF is presented duplicate packets may lead to out of order delivery of duplicate packets as well as to increased delays.

The algorithm further requires that the delay difference between two replicated packets that arrive at the PRF 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 tradeoffs. For simplicity, it is designed in a way that allows 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 herein provides POF for DetNet networks. The configuration parameters of POF can be derived during engineering the DetNet flow through the network.

The POF method is provided via:

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

2. Delay calculator: 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.

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

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

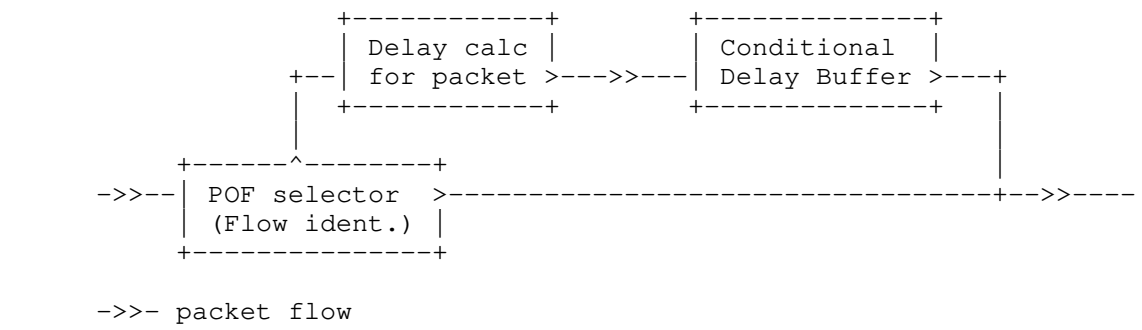


Figure 2: POF Building Blocks

4.3. The Basic POF Algorithm

The basic POF algorithm delays all out-of-order packets until all previous packet arrives 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 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).

Note: 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 of the circular sequence number space.

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

* The next received packet must be forwarded and the POFLastSent updated when the POF function was reset OR no packet was received for a predefined time ("POFTakeAnyTime").

* The reset of POF erases all frames/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 RECOVERY_TIMEOUT related settings of the Elimination function(s) before the POF, the flow characteristics (e.g., inter frame/packet time), and the delay difference of the paths used by the flow.

Design of these parameters is out-of-scope in this document.

Note: multiple network failures may 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. Packets being in order 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.

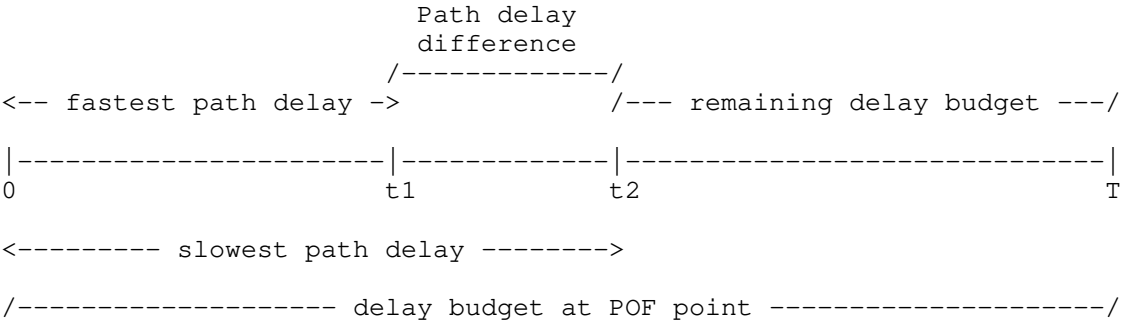


Figure 3: Delay Budget Relations at the POF Point

4.4. The Advanced POF Algorithm

In network scenario 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 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 must be arrived via 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 frame, the POF algorithm can use this information to select what predefined time "POFMaxDelay_i" to apply for the buffered frame/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.

The method for identification of 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 in other means. Method for identification of the packet's path is out of scope in this document.

Note: in case of using the advanced POF algorithm it might be advantageous to combine PEF and POF locations in the DetNet network, as it can simplify the method used for identification of 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 may 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 was received - may not be corrected. Motivation behind such an initialization is POF implementation simplicity.

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 buffer is selected, 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 may impact the result of POF (e.g., packet received out of the expected worst-case delay window - calculated based on the path delay difference - may 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 re-ordering. 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 history window size.

5. Control and Management Plane Parameters for POF

POF algorithms needs setting of the following parameters:

* Basic POF
  - "POFMaxDelay"
  - "POFTakeAnyTime"
* Advanced POF
  - "POFMaxDelay_i"
  - "POFTakeAnyTime"
  - Network path identification related configuration(s)

Note, that in a proper design "POFTakeAnyTime" must be 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 makes no IANA requests.

8. Acknowledgements

Authors extend their appreciation to Gyorgy Miklos for his insightful comments and productive discussion that helped to improve the document.

9. References

9.1. Normative References

[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>.

[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>.

[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>.

9.2. Informative References

[IEEE8021CB] IEEE, "IEEE Standard for Local and metropolitan area networks -- Frame Replication and Elimination for Reliability", 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>.

Authors' Addresses

Balázs Varga (editor)
Ericsson
Budapest
Magyar Tudosok krt. 11.
1117
Hungary

Email: balazs.a.varga@ericsson.com

János 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