DetNet
Internet Draft
Intended status: Informational
Expires: 5 July 2024

D. Guo
G. Wen
New H3C Technologies Co., Ltd
K. Yao
China Mobile
Q. Xiong
ZTE Corporation
G. Peng
Beijing University of Posts and Telecommunications
X. You
S. Zhu
New H3C Technologies Co.,Ltd

5 January 2024

Deterministic Networking (DetNet) Controller Plane - VPFC Planning Information Model Based on VPFP in Scaling Deterministic Networks

draft-guo-detnet-vpfc-planning-03.txt

Abstract

The cycle-based queuing and forwarding mechanisms are an effective method to ensure that the transmission has a definite upper bound of jitter, as well as latency. The prerequisite for this method is to ensure that queuing resources do not conflict. In scaling deterministic networks, how to avoid the queuing resources conflict of this method is an open problem. This document proposes the information model of planning virtual periodic forwarding channel (VPFC) based on virtual periodic forwarding path (VPFP). The solution can solve the queuing resources conflict of cycle-specified queuing and forwarding in nonlinear topology domain, ensuring the bounded latency of DetNet flow in the same periodic forwarding domain.

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 5, 2024.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction....3
1.1. Problem Statement....4
1.2. Document Roadmap....6
2. Terminology and Definitions....7
3. VPFP/VPFC and Configuration Data Models....7
3.1. VPFP: Virtual Periodic Forwarding Path....8
3.2. VPFC: Virtual Periodic Forwarding Channel....10
3.3. Configuration Data Model....11
3.3.1. Stack Model....11
3.3.2. Swap Model (informative)....13
4. Consideration for Resources Planning and Reservation Model....14
4.1. Theoretical Model....16
4.1.1. Measurement and Calibration....16
4.1.2. Mapping Function and Scheduling Cycle Conflict Resolution....18
4.1.3. Proposed Resources Planning Scheme....21
4.2. Detailed Description of resources Reservation Scheme....22
4.2.1. Establish New resources Metrics....22

4.2.2. Resources Reservation Corresponding to the Scheduling Cycle.....24
4.2.3. Resources of Scheduling Cycle Description.....26
4.2.4. Mapping Function.....27
4.2.5. Resources Demand.....28
4.2.6. Resources Reservation Process.....30
4.2.7. Resources Reservation Results.....30
5. Examples of Resources-Related Processing.....32
5.1. Collection Process of Cycle Resources.....32
5.2. Process Flow of Reserving Cycle Resources.....33
5.2.1. Reservation Calculation for Resources with Specified Cycle.....34
5.2.2. Reservation Calculation for Resources with Non-Specified Cycle.....35
5.2.3. Execution of Cycle Resources Reservation.....38
5.2.4. Resources Reservation for PREOF.....38
5.2.5. Bandwidth Increase Procedure.....39
5.2.6. Reroute.....39
5.2.7. Reclaiming Reserved Resources.....40
6. Security Considerations.....40
7. IANA Considerations.....41
8. Acknowledgements.....41
9. Contributors.....41
10. References.....42
10.1. Normative References.....42
10.2. Informative References.....43

1. Introduction

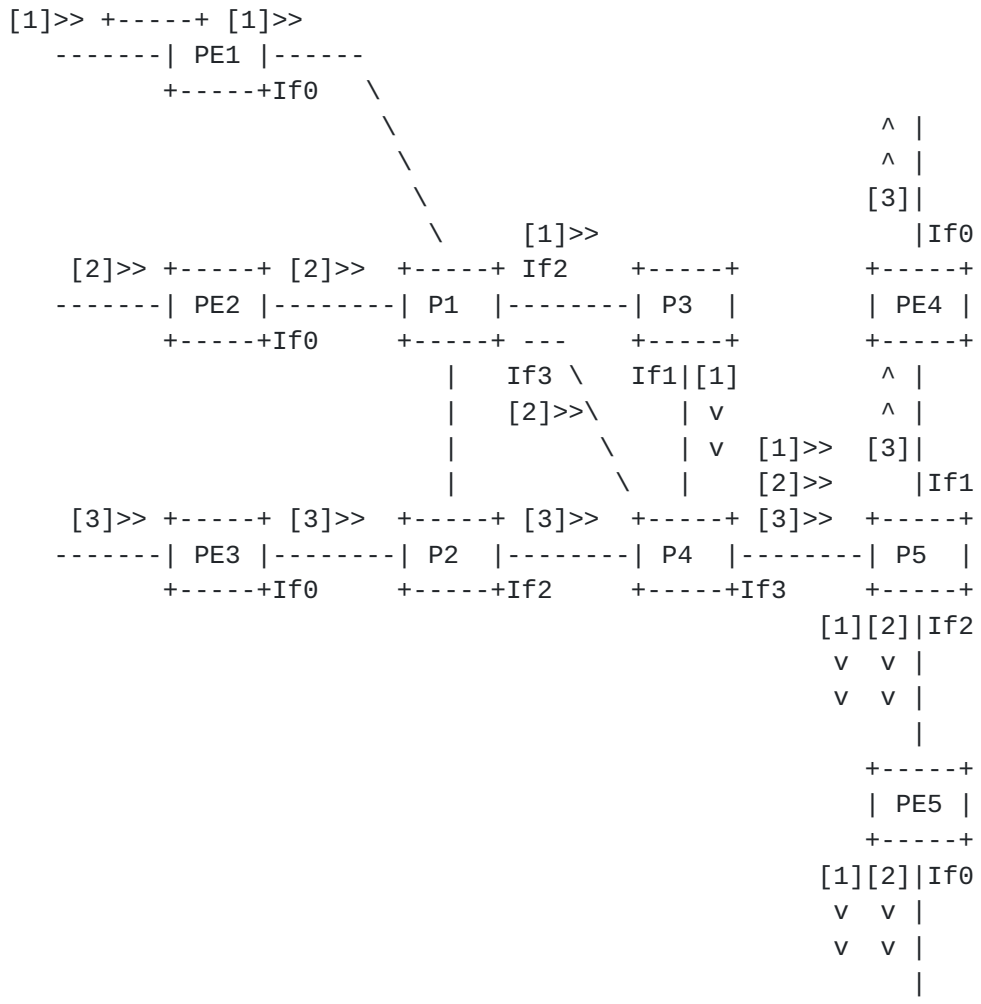
As described in [I-D.ietf-detnet-scaling-requirements], enhanced DetNet needs to support not only large amounts of flows and devices, but also large single-hop propagation latency and accommodate a variety of data plane queuing and forwarding mechanisms to carry App-flows with different levels of SLA. For some App-flows with strict requirements on delivery delay and delay variation (jitter), it is necessary to adopt a mechanism based on cyclic queuing and forwarding similar to CQF [IEEE802.1Qch]. These mechanisms are extended for WAN and make forwarding in DetNet transit nodes lightweight, without per-flow and per-hop state, and suitable for high-performance hardware implementation.

CSQF [I-D.chen-detnet-sr-based-bounded-latency] and TCQF [I-D.eckert-detnet-tcqf] propose different methods based on CQF extensions for cycle-based queuing and forwarding mechanisms. For multi-hop forwarding with these methods, the delay variation (jitter) does not exceed the length of 2 cycles. TQF [I-D.peng-detnet-packet-timeslot-

mechanism] proposes timeslot queueing and forwarding mechanism. These methods are all based on periodic queuing and forwarding mechanisms. All of these methods need to work with some resource reservation control, but none of them gives how to realize the resource reservation scheme in detail. This document proposes a VPFC planning information model based on VPFP to meet this demand.

1.1. Problem Statement

The resource reservation method of queuing and forwarding with specified cycle is very different from the traditional resource reservation method. Traditional resources reservation methods, such as RSVP-TE [RFC3473], only consider bandwidth availability for best-effort flows, that is, the reserved bandwidth meets the Peak Data Rate (PDR) of the service flow at the macroscopic level, which do not take into account the injection time of the packets at the microscopic level. The result of applying these methods to the resource reservation of cyclic queuing and forwarding is that the bandwidth resources meet the transmission demand at the macroscopic level, but there may be no resources in a specific cycle. If this problem remains unsolved, the prerequisite of CSQF/TCQF bounded latency cannot be satisfied.



[N] : Flow N
>/^/v : Direction of flows, Left/Up/Down
IfN : Interface N of the Router

Figure 1 Multiple Flows Converging

The prerequisite of CSQF/TCQF is that the data corresponding to a cycle can be forwarded during the cycle. In a single path model, the prerequisite of CSQF/TCQF is easy to meet, but in reality, networks are all nonlinear topologies, and to ensure the prerequisite, more work needs to be done.

For example, as shown in Figure 1, three flows (or aggregation of multiple service flows) flow1, flow2, and flow3 are injected into PE1, PE2, and PE3 respectively, and are converged on P4 after being forwarded. On the macroscopic level, the converged traffic of flow1, flow2 and flow3 does not exceed the bandwidth of the outgoing interface of P4 (set to intf3).

In a certain scenario (which is definitely unavoidable in practical applications), the three DetNet flows arrive at P4 within the same cycle interval and need to be forwarded to P5 through intf3. Assume that all physical links have 100Gbps bandwidth and the cycle interval is planned to be 10us, so about 125,000 bytes of data can be transmitted within this interval. Also assume for each of the three flows, 125,000 bytes of data reaches P4 within 10us, but no data arrives within 990us after that. In the micro of 10us time interval, each flow rate reaches 100Gbps over a 10us time interval, but only 1Gbps per flow over a 1ms time interval. In this case, if the traffic arriving at P4 at the same time is scheduled in the same cycle of Intf3 of P4, a conflict will occur in this cycle (the deterministic data that arrives cannot be sent within the specified cycle, resulting in additional random queuing delay, thereby affecting the deterministic forwarding of the next node), and the theoretical prerequisite of CSQF cannot be guaranteed. Therefore, the theoretical upper boundary of end-to-end jitter of CSQF, which should be less than two cycles, cannot be achieved. Especially after multi-hop accumulation, the jitter will exceed the upper limit that the App-flow can tolerate.

For a small-scale deterministic network, the conflict in the domain is not very prominent, but in an LDN, multiple App-flows access to the deterministic domain from different edge devices, and the topology formed by the forwarding paths is nonlinear. After further consideration of factors such as time injection, different link bandwidths, etc., the situation becomes very complicated. For an LDN, this document presents a general scheme to avoid the conflict of resources in the domain for cycle-based queuing and forwarding.

Note: To simplify the description, CSQF is used in the following examples.

1.2. Document Roadmap

In the following chapters, Section 2 gives the definition of relevant terminology; Section 3 specifies VPFP, VPFC and their configuration data models in our proposed scheme in detail; Section 4 describes the resources planning and reservation model in detail, in which Section 4.1 describes the relevant principles, and Section 4.2.1 describes the resources planning and reservation scheme in detail on the basis of Section 4.1. The resources reservation process involved in it is detailed in Section 5 separately due to too much content. In Section 5, the detailed processing flow related to resources is given.

2. Terminology and Definitions

This document uses the terms defined as [RFC8655], [RFC8938], [I-D.ietf-detnet-controller-plane-framework] and [RFC9320]. Moreover, the following terms are used in this document:

MCPE
: Management/Control Plane Entity.

Forwarding Path
: The candidata path which is used to forward DetNet flows.

Virtual Periodic Forwarding Path (VPFP)
: A virtual forwarding path which is used to forward DetNet flows based on the cycle and the mapping relationship between cycles.

Virtual Periodic Forwarding Channel (VPFC)
: A forwarding channel established on VPFP.

NQA
: Network quality analyzer.

TWAMP
: Two-Way Active Measurement Protocol.

CSPF
: Constrained Shortest Path First.

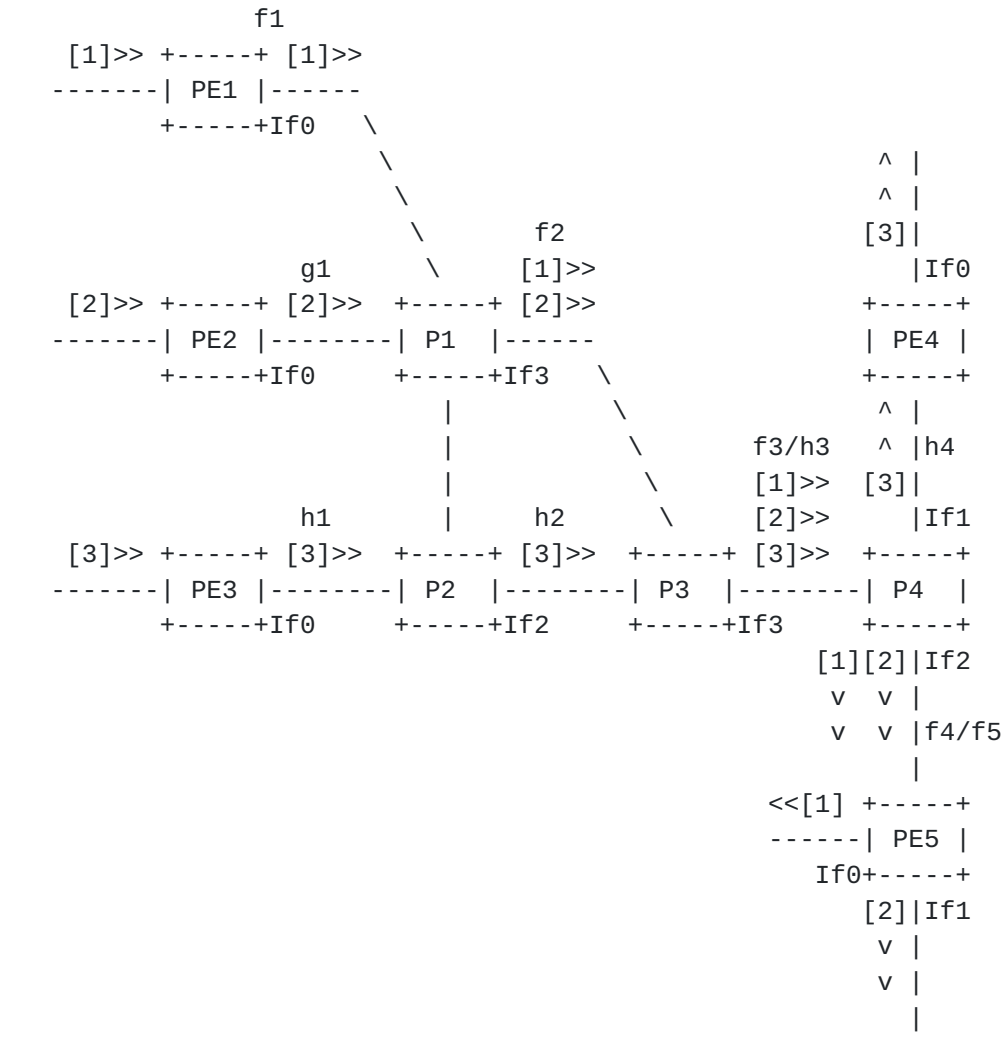
3. VPFP/VPFC and Configuration Data Models

As per [I-D.ietf-detnet-controller-plane-framework], in enhanced DetNet control plane, the cycle-based queuing and forwarding capabilities should be collected. When a DetNet flow being transmitted, for the cycle-based queuing mechanisms, the forwarding path needs to be calculated, the cycle-to-cycle relationship mapping function should be established, and cycle-based resources reservation should be completed.

This document proposes the Virtual Periodic Forwarding Path (VPFP) to indicate the forwarding path with the mapping function and the Virtual Periodic Forwarding Channel (VPFC) to indicate the forwarding channel within VPFP to transmit the DetNet flows.

This section specifies VPFP, VPFC and the configuration information models in details.

3.1. VPFP: Virtual Periodic Forwarding Path



[N] : Flow N
>/</^/v : Direction of flows, Left/Right/Up/Down
IfN : Interface N of the Router
fN/gN/hN : Mapping functions

Figure 2 VPFP: Virtual Periodic Forwarding Path

When there is a transmission requirement of a deterministic service flow, the forwarding path needs to be calculated in advance. Then add cycle-based queuing and forwarding capabilities, and establish cycle-to-cycle mapping relationships between adjacent nodes. We further abstract this mapping relationship as a function. After the mapping function is added to the forwarding path, a VPFP is formed.

Virtual Periodic Forwarding Path (VPFP):

A virtual forwarding path based on the cycles and the mapping functions between the cycles is called a VPFP. The mapping function is established between an outgoing interface scheduling cycle of an upstream node and an outgoing interface scheduling cycle of a downstream node, and the upstream node and the downstream must be adjacent nodes. The VPFP has the following characteristics:

* The outbound interface of each node in the forwarding path supports cycle-based forwarding;

* In each segment link of the path, there is a mapping relationship between the scheduling cycle of the outbound interface of the upstream node and the scheduling cycle of the outbound interface of the downstream node; The sending cycle on the upstream node and the mapping relationship together determine the forwarding cycle on the outbound interface of the downstream node.

VPFP has the following attributes:

* VPFPID (VPFP Identifier): an integer that uniquely identifies a VPFP within a deterministic periodic forwarding domain.

* Mapping Function: indicate the cycle-to-cycle relationship mapping.

* Interface ID: an integer that uniquely identifies an interface.

In Figure 2 as an example, the mapping relationship may be as following shown.

((PE1,Intf0), (P1,Intf3)): f1;

((P1,Intf3), (P3,Intf3)): f2;

((P3,intf3), (P4,intf2)): f3;

((P4,Intf2), (PE5,Intf0)): f4;

((P4,intf2), (PE5,Intf1)): f5;

((PE2,Intf0), (P1,intf3)): g1;

((PE3,intf0), (P2,intf2)): h1;

((P2,intf2), (P3,intf3)): h2;

((P3,intf3), (P4,intf1)): h3;

((P4,intf1), (PE4,Intf0)): h4.

The controller plane maintains the mapping function between the scheduling cycles of the outgoing interfaces of each pair of adjacent nodes. This function can be unique or multiple. Once the path is determined, the function is also determined. When the resources reservation fails, the physical path carrying the VPFP can be changed, or the mapping function in the VPFP can be changed to calculate the reserved resources again (TBD).

The forwarding path carrying the VPFP is generated by the MCPE or network administrator after calculating the path, and the mapping function is generated by the calibration after measurement (Section 4.1.1). As shown in Figure 2, assuming that there are three deterministic flow paths, the VPFPs form are:

VPFP1: (PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0)

VPFP2: (PE2,Intf0) g1 (P1,intf3) f2 (P3,intf3) f3 (P4,intf2) f5 (PE5,Intf1)

VPFP3: (PE3,intf0) h1 (P2,intf2) h2 (P3,intf3) h3 (P4,intf1) h4 (PE4,Intf0)

Where f1~5, g1, h1~3 are injective functions, see Section 4.1.2 for details.

3.2. VPFC: Virtual Periodic Forwarding Channel

In scaling networks, multiple intersecting VPFPs form a mesh topology. In order to meet the transmission requirements of a specific DetNet flow, one or more VPFCs needs to be planned on one or more VPFPs with cycle-based resources reservation.

Virtual Periodic Forwarding Channel (VPFC): a forwarding channel established within VPFP. The basic attributes of a VPFC are:

* VPFCID (VPFC Identifier). VPFCID is an integer that uniquely identifies a VPFC within a VPFP.

* VPFPID. VPFP is the path that carries the VPFC, see Section 3.1 for detail;

* Cycle Info. Cycle Info contains the scheduling cycle and the allocated resources corresponding to the scheduling cycle, describes the bandwidth and periodicity characteristics of the VPFC, and is the result of resources reservation. For details, see Section 4.2.7.

A forwarding path can carry multiple VPFPs. When all the mapping relationships along the path are determined, a unique VPFP is determined. Multiple VPFCs can be established in one VPFP.

3.3. Configuration Data Model

As per [RFC9016], information models should be provided for DetNet control plane and configuration information models can be used between the management/control plane entity of the network and the network nodes. As shown in Figure 3, this document proposes a configuration information model based on VPFP/VPFC planning for enhanced DetNet control plane to describe the settings required on network nodes to provide deterministic service to a data flow in scaling networks.

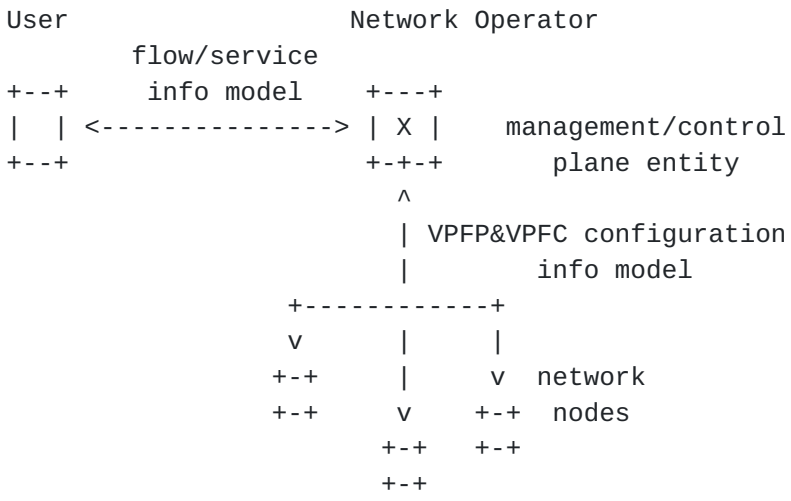


Figure 3 VPFP & VPFC configuration data model

If the cycle mapping mode is stack mode, the VPFP parameters should be deployed to the ingress node of the VPFP to generate the information for directing forwarding. If the cycle mapping mode is swap mode, VPFP related information needs to be deployed to each node of VPFP, including Ingress PE, P, and Egress PE. In both modes, the VPFC parameters should be deployed to the head node of the VPFP. The specific process is beyond the scope of this document.

3.3.1. Stack Model

In stack mode as an example, assuming the VPFP for a DetNet flow is:

VPFP1 = (PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0).

Assuming the successfully reserved result list on PE1 is:

{

(VPFP1): (intf0,Cycle0,1),

(VPFP1): (intf0,Cycle1,1),

(VPFP1): (intf0,Cycle2,1),

(VPFP1): (intf0,Cycle3,1),

(VPFP1): (intf0,Cycle4,1),

(VPFP1): (intf0,Cycle5,1),

(VPFP1): (intf0,Cycle6,1),

(VPFP1): (intf0,Cycle7,1),

}

The VPFC consists of a VPFP and resources reserved along the VPFP. When the MCPE deploys the VFPC to the head node of the VPFP, the parameters that need to be configured are summarized as below.

```
+-- uint16 vpfcid # Virtual Periodic Forwarding Channel Identifier
+-- uint16 vpfpid # virtual periodic forwarding path identifier
+-- if_config[oif] # Outgoing InterFace configuration
    +-- uint16 cycles # Number of cycles involved in resources
                      # reservation
    +-- cycleinfo[0..cycles-1] #Cycle Info
        +--uint16 cycleid #Cycle ID
        +--uint16 res #Number of Resources
```

Figure 4 VPFC configuration data structure

Note: A vpfcid uniquely identifies a VPFC, and a vpfpid uniquely identifies a VPFP. For PREOF, a returned result list may create multiple VPFCs, each corresponding to a VPFP in PREOF paths.

```
   +-- uint16 vpfpid # virtual periodic forwarding path identifier
   +-- uint8 cycles # is the number of cycles used across all
                    # interfaces in the CSQF/TCQF domain.
   +-- policy_info [policy] # Policy information, e.g. SRv6 policy
   +-- pipe_info[0..cycles-1] # The scheduling cycle pipeline
                              # corresponding to each scheduling cycle
                              # on the head node
       +-- uint8 hops # Number of hops
       +-- map_info[0..hops-1] # The mapping target in each pipeline
                               # is a specific scheduling cycle
            +--uint8 out_cycle #output cycle
```

Figure 5 VPFP configuration data structure for stack mode

In the head node (e.g., Ingress PE) of the VPFP, the forwarding information is generated based on the VPFP configuration, which is used for cyclic queueing and forwarding, as well as packet encapsulating in the CSQF domain. At the same time, according to the configuration information of VPFC, the selection of the scheduling cycle in the head node is strictly stipulated, which is used to realize the PSPF function similar to [IEEE802.1Qci].

With these configuration data models, the creation, deletion, and modification operations of VPFP and VPFC can be achieved.

3.3.2. Swap Model (informative)

The VPFC configuration data model in the swap mode is consistent with that in the stack mode, and only the HEAD node of VPFP needs to be configured. For the swap mode, the VPFP information needs to be configured for each non-HEAD node.

```
   vpfp
+-- uint16 vpfpid          # virtual periodic forwarding path identifier
+-- uint8  cycles          # is the number of cycles used across all
                           # interfaces in the CSQF/TCQF domain.
   +-- map_info            # The mapping info
   +-- uint8  iif          # input interface
   +-- uint8  oif          # output interface
    +-- up_cycles_info[0..cycles-1]  # The output cycle of upstream
              # node map to output cycle of the current node one by one
      +--uint8 out_cycle         #output cycle
```

Figure 6 VPFP configuration data structure for swap mode

Configure the VPFP information to each node that VPFP passes through, which can form forwarding information at each node to guide the queuing and forwarding of DetNet flows at this node.

As mentioned before, this planning scheme is applicable to resource planning based on cyclic queuing, and these queuing methods also include TCQF. TCQF's cycle mapping mode is a typical SWAP mode. TCQF has its configuration data model, but this configuration data model is also applicable. In order to use this configuration model, it is necessary to add the conversion configuration of the period information and Tag information in TCQF, which is related to specific implementation and beyond this solution.

4. Consideration for Resources Planning and Reservation Model

The establishment of periodic forwarding resources system is a complex system engineering. It is more realistic to establish the system based on the existing best-effort system. The whole process needs the cooperation of user plane, management/control plane and data plane. To show the overall framework of resources reservation, the content shown in Figure 7 is copied from [RFC9016]. The management/control plane entity (MCPE) is responsible for the managing, planning, reserving, and recycling of cyclic forwarding resources for deterministic service flows. To get a sense of the whole picture, the main resources planning related processes are listed as follows:

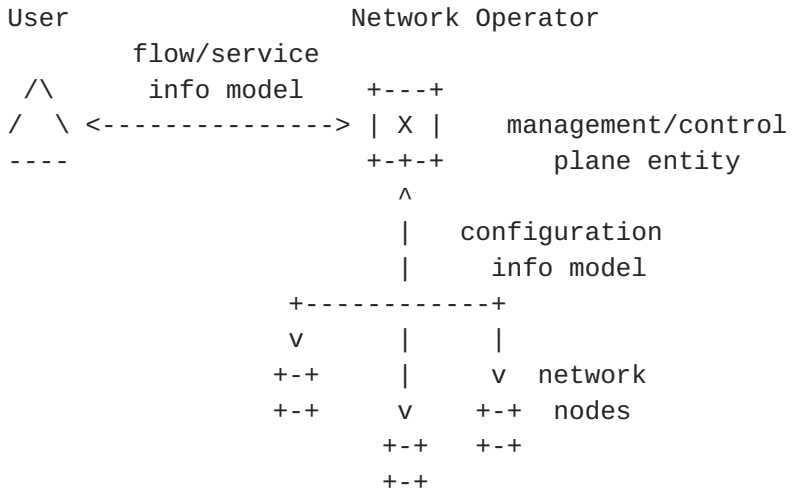


Figure 7 VPFP configuration data structure Usage of Information Models (Flow, Service, and Configuration)

1. The data plane generates topology information. IGP (OSPF or IS-IS) collects the network topology information. Using the powerful route

selection and calculation capabilities of BGP protocol, BGP-LS protocol summarizes the topology information discovered by IGP protocol and sends it to the MCPE (management/control plane entity, see Figure 7). The MCPE stores this information in the topology database of the control plane.

2. MCPE measures the transmission delay between nodes through NQA or TWAMP. This delay is a relatively coarse granularity in accuracy, usually in milliseconds.

3. MCPE obtains the link transmission delay measurement results through NETCONF [RFC6241]/YANG [RFC6020] and uses the results with less accurate delay info to update the topology data.

4. User (see Figure 7) provides flow information required to establish a session (see [RFC9016] for specific parameters)

5. MCPE uses CSPF to calculate the end-to-end path to obtain an optimal path, or multiple paths with close propagation delays for PREOF.

6. The MCPE performs accurate segmentation measurement on the forwarding path (the measurement mechanism with microsecond-level accuracy needs to be solved). According to the measurement results and the resident delay in the node, the correlation mapping is established to form VPFP. VPFP is delivered to the data plane and integrated into the forwarding table to direct data forwarding.

7. MCPE uses the resources planning scheme provided in this document to reserve resources (see Section 4 and Section 5 for details). After the planning is successful, the result forms VPFC.

8. MCPE delivers the planned resources to the head node (e.g., Ingress PE) of VPFP, and creates VPFC. (The VPFP&VPFC configuration data models are defined in Section 3.3)

9. The head node of VPFP (network node in Figure 7, Ingress PE) schedules the data of the DetNet flow according to the cycle resources owned by the VPFC to which the DetNet flow belongs.

As stated in [RFC8557], whether a distributed alternative without a PCE can be valuable could be studied. For example, such an alternative could be to build a solution similar to that in [RFC3209]. But the focus of current work on DetNet should be to provide a centralized method first. The solution provided in this document belongs to the centralized solution, and can make the implementation of resources reservation by data plane devices as lightweight and stateless as possible.

In the following contents of this chapter, Section 4.1 first describes the basic principles, in which the measurement and calibration in Section 4.1.1 are the prerequisite for establishing the function mapping of VPFP. Section 4.1.2 describes how to use the characteristics of mapping functions to resolve scheduling cycle planning conflicts; combined with the theory in Section 4.1.2, Section 4.1.3 briefly introduces the overall process of resources planning. Based on the principles of Section 4.1, Section 4.2 describes the complete scheme of resources reservation in detail, including various data models involved in the scheme. Due to too much content, the resources reservation process involved is elaborated separately in Section 5.

4.1. Theoretical Model

4.1.1. Measurement and Calibration

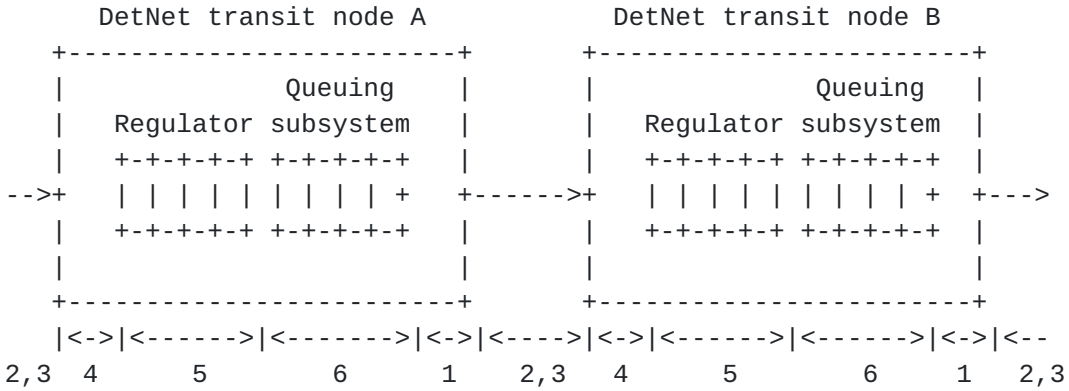
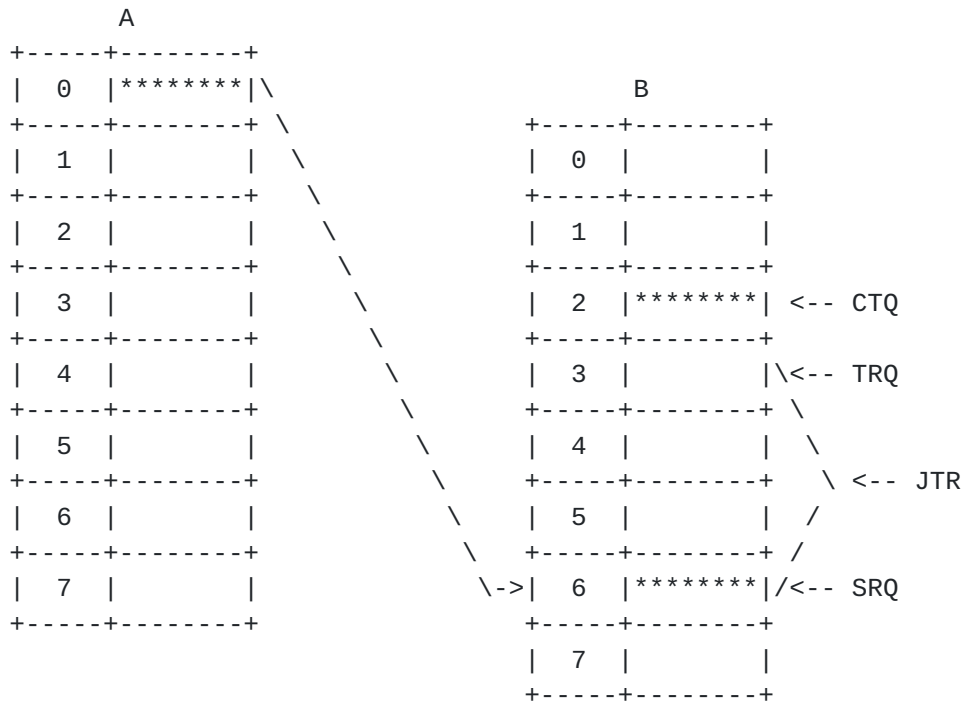


Figure 8 Timing model for DetNet or TSN

As shown in Figure 8, [RFC9320] highly abstracts the timing model of the DetNet transit node. In a large-scale deterministic network, the implementation of some DetNet transit nodes is a distributed architecture, and the processing delay in these nodes varies widely, which is the operation that contributes the most to the delay jitter. In cycle-based queuing and forwarding, the jitter introduced by various operations needs to be fully considered, so that the end-to-end transmission delay can reach a definite bound.



CTQ : Current transmitting queue when the packet arrives
TRQ : Theoretical receiving queue for the packet
SRQ : Safe receiving queue for the packet
JTR : Jitter of the node

Figure 9 Mapping relationship between scheduling cycles of outbound interfaces of upstream and downstream nodes

Taking CSQF as an example, before forwarding, it is necessary to establish a mapping relationship between the scheduling cycles of the outgoing interfaces of upstream and downstream nodes, and the process is completed by measurement and calibration. (In order to simplify the description, the specific interface of the Node A is uniformly replaced by the Node A, and similar for Node B.)

As shown in Figure 9, taking 8 cycles as an example, Node A and Node B are two adjacent CSQF nodes. Node A is the upstream node, and Node B is the downstream node. To know which cycle is being scheduled in Node B (for example, cycle 2 in Figure 9) when packets sent from a certain cycle in Node A (for example, cycle 0 in Figure 9) reaches it, a measurement should be applied. The specific implementation of the measurement is beyond the scope of this demo, and will not be described here. After the measurement is done, the forwarding cycle in Node B for these packets should be decided, which should take into account the processing delay variant in the device. In this example, for packets sent in cycle 0 of Node A, cycle 6 is chosen as

forwarding cycle in Node B. The packets sent in cycle 1 of Node A are assigned to cycle 7 of Node B, and so on. This is the task to be done by calibration.

After calibration, the scheduling cycles of A and that of B have the following mapping relationship:

0 --> 6

1 --> 7

2 --> 0

3 --> 1

4 --> 2

5 --> 3

6 --> 4

7 --> 5.

In terms of jitter absorption in Figure 9, it is feasible to assign the packet sent in the 0th scheduling cycle of Node A to the queue in the 6th or 7th or 0th scheduling cycle of Node B. So there is more than one mapping that can be calibrated.

4.1.2. Mapping Function and Scheduling Cycle Conflict Resolution

As described in Section 4.1.1, after the calibration is completed, a definite mapping relationship is established between the scheduling cycles of the outbound interface of the two adjacent nodes. This relationship can be regarded as a function f: its domain is the scheduling cycle range of Node A, 0~7, and its range is the scheduling cycle range of Node B, 0~7. Further constraint can be imposed on the mapping relationship: during calibration, any scheduling cycle in A has one and only one scheduling cycle in B which is calibrated with it. Under this constraint, the function f becomes an injective function.

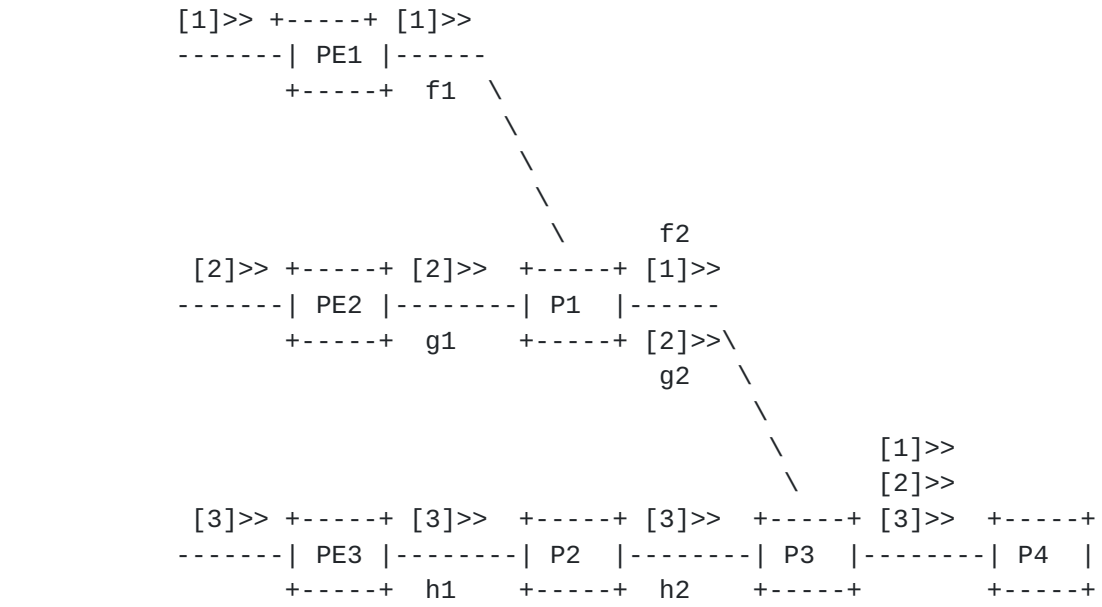


Figure 10 Schematic diagram of multiple flows convergence with cycle mapping function

The cycle planning is further constrained: in the same CSQF domain, all interface plan the same number of scheduling cycles. Under this constraint, all mapping functions have the same domain and range.

Note: Different interfaces of the same node can belong to different domains, and the cross-domain processing is beyond the scope of this document.

It is assumed that the calibrated mapping between the scheduling cycles of the outbound interfaces of the upstream and downstream nodes described in Figure 10 also satisfies the injective function relationship, and the calibrated relationship between the scheduling cycles of the outbound interfaces of PE1 and P1 is the functional relationship f1, and the calibrated mapping relationship between the scheduling cycles of the outbound interfaces of P1 and P3 is the functional relationship f2. The mapping relationship between the PE and the scheduling cycles of the P3 outbound interface satisfies the composite function f:

$$f = f2(f1)$$

According to the property of injective function, f is also an injective function.

Similarly, we can get:

g = g2(g1)

h = h2(h1)

and g, h are both injective functions, where g2 is the mapping relationship between the scheduling cycle of outbound interface of P1 and the scheduling cycle of outbound interface of P3, and h2 is the mapping relationship between the scheduling cycle of outbound interface of P2 and the scheduling cycle of outbound interface of P3.

With the above constraints, assume the flow from PE1, PE2, and PE3 conflicts at P3, that is, they are mapped to the same scheduling cycle. Let the scheduling cycles of flow in PE1, PE2, and PE3 be a, b, and c respectively, the following situation occurs:

The function values of f(a), g(b) and h(c) appear the same, which is a conflict. According to the property of the injective function, if a is changed to d (d!= a), then f(d) != f(a), so f(d) != g(b) and f(d) != h(c). Similarly, changing the input of function g or h can also achieve the effect of eliminating conflict.

At the same time, it is easy to draw the following conclusions:

For f = f2(f1), if d != a, then f(d) != f(a) and f1(d) != f1(a).

Further generalize this conclusion: Suppose that the ordered set <f1, f2, ..., fn> is a set composed of injective functions (where n is a natural number), and the domain and range are both A, A={x|0<=x<k , x and k are natural numbers}. The composite function composed of the first i (1<=i<=n) functions is denoted as:

f[i] = fi(fi-1(...(f1))).

Let the proper subsets B and C of set A satisfy the condition:

The union of B and C is A, the intersection of B and C is empty set, then for any b in B and any c in C, f[i](b) != f[i](c).

When planning the usage of scheduling cycles, the scheduling cycles that have been traversed in the scheduling cycles of the head node of VPFP (e.g., Ingress PE) are regarded as set B, and the scheduling cycles that have not been traversed are regarded as set C. When a conflict for a scheduling cycle occurs when converging with other

paths, a new scheduling cycle c is selected from C, and the new c and the set elements that have been traversed in set B as input produce different results. That is, there will be no cycle planning conflicts starting from the same head node along the currently planning VPFP. In this way, it is only necessary to judge whether there is a conflict with other path aggregation, and if there is no conflict, the scheduling cycle planning is successful.

Note: When there are multiple mapping relationships that can be calibrated, each calibrated relationship corresponds to a function. In multiple planning, different function mappings can be used each time.

4.1.3. Proposed Resources Planning Scheme

Under the prerequisite of rational utilization of resources, the key issue to ensure that the theoretical conditions of CSQF are satisfied is to plan a VPFC to be scheduled during a certain cycle of the interface of the corresponding node. In other words, the forwarding capability corresponding to the specified cycle interval on the interface of the corresponding node is allocated to the VPFC. The common feature of cycle-based forwarding is that all data is first buffered and then forwarded in a specific cycle interval. Combined with this feature, the more abstract forwarding capability during a cycle can be converted into the cache resources required for the specific data that can be forwarded during this cycle. The problem is transformed into a buffer resources reservation problem, that is, the buffer resources is reserved for the deterministic flows that are allowed to be scheduled during the cycle (referred to as resources reservation for cyclic forwarding), and the deterministic flows that do not have buffer resources reserved are not scheduled during the cycle.

According to the conclusion in Section 4.1.2, the resources in the CSQF domain can be reasonably planned. When a scheduling cycle conflict occurs on the convergence point or the outbound interface with small bandwidth and the traffic entering from other paths, change the planning cycle of the head node, then perform cycle calculation along the VPFP and try to reserve resources. After the attempt is successful, the conflict can be eliminated.

At the same time, because the mapping functions along the VPFP are injection functions, we can regard the scheduling cycle and buffering resources of the non-head node's aggregation interface or low-bandwidth outbound interface to be shared as a common resource, and allocate these common resources to the head nodes with deterministic transmission requirements. The resources allocated on the head node and the VPFP constitute the VPFC of our scheme.

The injective function also strictly constrains the corresponding relationship between the head node and convergent node resources. Therefore, the head node only needs to save the allocated resources of its own node, and does not need to save the allocated resources of the non-head node (including sink node). The non-head node does not need to save the resources allocation state, and the resources allocation state is saved by the MCPE, so as to achieve the lightweight implementation of the resources reservation of the non-head node (e.g., P node).

While the head node of VPFP performs VPFC scheduling strictly according to the resources allocated to the VPFC, conflicts can be avoided on the non-head node's aggregation interface or low-bandwidth outbound interface sharing transmission. Scheduling strictly according to allocated resources on the head node is a key issue, which will be further studied in other literatures.

According to [RFC8655], service flows can be aggregated and resources can be reserved for the aggregated flows. With our solution, the aggregated flows share the scheduled resources reserved on the edge nodes, and the resources competition is localized. The delay jitter caused by the aggregated flows will also be local, and it is easier to realize the time delay bounded. In order to further optimize the jitter of the member service flows in the aggregate flow, only the scheduling resources allocated to the edge nodes of the aggregate flow need to be further refined, and this arrangement will not cause the change of global resources allocation.

By separating resources planning from measurement and calibration, there is no need to consider the problem of path aggregation when performing calibration after measurement, which can greatly reduce the complexity of calibration.

4.2. Detailed Description of resources Reservation Scheme

In Section 4.1, we give a highly abstract description of the principle and method of our resources reservation scheme, and give a very high-level idea. This section discusses the scheme in detail, including quantitative representation of forwarding resources corresponding to the scheduling cycle, and description of the main elements involved in the scheme. The detailed process of resources reservation in this scheme is described in Section 5.

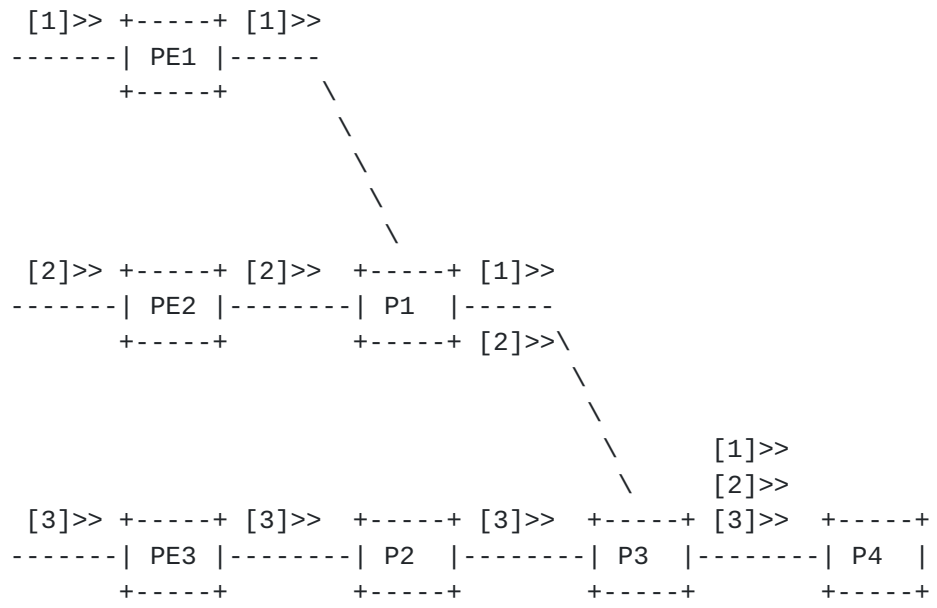
4.2.1. Establish New resources Metrics

Forwarding resources are a relatively vague concept. They include not only bandwidth resources, but also device storage resources. For example, high-speed on-chip caches inside ASICs are often measured in

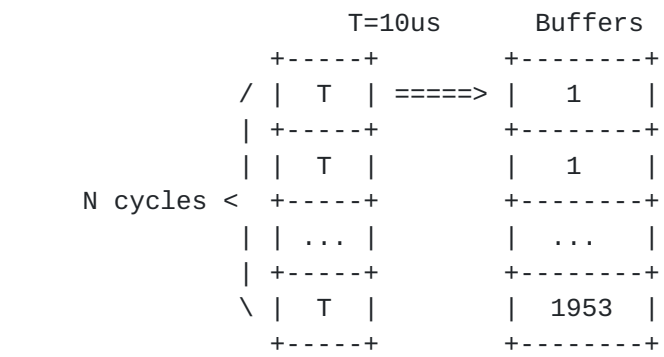
units of bytes rather than bps. It is not enough to consider bandwidth only when reserving resources, we have to establish a new resources metric in bytes or bytes of a certain length, for example, 64 bytes is one resources unit. The comprehensive capability of one cycle is measured in new resources units, which covers cache capacity, cycle interval time, and interface bandwidth. In this way, the three dimensions of cycle duration, cache capacity, and physical bandwidth are simplified into one dimension: the number of resources units, so as to simplify the implementation of resources reservation.

For example, assuming that the backbone network is uniformly divided into a 10us scheduling cycle and one resources unit has 64 bytes, the data that can be transmitted on a 400G interface in each scheduling cycle is about 7812 resources units, 1953 on a 100G interface, 195 on a 10G interface, and 19 on a 1G interface. The amount of resources that can be provided by the scheduling cycle of an interface is given by the comprehensive evaluation of the implementation specifications of devices such as interface bandwidth and storage resources. Resources planning is done based on this quantity, which simplifies implementation.

4.2.2. Resources Reservation Corresponding to the Scheduling Cycle



Resourc of the interfaces:



[N]: Flow N
> : Direction of flows

Figure 11 Scheduling cycle resources of the interface

This section extends the description of the method described in Section 4.1 in conjunction with the constraints of Section 4.2.1. According to the theory in Section 4.1, if the scheduling cycle a of PE1 is mapped to the scheduling cycle c of P1, and the scheduling cycle b of PE2 is also mapped to the scheduling cycle c of P1, then a cycle conflict occurs. The conflict can be resolved by adjusting the scheduling cycle a or b, but if P1 has multiple units of resources in the same scheduling cycle, it is allowed to allocate resources in the

same scheduling cycle to PE1 and PE2 as needed, thereby increasing the utilization of resources in the scheduling cycle.

Based on the resources metrics proposed in Section 4.2.1, the resources of the scheduling cycle are uniformly quantified. As shown in Figure 11, assuming the data rates of all physical links are 100Gbps, and the length of the scheduling cycle is 10us, each scheduling cycle can transmit 1953 64-byte data resources units. Because of multiple resources belonging to one cycle, instead of explicitly judging whether a scheduling cycle conflict occurs, it is changed to judge whether the resources corresponding to the scheduling cycle on the path meet the demand. If the demand for resources can be satisfied, the corresponding resources are reserved. Otherwise, the MCPE chooses another scheduling cycle as the input of the function, and tries iterative calculation and resources reservation again.

According to the resources demand of a DetNet flow, starting from the head node of the VPFP, the functions in the path are sequentially called along the VPFP for calculation. Firstly, the first value in the domain of the function associated with the head node, is used as the input of that function, and then the previous function's value is used as the input value of the next function for iterative calculation. Concomitantly, it is judged whether the resources of the cycle corresponding to each input value and output value meet the demand. If the resources do not meet the demand, the current iterative calculation is terminated, and the next value in the domain of the function associated with the head node is used as a new input, and the above processing is continued. When all the values in the domain of the function of the head node are traversed and fail to meet the demand, no VPFC is successfully planned for the DetNet flow. If the resources meet the demand, a VPFC is successfully planned, and the remaining resources corresponding to the scheduling cycle are updated along the planned VPFC. The controller plane records the VPFC, and delivers the VPFC to the head node of the VPFP.

To optimize resources usage, multiple DetNet flows can be aggregated together to share a VPFC. For example, suppose the cycle duration is 10us and 10 cycles are used, then each cycle will be scheduled once every 100us. If 1 unit of resources is allocated to a VPFC for a DetNet flow which sends 1 unit of data every 1ms, then only 1/10 of the allocated resources is really used. If another DetNet flow regularly sends 1 unit of data every 3ms, which has the same forwarding path with the VPFC, then this VPFC can be shared. Of course, this inevitably introduces jitter, but this jitter is bounded, unperceived or tolerated by most applications, and can be eliminated by other methods, which are beyond the scope of this document.

4.2.3. Resources of Scheduling Cycle Description

Scheduling cycle resources description:

(node, interface, scheduling cycle): (number of available resources, number of initial resources), where the number of resources is measured in uniform resources units. For example, in Figure 11, assuming that the total number of cycles in the domain is uniformed as 8, the cycle interval is 10us, the PE1's Intf0 is a 10Gbps interface, and each cycle can forward about 193 units of resources. For factors that have not been considered, some capabilities need to be reserved, the number of resources can be initialized as 180. The resources numbers of PE1's Intf0 are initialized as follows:

(PE1,Intf0,Cycle0): (180,180);

(PE1,Intf0,Cycle1): (180,180);

(PE1,Intf0,Cycle2): (180,180);

(PE1,Intf0,Cycle3): (180,180);

(PE1,Intf0,Cycle4): (180,180);

(PE1,Intf0,Cycle5): (180,180);

(PE1,Intf0,Cycle6): (180,180);

(PE1,Intf0,Cycle7): (180,180);

The P1's Intf3 is a 100Gbps interface, and each cycle can forward about 1953 units of resources. For factors that have not been considered, some capabilities need to be reserved, the number of resources can be initialized as 1900. The resources numbers of P1's Intf3 are initialized as follows:

(P1,Intf3,Cycle0): (1900, 1900);

(P1,Intf3,Cycle1): (1900, 1900);

(P1,Intf3,Cycle2): (1900, 1900);

(P1,Intf3,Cycle3): (1900, 1900);

(P1,Intf3,Cycle4): (1900, 1900);

(P1,Intf3,Cycle5): (1900, 1900);

(P1,Intf3,Cycle6): (1900, 1900);

(P1,Intf3,Cycle7): (1900, 1900);

The controller plane maintains the resources information of the scheduling cycle of the interface of each node, and reduces the number of available resources in the corresponding scheduling cycle of the interface of the corresponding node after the DetNet flow path resources reservation is performed.

4.2.4. Mapping Function

Assuming that each node is divided into n equal-length scheduling cycles, after measurement and calibration, there are n function mapping relationships:

y = (x+k) mod n, the domain of definition is {x | 0<=x<n, x and n are natural numbers}, and the value range of the constant k is: {k | 0<=k<n, k and n are natural numbers, n>3}, (In principle, the scheduling cycle and the number of corresponding CSQF queues can be less than 3, but in large-scale deterministic networks, it is unrealistic to be less than or equal to 3, and it is not conducive to resources planning).

Taking the network in Figure 10 as an example, it is assumed that each node is divided into 8 (n=8) equal-length scheduling cycles. After measurement, the function mapping relationships such as f1~2, g1~2, and h1~2 are obtained by calibration as one of the following 8 functions:

y = (x+0) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+1) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+2) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+3) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+4) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+5) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+6) mod 8, {x | 0<=x<8, x is a natural number};

y = (x+7) mod 8, {x | 0<=x<8, x is a natural number};

As a specific example, these functions can be finally decided as:

f1(x) = (x+3) mod 8, {x | 0<=x<8, x is a natural number};

f2(x) = (x+1) mod 8, {x | 0<=x<8, x is a natural number};

g1(x) = (x+4) mod 8, {x | 0<=x<8, x is a natural number};

g2(x) = (x+5) mod 8, {x | 0<=x<8, x is a natural number};

h1(x) = (x+6) mod 8, {x | 0<=x<8, x is a natural number};

h2(x) = (x+7) mod 8, {x | 0<=x<8, x is a natural number};

The controller plane saves the mapping function which is one of the components used to describe VPFP.

4.2.5. Resources Demand

The resources demand here is the input for reservation processing after conversion processing of app-flow's requirements (see [RFC9016]), not the original transmission requirement of an app-flow. Resources demands are described as a list of requirements:

{sub-demand 1, sub-demand 2, ...}

Each sub-demand looks like:

(Virtual Periodic Forwarding Path): (Outbound Interface, Scheduling Cycle, Number of Resources Units Required, Minimum Allocation Granularity Per Cycle).

The components of the above requirements are described as follows:

* "Virtual Periodic Forwarding Path" is used to specify the virtual periodic forwarding path for allocating resources;

* "Outbound Interface" is the outgoing interface of the first node in the virtual periodic forwarding path;

* "Scheduling Cycle" is used to specify which cycle to be selected in the first node to send data. The resources of the cycle need to be allocated. The scheduling cycle may not be specified, indicating that any cycle can be used. Otherwise, the resources are allocated according to the specified cycle;

* "Number of Resources Units Required" is used to specify the number of resources required by the resources requirement;

* "Minimum Allocation Granularity per Cycle" specifies the minimum number of resources allocated in the same scheduling cycle to ensure that the same data packet of a deterministic service flow does not cross the scheduling cycles. For example, if each packet transmission of a service flow requires 2 resources units, then the "Minimum Allocation Granularity per Cycle" will be 2, means at least 2 resources units needs to be allocated from the same scheduling cycle.

For the transmission requirements of app-flows with strict jitter upper bound requirements, resources for a specified cycle may be allocated. For example, a certain DetNet flow has a transmission requirement of one resources unit, but it is required to be transmitted in time, and the jitter is less than 2 scheduling cycles. CSQF can meet this requirement. A unit of resources is allocated from each cycle along the VPFP, so that no matter when the data of the service flow arrives, there is always a unit of resources is ready.

In order to facilitate the following description, the demand list is further symbolized, and the resources demand is described as the demand list DemandList:

{SubDemand1, SubDemand2, ...}

SubDemand for each specified cycle is in the form of:

(VPFP): (oif,cycle,res,min);

That is, SubDemand1 is set to "(VPFP): (oif1, cycle1, res1, min1)".

The allocation of sub-requirements for each non-specified cycle is as follows:

(VPFP): (oif, InvalidCycle, res, min);

That is, SubDemand1 is set to "(VPFP): (oif1, InvalidCycle, res1, min1)".

where VPFP is described in Section 3.1, for example, VPFP is:

(PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0)

For example, for a flow, its path is the above VPFP, and for its specified cycle allocation, its resources demand list DemandList on PE1 can be expressed as:

{

(VPFP): (intf0,Cycle0,1,1),

(VPFP): (intf0,Cycle1,1,1),

(VPFP): (intf0,Cycle2,1,1),

(VPFP): (intf0,Cycle3,1,1),

(VPFP): (intf0,Cycle4,1,1),

(VPFP): (intf0,Cycle5,1,1),

(VPFP): (intf0,Cycle6,1,1),

(VPFP): (intf0,Cycle7,1,1)

}

The above allocation indicates that resources of 0 to 7 cycles are allocated to the flow, and one unit of resources is allocated from each cycle; and if the resources list DemandList allocated by the scheduling cycle is not specified, it can be expressed as:

{(VPFP): (intf0, InvalidCycle, 10, 2)}, where InvalidCycle is the invalid cycle, defined by the implementation.

4.2.6. Resources Reservation Process

For details, see Section 5.

4.2.7. Resources Reservation Results

The resources allocation result of the specified scheduling cycle is exactly the same as the resources allocation result of the non-specified cycle, and it is described as a result list:

{subresult 1, subresult 2, ...}

Each sub-result looks like:

(path information): (outbound interface, scheduling cycle, number of resources units).

In order to facilitate the following description, the resources allocation result list is further symbolized, and the resources allocation result is the result list ResultList:

{SubResult1,SubResult2, ...}

SubResult of each specified cycle sub-requirement is as follows:

(VPFP): (oif, cycle, res);

That is, SubResult1 is set to (VPFP): (oif1, cycle1, res1).

Where VPFP is described in Section 3.3, for example:

VPFP is:

(PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0)

List of results after the specified cycle reservation method is successful:

{

(VPFP): (intf0,Cycle0,1),

(VPFP): (intf0,Cycle1,1),

(VPFP): (intf0,Cycle2,1),

(VPFP): (intf0,Cycle3,1),

(VPFP): (intf0,Cycle4,1),

(VPFP): (intf0,Cycle5,1),

(VPFP): (intf0,Cycle6,1),

(VPFP): (intf0,Cycle7,1),

}

After the reservation of non-specified cycle resources is successful, the resulting list contains the actually reserved cycles and their corresponding resources. Suppose the requirements of Flow1 are as follows:

{(VPFP): (intf0, InvalidCycle, 10, 2)}

After the allocation is successful, the result list may be:

{(VPFP): (intf0,Cycle0,10)}

It may also be as follows, when one cycle cannot meet the transmission demand, it is allocated from multiple cycles:

{

(VPFP): (intf0, Cycle0, 5),

(VPFP): (intf0, Cycle1, 5)

}

Note: When resources requirements are allocated for multiple scheduling cycles, ensure that the resources allocated for each cycle can transmit the full packet in service.

After the resources is successfully reserved, the MCPE needs to record the VPFC planned for the DetNet flow, which will be used for the reasons of VPFC recycling, modifying, etc. Since the topology may change, resources reclamation cannot rely on topology information. Therefore, it is necessary to save the VPFC allocated on the PE on the controller plane.

5. Examples of Resources-Related Processing

This section gives a detailed resources-related processing flow. At the same time, the complex issue of resources recycling will be briefly covered, and there will be discussions of unresolved issues related to resources reservation. These discussions do not give any direction to the solution developer for how they should do with the forwarding resources recycling, but to point out that these issues should not be ignored in the implementation.

5.1. Collection Process of Cycle Resources

For simplicity, this solution is based on the existing best-effort forwarding mechanism and do some extensions. In terms of network topology and path planning, it directly inherits current implementation. For example, collecting topology through IGP and BGP-LS, measuring inter-node delay through NQA or TWAMP, collecting link delay through NETCONF, planning paths that meet application delay requirements based on CSPF, are all existing technologies. The specific implementation is beyond the scope of this document. In order to implement this solution, it is necessary to add some new functions on the basis of the existing implementation, including establishing a resources database in the periodic forwarding domain. The database needs to include the association relationship of interface, cycle, and forwarding resources, and also needs to include the mapping relationship between the outgoing interfaces of the upstream node and the downstream node. For reference, a new added process can be:

1. Plan the scheduling cycle of nodes in the deterministic domain, including the cycle length and the number.

2. Install the planned path, and configure the scheduling cycle to the interfaces of the nodes along the path;

3. The MCPE collects the number of resources of the scheduling cycle on the outgoing interface of each node along the path through YANG/NETCONF.

4. The MCPE measures the mapping relationship between the outgoing interface cycles of the upstream node and the downstream node. For example, if a packet is sent from cycle 0 in the upstream node's outgoing interface, and when it reaches the downstream node, cycle 5 of the downstream node's outgoing interface is being scheduled, then the mapping relationship between the two interfaces is 0 to 5. To achieve this, a new measurement method is needed, which will be described in a separate document (TBD);

5. The MCPE collects the cycle mapping information between nodes and the processing delay inside each node through NETCONF/YANG. Based on this information, the injective function described in Section 4.2.4 is established;

6. The MCPE uses the collected path information and the cycle mapping information to establish the path information described in Section 3.1, and saves them in the control plane.

5.2. Process Flow of Reserving Cycle Resources

When a deterministic service flow session needs to be established, it sends a request to the MCPE. The MCPE selects a path and allocate resources according to the flow characteristics. The overall process is as follows:

1. When the USER has a deterministic service flow session to establish, it sends a request and the flow characteristics to the MCPE, following the format described in [RFC9016];

2. The MCPE translates the flow characteristics provided by the USER into resources demands. If there are any interfaces in the planned path whose periodic resources information has not been collected by the MCPE, the MCPE will install the interfaces and collect their resources information as described in Section 5.1;

3. The MCPE translates the resources demand into the resources units demand as described in Section 4.2.1, and decides whether the demand type is specified cycle (type 1, see Section 5.2.1) or un-specified cycle (type 2, Section 5.2.2). The form of the translated resources demand is described in Section 4.2.5;

4. The MCPE reserves the resources according to the demand type. See Section 5.2.1 and Section 5.2.2 for detail;

5. If the resources reservation succeeds, the session will be established. Otherwise, the MCPE will select a new path and repeat from step 2, until the timeout is hit;

6. When successfully planned VPFC, the allocated resources need to be delivered to the head node of the VPFP. For the configuration data model, see Section 3.3. The head node of the VPFP schedules according to the allocated resources. The implementation of the head node of the VPFP is beyond the scope of this document and will be described in detail in other documents (TBD).

After a VPFC is established, the VPFC is scheduled based on the its buffer resources in the scheduling cycle of the VPFP head node, so that conflict of periodic data forwarding will not occur in the CSQF domain.

5.2.1. Reservation Calculation for Resources with Specified Cycle

Assuming the format of the path information calculated by MCPE is as described in Section 3.1, then VPFP1 is:

(PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0)

The resources demand list for specified cycle is:

{

(VPFP1): (intf0,Cycle0,1,1),

(VPFP1): (intf0,Cycle1,1,1),

(VPFP1): (intf0,Cycle2,1,1),

(VPFP1): (intf0,Cycle3,1,1),

(VPFP1): (intf0,Cycle4,1,1),

(VPFP1): (intf0,Cycle5,1,1),

(VPFP1): (intf0,Cycle6,1,1),

(VPFP1): (intf0,Cycle7,1,1)

}

For the convenience, the node, outgoing interface, and available resources are respectively abbreviated as cn (Current Node), ci, and ar. So cn.ci.ar[c] denotes the available resources in cycle c of the outgoing interface of current node. For the above resources demands for specified scheduling cycle, perform the following resources reservation calculation and reservation processing:

1. Let SubDemand be the first entry(sub-demand) of DemandList. The format of SubDemand is: (VPFP):(oif,cycle,res,min); Go to 2);

2. Get VPFP from SubDemand. Let IngressPE be the first node in VPFP, and EgressPE be the last node in VPFP. Let cn be IngressPE. Variant j keeps the value of the current node's scheduling cycle, let j=cycle. Go to 3);

3. If cn.ci.ar[j] >= res, it indicates that the available resources in the current scheduling cycle meet the resources demand, go to 4); otherwise, the reservation fails, go to 8);

4. If cn is the EgressPE of VPFP, then the resources reservation of the entire VPFP for SubDemand completed. Save the result to ResultList, then go to 5); otherwise, go to 6);

5. If SubDemand is the last entry in DemandList, it indicates that all sub-demands have been satisfied, then go to 7); Otherwise, let SubDemand be the next entry in DemandList, then go to 2);

6. From VPFP, get the mapping function fx between cn.ci and the outgoing interface of the next node. Then update cn, let it be the next node in VPFP, and cn.ci is the corresponding outgoing interface. Calculate the scheduling cycle for the outgoing interface of the next node, and update the value of j. That is, let j=fx(j). Go to 3);

7. The resources reservation calculation succeed, and the resources reservation will be performed. For details, see Section 5.2.3.

8. The resources reservation calculation fails, the ResultList should be recycled. For details, see the overall process at the beginning of Section 5.2.

5.2.2. Reservation Calculation for Resources with Non-Specified Cycle

Assuming the format of the path calculated by MCPE is as described in Section 3.1, and VPFP1 is:

(PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0),

and VPFP2 is:

(PE2,Intf0) g1 (P1,intf3) f2 (P3,intf3) f3 (P4,intf2) f5 (PE5,Intf1)

The resources demand list for unspecified cycle is:

{

(VPFP1): (intf0, InvalidCycle, 10,2);

(VPFP2): (intf0, InvalidCycle, 8,2)

},

Where InvalidCycle means invalid cycle, whose value is defined by the implementation.

For the above resources demands for non-specified scheduling cycle, perform the following resources reservation calculation and reservation processing:

1. Let SubDemand be the first sub-demand entry of DemandList. The format of SubDemand is: (VPFP): (oif,cycle,res,min). Go to 2);

2. Get VPFP from SubDemand. Let IngressPE be the first node in VPFP, and EgressPE be the last node in VPFP. Let cn be IngressPE. Some variants are defined: j keeps the value of the current node's scheduling cycle; c keeps the value of the scheduling cycle in IngressPE; Demand keeps the number of resources to be reserved; CandDemandRes keeps the number of candidate demand resources corresponding to cycle j. The search starts from the cycle 0 in IngressPE, so the initial values are

: cn=IngressPE; c=j=0; SubDemandRes=res;

CandDemandRes=SubDemandRes; Go to 3);

3. If cn.ci.ar[j] >= CandDemandRes, it indicates that the available resources of the current scheduling cycle meet the resources demand SubDemandRes. Let j=c, go to 4); otherwise, go to 8);

4. If cn is the EgressPE of VPFP, it indicates that the available resources of all scheduling cycles along the VPFP corresponding to the scheduling cycle c of the head node meet the resources demand CandDemandRes, then go to 5); otherwise, go to 6);

5. Record the SubResult currently calculated, where SubResult is (VPFP): (oif, c, CandDemandRes), add SubResult to the result-list ResultList, go to 7);

6. From VPFP, get the mapping function fx between cn.ci and the outgoing interface of the next node. Then update cn, let it be the next node in VPFP, and cn.ci is the corresponding outgoing interface. Calculate the scheduling cycle for the outgoing interface of the next node, and update the value of j. That is, let j=fx(j). Go to 3);

7. If CandDemandRes >= SubDemandRes, it indicates that the remaining resources demand of SubDemand is met, go to 8); otherwise, go to 10);

8. If SubDemand is the last entry in DemandList, it indicates thatall sub-demands have been satisfied, then go to step 19); otherwise, go to 9);

9. Let SubDemand be the next entry in DemandList, then go to 2);

10. If c<cycles-1, which means c is not the last scheduling cycle, go to 11); otherwise, it means all cycles of IngressPE have been traversed, but the resources demands cannot be met, go to step 20);

11. Let SubDemandRes = SubDemandRes - CandDemandRes. Go to 12);

12. If SubDemandRes<min, it indicates that the remaining resources demand is less than the minimum allocation granularity, then go to 13); otherwise, go to 14);

13. Let SubDemandRes=min, which updates the remaining resources demand as the minimum allocation granularity; go to 14);

14. Prepare for the next scheduling cycle of IngressPE to be processed. Let CandDemandRes=SubDemandRes, c=c+1, j=c, cn=IngressPE. Go to step 3);

15. If c<cycles-1, which means c is not the last scheduling cycle, then go to 16); otherwise, it means all cycles of IngressPE have been traversed, but the resources demands cannot be met, go to step 20);

16. Let k=floor(cn.ci.ar[j]/min), where min is the minimum allocation granularity. Go to 17);

17. If k>0, it indicates that the available resources of the current scheduling cycle of the current node meet part of the resources demand, then go to 18); otherwise go to 14);

18. Let CandDemandRes=k*min, go to 4);

19. The resources reservation calculation succeeds, and the resources reservation will be performed. For details, see Section 5.2.3.

20. The resources reservation calculation fails, release the resources in the ResultList. For failure handling, see the overall process at the beginning of Section 5.2.

5.2.3. Execution of Cycle Resources Reservation

After the resources reservation calculation, the resources reservation is executed. The MCPE traverse the ResultList and perform the following resources reservation operations:

1. Let SubResult be the first entry in ResultList. The format of SubResult is: (VPFP): (oif, cycle, res). Go to 2);

2. Get VPFP from SubResult. Let IngressPE be the first node in VPFP, and EgressPE be the last node in VPFP. Let cn be IngressPE. Let j be the current cycle, so j=cycle. Go to 3);

3. Update the resources of the cycle j, that is, the number of resources of cn.ci.ar[j] is reduced by res. See Section 4.2.3 for the description of the resources of the cycle. Go to 4);

4. If cn is the EgressPE of VPFP, then the resources reservation of the entire VPFP completed, go to 5); otherwise, go to 6);

5. If SubResult is the last entry in ResultList, then the reservation is completed for all sub-results, go to 7); otherwise, let SubResult be the next entry in ResultList, go to 2);

6. From VPFP, get the mapping function fx between cn.ci and the outgoing interface of the next node. Then update cn, let it be the next node in VPFP, and cn.ci is the corresponding outgoing interface. Calculate the cycle for the outgoing interface of the next node, and update the value of j. That is, let j=fx(j). Go to 4);

7. Save ResultList to the database, then return success.

5.2.4. Resources Reservation for PREOF

For a PREOF implementation, each resources reservation demand on a VPFP forms a sub-demand (see Section 4.2.5). Multiple sub-demands form a demand list for resources reservation calculation and reservation (see Section 4.2.1, Section 4.2.2 and Section 4.2.3). For example, suppose there is a deterministic service flow that requires two member paths to form a compound path to increase reliability. Where one of the member paths is VPFP1:

(PE1,Intf0) f1 (P1,Intf3) f2 (P3,Intf3) f3 (P4,Intf2) f4 (PE5,Intf0)

Another Member Path is VPFP2:

(PE1,Intf1) g1 (P2,Intf3) f2 (P5,intf3) f3 (P6,intf2) f5 (PE5,Intf0)

In this example, the DetNet flow is injected from PE1 and copied on PE1. The original flow and the copy are sent from Intf0 and Intf1 respectively. The original flow and the copy are finally aggregated on PE5, and the aggregated data flows out from Intf0 of PE5 after processing. The bandwidth requirement of this service flow is 10 resources units. Due to the multi-path, the jitter caused by unequal path lengths is greater than the jitter caused by the access PE scheduling cycle. Therefore, for the PREOF deployment method, the resources reservation method with a non-specified cycle is more practical. Assuming that the resources demand of the service flow is 10 resources units, and the minimum granularity of resources allocation in each cycle is 2 resources units, the following non-specified cycle resources demand list is formed:

{

(VPFP1): (intf0, InvalidCycle, 10,2);

(VPFP2): (intf0, InvalidCycle, 10,2);

},

Where InvalidCycle is the invalid cycle, whose value is defined by the implementation.

5.2.5. Bandwidth Increase Procedure

When the bandwidth demand of a service flow increases, convert the newly added bandwidth demand into resources demand to form the demand list described in Section 4.2.5, and execute the combined processing flow of Section 4.2.1 and Section 4.2.4 or Section 4.2.2 and Section 4.2.4.

5.2.6. Reroute

For a single path change, the MCPE recycles the old path resources and reserves demanded resources along the new path.

For the PREOF implementation, the process for one VPFP change is same as the process for a single path change.

5.2.7. Reclaiming Reserved Resources

Resources recycling is a key issue. The resources recycling process is relatively complex. In an LDN, resources that have been allocated will not be used for various reasons. If they are not recycled, resources "leakage" will occur, reducing the effective utilization of the network.

The reasons that may trigger the resources recovery include:

1. DetNet flow deletion;

2. Changes in service flow demands. One scenario is the flow's resources demand changes. In this case, the original VPFC may no longer meet the demand, and needs to be re-planned, so the allocated resources should be recycled and the new ones should be reserved. Another scenario is the resources demand of a flow is reduced. In this case, some resources that have been reserved for the flow need to be recycled, but no new resources needs to be reserved.

3. One or more nodes along the VPFP fail. In this case, the resources reserved by all service flows in the failure nodes need to be recycled. For PREOF, some resources may serve one than one VPFPs, in which case the resources can be recycle only when all the VPFPs fail. The detailed process for node failing is out of scope of this document and left for further study.

4. The controller detects a flow failure through monitoring methods like periodical handshaking. [I-D.ietf-detnet-controller-plane-framework] mentions the convergent management plane method. Resources recovery is a comprehensive and complex problem, and the convergent management plane method is also suitable.

5. In PREOF mode, if resources reservation for some member VPFPs fails, all the resources reserved for all member VPFPs should be recycled.

As a common resource, the scheduling cycle resources should be correlated with the OAM module. When OAM detects some failure or abnormality, recycling of the scheduling cycle resources should be triggered. Therefore, the scheduling cycle resources recovery is also a part of the OAM that needs to be enhanced.

6. Security Considerations

The security considerations related to resources reservation are the same as those described in [I-D.ietf-detnet-controller-plane-framework]. In addition, it is necessary to deal with the errors

mentioned in [IEEE802.1Qci], such as exceeding SDU, etc. This kind of process includes discarding and counting the packets, and is usually implemented on the forwarding plane.

7. IANA Considerations

This document makes no IANA requests.

8. Acknowledgements

The authors express their appreciation and gratitude to Min Liu for the review and helpful comments.

9. Contributors

The editor wishes to thank and acknowledge the following author for contributing text to this document.

Lei Zhou
New H3C Technologies
100094
Email: zhou.leiH@h3c.com

Zuopin Cheng
New H3C Technologies
100094
Email: czp@h3c.com

Ning Pan
New H3C Technologies
100094
Email: panning@h3c.com

Shenchao Xu
New H3C Technologies
100094
Email: xushenchao@h3c.com

Xusheng Chen
New H3C Technologies
100094
Email: cxs@h3c.com

Pin Wu
New H3C Technologies
100094
Email: wupin@h3c.com

Jun Chu
New H3C Technologies
100094
Email: chu.jun@h3c.com

Wei Wang
New H3C Technologies
100094
Email: david_wang@h3c.com

Xinmin Liu
New H3C Technologies
100094
Email: liuxinmin@h3c.com

10. References

10.1. 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>.

[RFC8938] Varga, B., Ed., Farkas, J., Berger, L., Malis, A., and S. Bryant, "Deterministic Networking (DetNet) Data Plane Framework", RFC 8938, DOI 10.17487/RFC8938, November 2020, <https://www.rfc-editor.org/info/rfc8938>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <https://www.rfc-editor.org/info/rfc3473>.

10.2. Informative References

[I-D.chen-detnet-sr-based-bounded-latency] Chen, M., Geng, X., Li, Z., Joung, J., and J. Ryoo, "Segment Routing (SR) Based Bounded Latency", Work inProgress, Internet-Draft, draft-chen-detnet-sr-based-bounded-latency-03, 7 July 2023, <https://datatracker.ietf.org/doc/html/draft-chen-detnet-sr-based-bounded-latency-03>.

[I-D.eckert-detnet-tcqf] Eckert, T. T., Li, Y., Bryant, S., Malis, A. G., Ryoo, J., Liu, P., Li, G., Ren, S., and F. Yang, "Deterministic Networking (DetNet) Data Plane - Tagged Cyclic Queuing and Forwarding (TCQF) for bounded latency with low jitter in large scale DetNets", Work in Progress, Internet-Draft, draft-eckert-detnet-tcqf-04, 7 July 2023, <https://datatracker.ietf.org/doc/html/draft-eckert-detnet-tcqf-04>.

[I-D.ietf-detnet-controller-plane-framework] Malis, A. G., Geng, X., Chen, M., Qin, F., Varga, B., and C. J. Bernardos, "Deterministic Networking (DetNet) Controller Plane Framework", Work in Progress, Internet-Draft, draft-ietf-detnet-controller-plane-framework-04, 13 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-detnet-controller-plane-framework-04>.

[I-D.ietf-detnet-scaling-requirements] Liu, P., Li, Y., Eckert, T. T., Xiong, Q., Ryoo, J., zhushiyin, and X. Geng, "Requirements for Scaling Deterministic Networks", Work in Progress, Internet-Draft, draft-ietf-detnet-scaling-requirements-03, 7 July 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-detnet-scaling-requirements-03>.

[I-D.peng-detnet-packet-timeslot-mechanism] Peng, S., Liu, P., Basu, K., Liu, A., Yang, D., and G. Peng, "Timeslot Queueing and Forwarding Mechanism", Work in Progress, Internet-Draft, draft-peng-detnet-packet-timeslot-mechanism-03, 5 July 2023, <https://datatracker.ietf.org/doc/html/draft-peng-detnet-packet-timeslot-mechanism-03>.

[IEEE802.1Qch] IEEE, "IEEE Standard for Local and metropolitan areanetworks -- Bridges and Bridged Networks - Amendment 29: Cyclic Queuing and Forwarding", IEEE 802.1Qch-2017, DOI 10.1109/IEEESTD.2017.7961303, 28 June 2017, <https://doi.org/10.1109/IEEESTD.2017.7961303>.

[IEEE802.1Qci] IEEE, "IEEE Standard for Local and metropolitan areanetworks -- Bridges and Bridged Networks - Amendment 28: Per-Stream Filtering and Policing", IEEE 802.1Qci-2017, DOI 10.1109/IEEESTD.2017.8064221, 28 September 2017, <https://doi.org/10.1109/IEEESTD.2017.8064221>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[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>.

[RFC9016] Varga, B., Farkas, J., Cummings, R., Jiang, Y., and D. Fedyk, "Flow and Service Information Model for Deterministic Networking (DetNet)", RFC 9016, DOI 10.17487/RFC9016, March 2021, <https://www.rfc-editor.org/info/rfc9016>.

[RFC9320] Finn, N., Le Boudec, J.-Y., Mohammadpour, E., Zhang, J., and B. Varga, "Deterministic Networking (DetNet) Bounded Latency", RFC 9320, DOI 10.17487/RFC9320, November 2022, <https://www.rfc-editor.org/info/rfc9320>.

Authors' Addresses

Daorong Guo
New H3C Technologies Co., Ltd
Beijing
100094
China
Email: guodaorong@h3c.com

Guangliang Wen
New H3C Technologies Co., Ltd
Beijing
100094
China
Email: wenguangliang@h3c.com

Kehan Yao
China Mobile
Beijing
100053
China
Email: yaokehan@chinamobile.com

Quan Xiong
ZTE Corporation
Wuhan
430223
China
Email: xiong.quan@zte.com.cn

Guoyu Peng
Beijing University of Posts and Telecommunications
Beijing
100876
China
Email: guoyupeng@bupt.edu.cn

XUEJUN YOU
New H3C Technologies Co., Ltd
Beijing
100094
China
Email: yoe@h3c.com

Shiyin Zhu
New H3C Technologies Co., Ltd
Beijing
100094
China
Email: zhushiyin@h3c.com