DetNet
Internet-Draft
Intended status: Standards Track
Expires: 5 September 2024

Shaofu. Peng
ZTE
Peng. Liu
China Mobile
Kashinath. Basu
Oxford Brookes University
Aihua. Liu
ZTE
Dong. Yang
Beijing Jiaotong University
Guoyu. Peng
Beijing University of Posts and Telecommunications

4 March 2024

Timeslot Queueing and Forwarding Mechanism
draft-peng-detnet-packet-timeslot-mechanism-06

Abstract

IP/MPLS networks use packet switching (with the feature store-and-forward) and are based on statistical multiplexing. Statistical multiplexing is essentially a variant of time division multiplexing, which refers to the asynchronous and dynamic allocation of link timeslot resources. In this case, the service flow does not occupy a fixed timeslot, and the length of the timeslot is not fixed, but depends on the size of the packet. Statistical multiplexing has certain challenges and complexity in meeting deterministic QoS, and its delay performance is dependent on the used queueing mechanism. This document further describes a generic time division multiplexing scheme in IP/MPLS networks, which we call timeslot queueing and forwarding (TQF) mechanism. It aims to bring timeslot resources to layer-3, to make it easier for the control plane to calculate the delay performance based on the timeslot resources, and also make it easier for the data plane to create more flexible timeslot mapping. The functions of TQF can better meet large scaling requirements.

Copyright Notice

Table of Contents

1. Introduction

IP/MPLS networks use packet switching (with the feature store-and-forward) and are based on statistical multiplexing. The discussion of supporting multiplexing in the network was first seen in the time division multiplexing (TDM), frequency division multiplexing (FDM) and other technologies of telephone communication network (using circuit switching). Statistical multiplexing is essentially a variant of time division multiplexing, which refers to the asynchronous and dynamic allocation of link resources. In this case, the service flow does not occupy a fixed timeslot, and the length of the timeslot is not fixed, but depends on the size of the packet. In contrast, synchronous time division multiplexing means that a sampling frame (or termed as time frame) includes a fixed number of fixed length timeslots, and the timeslot at a specific position is allocated to a specific service. The utilization rate of link resources in statistical multiplexing is higher than that in synchronous time division multiplexing. However, if we want to provide deterministic end-to-end delay in packet switched networks based on statistical multiplexing, the difficulty is greater than that in synchronous time division multiplexing. The main challenge is to obtain a deterministic upper bound on the queueing delay, which is closely related to the queueing mechanism used in the network.

In addition to IP/MPLS network, other packet switched network technologies, such as ATM, also discusses how to provide corresponding transmission quality guarantee for different service types. Before service communication, ATM needs to establish a connection to reserve virtual path/channel resources, and use fixed-length short cells and timeslots. The advantage of short cell is small interference delay, but the disadvantage is low encoding efficiency. The mapping relationship between ATM cells and timeslots is not fixed, so it still depends on a specific cells scheduling

mechanism (such as [ATM-LATENCY]) to ensure delay performance. Although the calculation of delay performance based on short and fixed-length cells is more concise than that of IP/MPLS networks based on variable length packets, they all essentially depend on the queueing mechanism.

[TAS] introduces a synchronous time-division multiplexing method based on gate control list (GCL) rotation in Ethernet LAN. Its basic idea is to calculate when the packets of the service flow arrive at a certain node, then the node will turn on the green light (i.e., the transmission state is set to OPEN) for the corresponding queue inserted by the service flow at that time duration, which is defined as TimeInterval between two adjacent items in gating cycle. The TimeInterval is exactly the timeslot resource that can be reserved for service flow. A set of queues is controlled by the GCL, with round robin per gating cycle. The gating cycle (e.g, 250 us) contains a lot of items, and each item is used to set the OPEN/CLOSED states of all traffic class queues. By strictly controlling the release time of service flow at the network entry node, multiple flows always arrive sequentially during each gating cycle at the intermediate node and are sent during their respective fixed timeslot to avoid conflicts, with extremely low queueing delay. However, the GCL state (i.e., items set, and different TimeInterval value between any two adjacent items) is related with all ordered flows that passes through the node. Calculating and installing GCL states separately on each node has scalability issues.

[CQF] introduces a synchronous time-division multiplexing method based on fixed-length cycle in Ethernet LAN. [ECQF] is a further enhancement of the classic CQF and may be applicable to large scaling networks. CQF with 2-bin mode or ECQF with 3-bin mode only uses a small number of cycles to establish the cycle mapping between a port-pair of two adjacent nodes, which is independent of the individual service flow. The cycle mapping may be maintained on each node and swaped based on a single cycle id carried in the packet during forwarding ([I-D.eckert-detnet-tcqf]), or all cycle mappings are carried in the packet as a cycle stack and read per hop during forwarding ([I-D.chen-detnet-sr-based-bounded-latency]). According to [ECQF], how many cycles (i.e., x-bin mode) are required depends on the proportion of the variation in intra-node forwarding delay relative to the cycle size. If the proportion is small, 3-bin is enough, otherwise, more than 3 bins needed. Compared to TAS, CQF/ECQF no longer maintains GCL on each node, but instead replaces the large number of variable length of timeslots related to service flows in GCL with a small number of fixed length cycles unrelated to service flows. Thus, CQF/ECQF simplifies the data plane, but leaves the complexity to the control plane, by calculating and controlling the release time of service flow at the network entry, to guarantee no conflicts between flows in any cycle on any intermediate nodes.

In order to meet the large scaling requirements, this document describes a scheduling mechanism for enhancing TAS. Firstly, it brings timeslot type of resources to layer-3 and construct timeslot resources on each link within gating cycle, which are advertised in the network, and opened and reserved for DetNet flows to implement timeslot orchestration (i.e., flow interleaving). Secondly, it defines timeslot based queueing mechanism on the data plane with on-time or in-time behavior. We call this mechanism as Timeslot Queueing and Forwarding (TQF), as a supplement to IEEE 802.1 TSN TAS. In TQF, The selected length of gating cycle depends on the length of the supported service burst interval.

Similar to TAS and CQF/ECQF, TQF is also TDM based scheduling mechanisms.

* Compared to classic TAS, TQF may use round robin queues corresponding to the count of timeslots during gating cycle, while TAS only maintains queues corresponding to the number of traffic classes and one of them is used for the Scheduled Traffic (i.e., DetNet flows). That means TQF need more queues than TAS (i.e., multiple timeslot queues vs single traffic class queue). However, TAS needs to use other complex methods to control the arrival order of all flows sharing the same traffic class queue to isolate them (so that each flow faces almost zero queuing delay), while TQF's timeslot queue naturally isolates flows by timeslot id of

gating cycle. And, TQF with in-time scheduling mode may use a single PIFO (put in first out) queue to approximate the ultra-low delay of TAS.

* Compared to CQF/ECQF, TQF on-time scheduling maintains round robin queues corresponding to the count of timeslots during gating cycle, while CQF/ECQF maintains extra tolerating queues depending on the proportion of the variation in intra-node forwarding delay relative to the cycle size. There is no gating cycle with its timeslot resources designed by CQF/ECQF, it needs to use additional flow interleaving method to control the arrival order of flows sharing the same cycle queue to isolate flows (or alternatively tolerate overprovision), while TQF's timeslot queue naturally isolates flows by timeslot id of gating cycle. This is also the semantic difference between cycle id and timeslot id, where the former is used to indicate the NO. of the aggregated queues such as sending, receiving, or tolerating queue, rather than indicating the individual timeslot resource within the gating cycle like the later. That is, after defining timeslot resources in IP/MPLS, TQF does not limit the implementations of the data structure type corresponding to timeslot resources on the data plane, which may be round robin queues, or a single PIFO queue.

1.1. 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.

2. Terminology

The following terminology is introduced in this document:

Timeslot: The unit of TQF scheduling. It needs to design a reasonable value, such as 10us, to send at least one complete packet. Different nodes can be configured with different length of timeslot.

Timeslot Scheduling: The packet is stored in the buffer zone corresponding to a specific timeslot id, and may be sent before (in-time mode) or within (on-time mode) that timeslot. The timeslot id is always a NO. from the orchestration period.

Service Burst Interval: The traffic specification of DetNet flow

generally follows the principle of generating a specific burst amounts within a specific length of periodic burst interval. For example, a service generates 1000 bits of burst per 1 ms, where 1 ms is the service burs interval.

Orchestration Period: The orchestration period is used to orchestrate the DetNet flow and depends on the service burst interval of DetNet flows. It is actually the gating cycle in TAS, and its length depends on the length of the service burst interval of all deterministic flows. It contains a fixed count (termed as N and numbered from 0 to N-1) of timeslots. For example, the orchestration period include 1000 timeslots and each timeslot length is 10 us. The timeslot resources within the orchestration period can be allocated for DetNet flows, i.e., which timeslots are occupied by flows and how many bits are occupied in a timeslot. The orchestration period is the Least Common Multiple of all service burst intervals. It is also a multiple of the scheduling period. It is recommended that all nodes of the network be configured with the same length of orchestration period (note that timeslot length may still be different), because it is service-related and also crucial for establishing a stable timeslot mapping relationship. It is possible to configure multiple instances of orchestration period with different lengths, however, nodes communicate with each other based on the same length of orchestration period.

Ongoing Sending Period: The orchestration period which the ongoing sending timeslot belongs to.

Scheduling Period: The scheduling period depends on the hardware's buffer resources that is supported by the device. Its length reflects the count of the timeslot resources (termed as M and numbered from 0 to M-1) that is actually instantiated on the data plane, which is limited by hardware capabilities. Scheduling period length may be less than or equal to orchestration period length in the case of on-time mode, or larger than or equal to orchestration period length in the case of in-time mode. Different nodes can be configured with different scheduling period length. When the orchestration period is larger than the scheduling period, different parts of the orchestration period can be mapped to a single scheduling period using appropriate mapping methods.

Incoming Timeslot: For the headend of the path, when the application flow received from the client side reaches the UNI port, the corresponding timeslot of the UNI port after traffic policing is the incoming timeslot of the packet. For an intermediate node in a specific path, the timeslot contained in the packet received

from the upstream node (i.e., the outgoing timeslot of the upstream node) is its incoming timeslot. An incoming timeslot is the timeslot id in the orchestration period.

Outgoing Timeslot: When sending a packet to the outgoing port, according to resource reservation or certain rules, it chooses to send packet in the specified timeslot of that port, which is the outgoing timeslot. An outgoing timeslot is the timeslot id in the orchestration period.

Ongoing Sending Timeslot: When the end of the incoming timeslot to which the packet belongs reaches a specific port, the timeslot currently in the sending state is the ongoing sending timeslot of that port. Note that the ongoing sending timeslot is different with the outgoing timeslot. An ongoing sending timeslot is the timeslot id in the orchestration period.

3. Overview

This scheme introduces the time-division multiplexing scheduling mechanism based on the fixed length timeslot in the IP/MPLS network. Note that the time-division multiplexing here is a L3 packet-level scheduling mechanism, rather than the TDM port (such as SONET/SDH) implemented in L1. The latter generally involves the time frame and the corresponding framing specification, which is not necessary in this document. The data structure associated with timeslot resources may be implemented using round robin queues, or a single PIFO queue, etc.

Figure 1 shows the TQF scheduling behavior implemented by the intermediate node P through which a deterministic path passes.

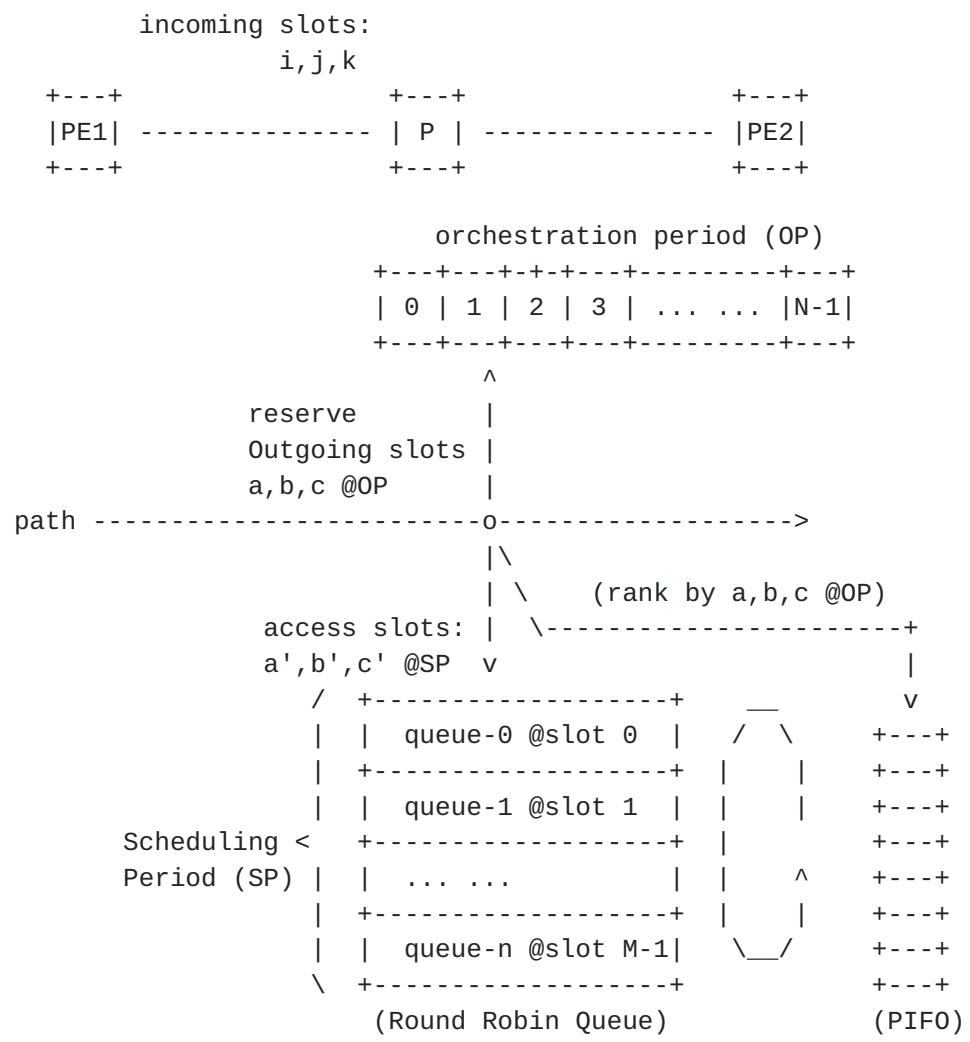

Figure 1: Timeslot Based Scheduling Mechanism

Where, both the orchestration period and the scheduling period consist of multiple timeslots, the number of timeslots supported by orchestration period is related to the length of the service burst interval, while the number of timeslots supported by scheduling period is limited by hardware capabilities, and it may be instantiated by a Round Robin queue or PIFO.

The total amount of bits that can be reserved or sent in each timeslot can be preset, generally not exceeding the result of the service rate multiplied by the timeslot length. The TQF scheduler may configure a specific service rate, which must not exceed the port bandwidth.

The orchestration period of all nodes in the network does not require phase alignment. However, within each node, the phase of timeslot of orchestration period and the scheduling period are strictly aligned. This is indeed natural because multiple scheduling periods forms an orchestration period. In other words, different parts of the orchestration period share and reuse the same scheduling period. That is, within a node, no synchronization mechanism is required between orchestration period and scheduling period.

In the figure, the path allocates timeslots a, b, c from the orchestration period of the outgoing port (link P-PE2) for incoming timeslots i, j, k respectively. It finally accesses the mapped timeslot from scheduling period. There is a mapping relationship between the timeslot z of orchestration period and the timeslot z' of scheduling period, i.e., z' = f(z). There are many mapping options, such as z'=z, z'=z+offset, z'=z%M, and z'=random(z), etc. Which option to use depends on the data structure instantiated for timeslot resources and the specific resource reservation method. In this document, we mainly discuss two mapping option, one is z'=z%M (in the case of round robin queue), the other is z'=z (in the case of PIFO). Section 3.2.1 provides more information on option z'=z%M. Note that option z'=z does not mean that the scheduling period physically instantiates every timeslot resource of the orchestration period.

In general, TQF mechanism implemented on all nodes in the network may use the same length of timeslot and scheduling period. However, considering the capability differences of each node in the network (for example, the capabilities of the edge nodes are weaker than the core nodes), it is feasible for different nodes/links to use different length of timeslot and scheduling period.

A TQF enabled link may configure multiple instances of orchestration period with different lengths. Nodes communicate (i.e., control plane messages, or data plane packets) with each other based on the same length of orchestration period. A specific orchestration periold instance constrains the scale of timeslot resources, and is used for both control plane and data plane. That is, orchestration periold instance should not be considered as just a management object.

The scheme involves two aspects: the path calculation and timeslot resource reservation in the control plane, and timeslot resource access in the data plane.

EDITOR'S NOTE: according to WG's discussion this document may be separated to two documents in the future, one focuses on timeslot resource reservation and the other on timeslot resource based scheduling.

3.1. Timeslot Resource Reservation in Control-plane

The control plane (centralized controller or distributed protocol) can reserve corresponding timeslot resources along the deterministic path. Note that if a path carries multiple DetNet flows, then the path may reserve timeslot resources for the aggregated DetNet flow, and may reserve the burst resources in multiple timeslots in the orchestration period at the same time. However, it would still be beneficial to distinguish between reservation sub-tasks corresponding to different DetNet flows in the combined reservation task. In this document, we refer to a reservation sub-task as an individual timeslot resource reservation action related to a DetNet flow. Note that one or more reservation sub-tasks for a specific DetNet flow may be derived based on its TSpec, and each reservation sub-task will allocate corresponding timeslot. The intermediate nodes do not maintain the state of DetNet flow and only reserve timeslot resources based on the reservation sub-tasks.

During resource reservation, it is necessary to distinguish the requirements between low latency service and non-low latency service. For low latency service requirements, the physical offset between the reserved outgoing timeslot and the incoming timeslot is small; while for loose latency service requirements, this physical offset can be large. It is necessary to maintain the end-to-end total residence delay budget for each reservation sub-task. This is used to select outgoing timeslot at each node. The sum of residence delays caused by all nodes should not exceed the total residence delay budget.

Multiple reservation sub-tasks may generate different incoming/ outgoing timeslot mapping relationships on node P. For example:

* The timeslot mapping relationship created by the sub-task-1:

  <(incoming port a, incoming slot id 3), (outgoing port b, outgoing slot id 60)>

* The timeslot mapping relationship created by the sub-task-2:

  <(incoming port a, incoming slot id 3), (outgoing port b, outgoing slot id 61)>

Special care should be taken not to confuse the use of different mapping relationships. For specific DetNet flows, P need to explicitly use specific timeslot mapping relationships.

It is recommended, but not mandatory, to reserve timeslot resources on the outgoing port of each hop from the headend of the path to the endpoint, that is, first determine the timeslot reserved on the

headend, then determine the timeslot reserved on the next hop , and so on. We assume that the DetNet flow has a periodic arrival time (i.e., the time when the regulated packet reaches the scheduler), and there is an ideal position relationship between the arrival time and the orchestration period of the headend, so selecting the outgoing timeslot closed to the arrival time or within the expected offset range in the orchestration period can minimize the residency delay of the packet on the headend. However, sometimes it is necessary to get a larger residence delay on the headend and a smaller residence delay on other nodes to ensure successful path calculation.

3.1.1. Timeslot Mapping Relationship

In order to reserve outgoing timeslot resources for the DetNet flow , it is necessary to first determine the ongoing sending timeslot that the incoming timeslot falls into, i.e., the mapping relationship between the incoming timeslot and the ongoing sending timeslot.

Two methods are provided in the following sub-sections to determine the mapping relationship between the incoming timeslot and the ongoing sending timeslot.

3.1.1.1. Deduced by BTM

Figure 2 shows that there are three nodes U, V, and W in turn along the path. All nodes are configured with orchestration period of the same length (termed as OPL), which is crucial for establishing a fixed timeslot mapping relationship.

* Port_u2 has timeslot length L_u2, and an orchestration period contains N_u2 timeslots.

* Port_v1 has timeslot length L_v1, and an orchestration period contains N_v1 timeslots.

* Port_v2 has timeslot length L_v2, and an orchestration period contains N_v2 timeslots.

Hence, L_u2*N_u2 = L_v1*N_v1 = L_v2*N_v2. In general, the link bandwidth of edge nodes is small, and they will be configured with a larger timeslot length than the aggregated/backbone nodes.

It has been mathematically proven that if the least common multiple of L_u# and L_v# is LCM, OPL is also a multiple of LCM.

Node U may send a detection packet from the end (or head, the process is similar) of an arbitrary timeslot i of port_u2 connected to node V. After a certain link propagation delay (D_propagation), the

packet is received by the incoming port of node V, and i is regarded as the incoming timeslot by V. At this time, the ongoing sending timeslot of port_v1 is j', and there is time T_ij' left before the end of the timeslot j'.

This mapping relationship is termed as:

* <instance OPL, port_u2 slot i, port_v1 slot j', T_ij'>

To avoid confusion, we refer to this mapping relationship as the base timeslot mapping (BTM), as it is independent of the DetNet flows. Later, we will see the timeslot mapping relationship related to DetNet flow, which is the mapping relationship between the outgoing timeslot of port_u2 and the outgoing timeslot of port_v2, which is based on timeslot resource reservation and termed as the forwarding timeslot mapping (FTM).

BTM is generally maintained by node V when processing probe message received from node U. However, node U may also obtain this information from node V, e.g, by an ACK message. The advantage of maintaining BTM by node U is that it is consistent with the unidirectional link from node U to V, so it is more appropriate for node U (rather than V) to advertise it in the network. How to detect BTM and then advertise it in the network (including contoller), will be described in separate documents.

Note that this document does not recommend directly detecting and maintaining BTM between the outgoing timeslot of port_u2 and the ongoing sending timeslot of port_v2 (i.e., the outgoing port of downstream node V), as this is too trivial. In fact, as shown above, maintaining only BTM between the outgoing timeslot of port_u2 and the ongoing sending timeslot of port_v1 (i.e., the incoming port of downstream node V) is sufficient to derive other mapping relationships.

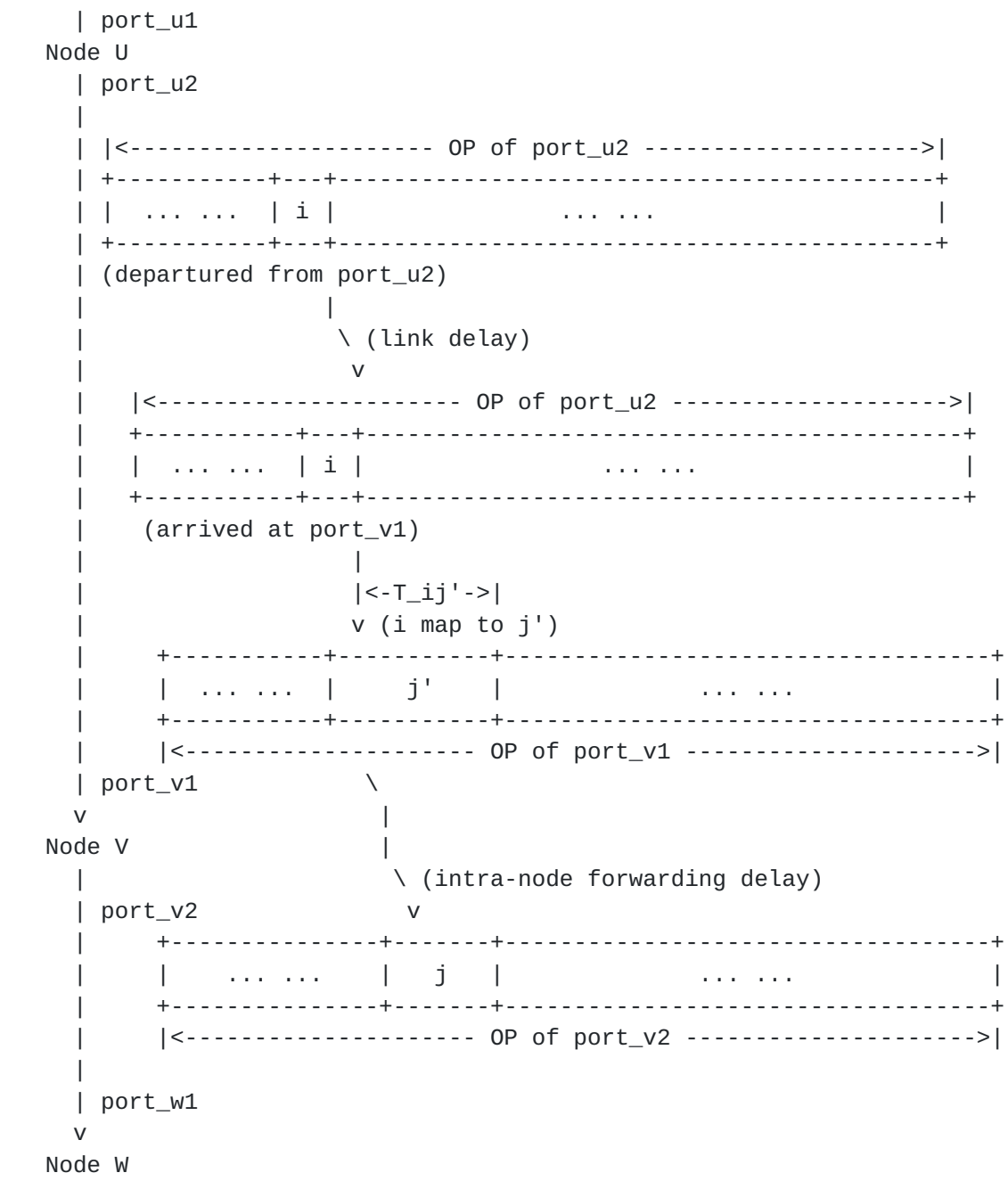

Figure 2: BTM Detection

Based on the above detected BTM, and knowing the intra-node forwarding delay (F) including parsing, table lookup, internal fabric exchange, we can derive BTM between any outgoing timelot x of port_u2 and the ongoing timeslot y of port_v2.

Let t is the offset between the end of the timeslot x of port_u2 and the beginning of the orchestration period of the port_v2.

* t = ((j'+1)*L_v1 - T_ij' + OPL + (x-i)*L_u2 + F) % OPL

Then,

* y = [t/L_v2]

And the time T_xy left before the end of the timeslot y is:

* T_xy = (y+1)*L_v2 - t

This document recommends that the time of each port within the same node must be synchronized, that is, all ports of a node share the same local system time, which is easy to achieve. It is also recommended that the begin time of the orchestration period for all ports within the same node be the same or differ by an integer multiple of OPL, e.g, maintaining a global initial time as the logical begin time for the first round of orchestration period for all ports. Whether node restart or port restart, this initial time should continue to take effect to avoid affecting the timeslot mapping relationship between each node. Depending on the implementation, considering that the initial time may be a historical time that is too far away from the current system time, regular updates may be made to it (e.g, self increasing k*OPL, where k is a natural number) to be closer to the current system time.

3.1.1.2. Deduced by BOM

Figure 3 shows that there are three nodes U, V, and W in turn along the path. Similar to Section 3.1.1.1, it still has L_u2*N_u2 = L_v1*N_v1 = L_v2*N_v2.

Node U may send a detection packet from the head (or end, the process is similar) of the orchestration period of port_u2 connected to node V. After a certain link propagation delay (D_propagation), the packet is received by the incoming port of node V. At this time, there is time P_uv left before the end of the ongoing sending period of port_v1.

This mapping relationship is termed as:

* <instance OPL, port_u2, port_v1, P_uv>

We refer to this mapping relationship as the base orchestration-period mapping (BOM), which it is independent of the DetNet flows.

BOM is generally maintained by node V when processing probe message received from node U. However, node U may also obtain this information from node V, e.g, by an ACK message. The advantage of

maintaining BOM by node U is that it is consistent with the unidirectional link from node U to V, so it is more appropriate for node U (rather than V) to advertise it in the network. How to detect BOM and then advertise it in the network (including contoller), will be described in separate documents.

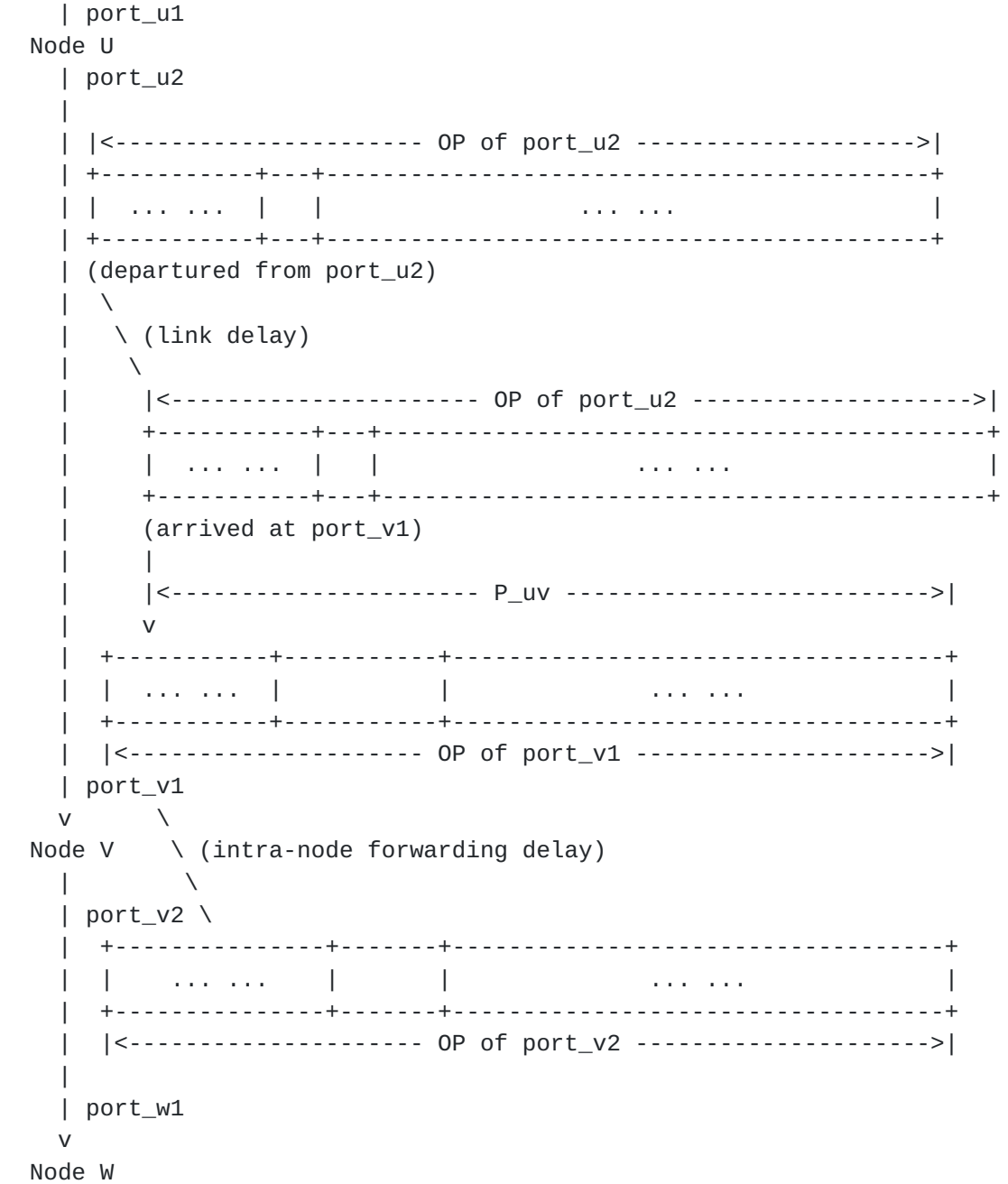

Figure 3: BOM Detection

Based on BOM, and knowing the intra-node forwarding delay (F), we can derive the mapping relationship between any outgoing timelot x of port_u2 and the ongoing timeslot y of port_v2.

Let t is the offset between the end of the timeslot x of port_u2 and the beginning of the orchestration period of the port_v2.

* t = ((x+1)*L_u2 + OPL - P_uv + F) % OPL

Then,

* y = [t/L_v2]

And the time T_xy left before the end of the timeslot y is:

* T_xy = (y+1)*L_v2 - t

3.1.2. Timeslot Resource Definition

The timeslot resources of a link can be represented as the corresponding bit amounts of all timeslots included in an orchestration period. Basically, the link capability should contain the following information:

* Timeslot Length (TL): Represents the length of the timeslot, in units of us. Generally, the length of each timeslot included in the orchestration period is the same.

* Orchestration Period Length (OPL): Represents the length of the orchestration period, in units of us. The orchestration period contains N timeslots, numbered sequentially from 0 to N-1. That is, OPL = N*TL.

* Scheduling Period Length (SPL): Represents the length of the scheduling period, in units of us. The scheduling period contains M timeslots, numbered sequentially from 0 to M-1. That is, SPL = M*TL..

Figure 4 shows the timeslot resource model of the link, with an orchestration period instance consisting of N timeslots numbered from 0 to N-1. The resource information of each timeslot includes the following attributes:

* Timeslot ID: Indicates the NO. of the timeslot in the orchestration period instance. The NO. of the first timeslot is 0, and the NO. of the last timeslot is N-1.

* Maximum Reservable Bursts (MRB): Refers to the maximum amount of bit quota corresponding to this timeslot, with unit of bits. It is a configurable preset value that is related to the service rate (termed as C) and the length of the timeslot (termed as TL), and the Maximum Reservable Bursts should be set to a value not exceeding C*TL. Generally, the Maximum Reservable Bursts of each timeslot included in the orchestration period are all the same.

* Unreserved Bursts (UB): Refers to the amount of unreserved bits reservable corresponding to this timeslot, with unit of bits.

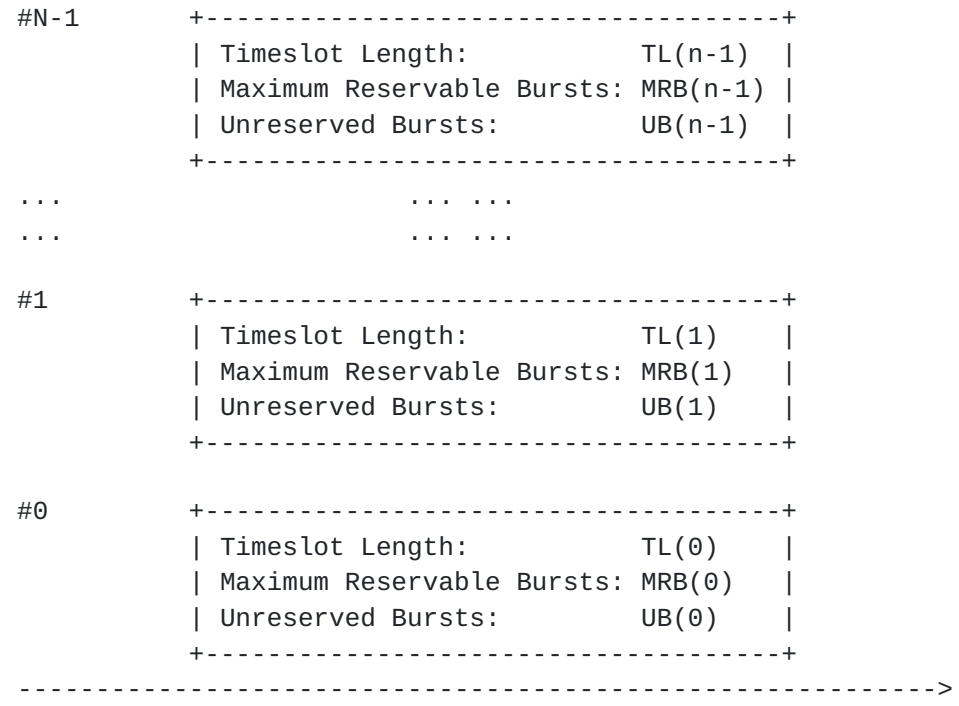

Figure 4: Timeslot Resources Model

The IGP/BGP extensions to advertise the link's capability and timeslot resource is defined in [I-D.peng-lsr-deterministic-traffic-engineering].

3.1.3. Arrival Postion in the Orchestration Period

Generally, a DetNet flow has its TSpec, such as periodically generating traffic of a specific burst size within a specific length of burst interval, which regularly reaches the network entry. The headend executes traffic regulation (e.g, setting appropriate parameters for leaky bucket shaping), which generally make packets evenly distributed within the service burst interval, i.e, there are

one or more shaped sub-burst in the service burst interval. There is an ideal positional relationship between the departure time (when each sub-burst leaves the regulator) and the orchestration period of UNI port, that is, each sub-burst corresponds to an ideal incoming timeslot of UNI port. Based on the ideal incoming timeslot, an ideal outgoing timeslot of NNI port is reserved for the sub-burst.

For example, if a DetNet flow distributes m sub-bursts during the orchestration period, the network entry should maintain m states for that flow:

* <OPL, ideal incoming slot i_1, ideal outgoing slot z_1>

* <OPL, ideal incoming slot i_2, ideal outgoing slot z_2>

* ... ...

* <OPL, ideal incoming slot i_m, ideal outgoing slot z_m>

However, the packets arrived at the network entry are not always ideal, and the departure time from regulator may not be in a certain ideal incoming timeslot. Therefore, an important operation that needs to be performed by the network entry is to determine the ideal incoming timeslot i based on the actual departure time. This can first determine the actual incoming timeslot based on the actual departure time, and then select an ideal incoming timeslot that is closest to the actual incoming timeslot and not earlier than the actual incoming timeslot.

Figure 5 shows, for some typical DetNet flows, the relationship between the service burst interval (SBI) and the length of orchestration period (OPL) of headend, as well as the possible timeslot resource reservation results for these DetNet flows.

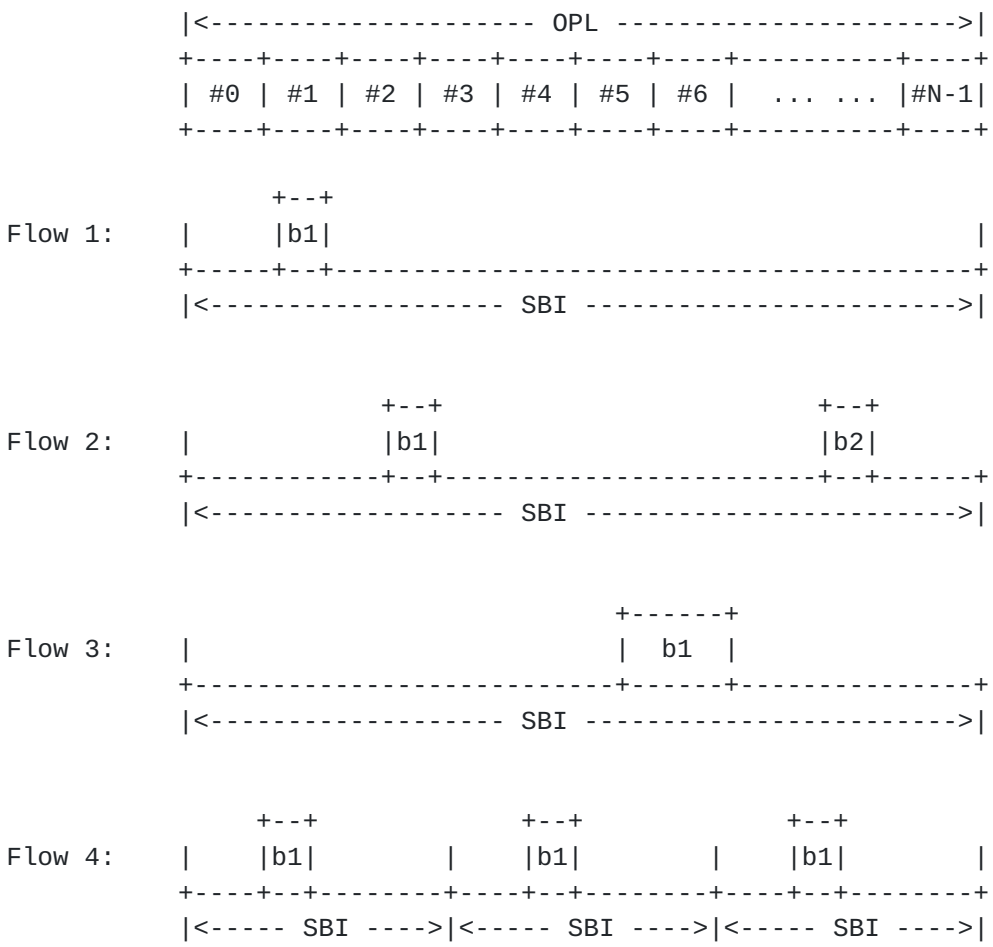

Figure 5: Relationship between SBI and OP

As shown in the figure, the length of service burst intervals for flows 1, 2, 3 is equal to the length of orchestration period, while the length of the service burst interval for flow 4 is only 1/3 of the orchestration period.

* Flow 1 generates a very small single burst amounts within its burst interval, which may reserve timeslot 2 or other subsequent timeslot in the orchestration period;

* Flow 2 generates two small discrete sub-bursts within its burst interval and also be shaped, which may reserve slots 4 and N-1 in the orchestration period for each sub-burst respectively;

* Flow 3 generates a large single burst amount within its burst interval but not be really shaped (due to purchasing a larger burst resource and served by a larger bucket depth), which may also be split to multiple back-to-back sub-bursts and reserve multiple timeslots in the orchestration period, such as timeslots 8 and 9.

* The length of the service burst interval for flow 4 is only 1/3 of the orchestration period. Hence, construct flow 4' with 3 occurrence of the flow 4 within an orchestration period. So flow 4' is similar to flow 2, generating a small amount of three separate sub-bursts within its burst interval. It may reserve timeslots 3, 7, and N-1 in the orchestration period.

Each sub-burst corresponds to a reservation sub-task. For simplicity, each regulated sub-burst in the service burst interval always reserves timeslot resources according to the maximum sub-bust size.

For a specific DetNet flow, to determine how many reservation sub-tasks are required, can be summarized as:

* First, align the service burst interval with the orchestration period of the headend to ensure that the two are of equal length. If the service burst interval is only a fraction of the orchestration period, multiply it several times to obtain the expanded service burst interval to get a new flow'.

* Check how many discrete sub-bursts will be generated during the orchestration period, and for each sub-burst:

  - If the proportion of the sub-burst size to the MRB of a single timeslot does not exceed a specific value, then the sub-burst corresponds to a reservation sub-task;

  - Otherwise, continue to split the sub-burst into multiple sub-sub-bursts (note that each sub-sub-burst must contain a complete packet), so that the proportion of each sub-sub-burst size to the MRB of a single timeslot does not exceed the specific value, and each sub-sub-burst corresponds to a reservation sub-task.

3.1.4. Proccess of Each Reservation Sub-task

Each reservation sub-task contains a separate parameter set, which is used in the process of timeslot resource reservation. Note that this set may be a local information for the path compuation engine (e.g, a controller), or may signal between nodes (e.g, RSVP-TE).

* Total Residence Budget: It is the sum of the residence delay allowed by the DetNet flow within all nodes in the path, which is equal to the end-to-end delay requirement of the DetNet flow minus the propagation delay of all links included in the path.

* Node Residence Budget: It refers to the residence delay budget of the current node traversed during the process of reserving timeslot resources on each node along the path in sequence. A simple way is to divide the Total Residence Budget by the number of nodes included in the path to obtain the average residence delay budget as the Node Residence Budget for each node, or use a specified budget list to specify the residence delay budget for each node separately.

* Accumulated Node Residence Budget: It refers to the accumulated residence delay budget of those nodes that have executed resource reservation.

* Accumulated Node Residence Evaluation: It refers to the accumulated evaluation value of the residence delay of nodes that have executed resource reservation. The residence delay evaluation value of a node refers to the residence delay evaluation value calculated based on the delay formula (see below) when the node actually reserves a certain outgoing timeslot for the reservation sub-task. Generally, if a node is able to reserve the expected outgoing timeslot according to its residence delay budget, the residence delay evaluation value does not differ from the residence delay budget. However, in some cases, due to insufficient resources in the expected timeslot, resources have to be reserved in the timeslot adjacent to the expected timeslot, which can lead to a difference between the residence delay evaluation value and the budget value.

* Accumulated Node Residence Deviation: It is equal to the Accumulated Node Residence Budget minus the Accumulated Node Residence Evaluation.

* Node Residence Budget Adjustment: It is equal to the Node Residence Budget plus the Accumulated Node Residence Deviation.

The usage for the above parameter set is:

* For specific reservation sub-task, determine the Node Residence Budget for each node in the path, which can be taken from the average residence delay budget per node or the specified budget list.

* From the headend to the endpoint, on each node's outgoing port in sequence, reserve outgoing timeslot resources based on the Node Residence Budget Adjustment, to let the residence delay evaluation value of the node obtained from the reserved outgoing timeslot be equal to or close to the Node Residence Budget Adjustment.

  - On the headend, the Accumulated Node Residence Deviation is the initial value of 0. Therefore, the Node Residence Budget Adjustment is equal to the Node Residence Budget.

  - On any other nodes, the Accumulated Node Residence Deviation is generally not 0. If the residence delay evaluation value of the node obtained from the reserved outgoing timeslot be equal to the Node Residence Budget Adjustment, it will cause the Accumulated Node Residence Deviation faced by the downstream node in the path to be 0 again.

Note that the above parameter set is only an implementation choice and is not mandatory. There may be more intelligent path calculation methods available.

3.1.4.1. Resource Reservation on the Ingress Node

On the headend H, as mentioned above, each sub-burst corresponds to an ideal incoming timeslot i of UNI port. After the intra-node forwarding delay (F), the end of the incoming timeslot i reaches the outgoing port, the timeslot currently in the sending state (i.e., the ongoing sending timeslot of NNI port) is j, and there is time T_ij left before the end of the timeslot j.

The outgoing timeslot reserved for the sub-burst by the headend is offset by o (>=1) timeslots after timeslot j, which means the outgoing timeslot is z = (j+o)%N_h2, where N_h2 is the number of timeslots in the orchestration period of NNI port.

Note that o must be less than M. (where o is the offset and M is the number of timeslot in the scheduling period as mentioned in Section 3.1.2)

Thus, on the headend H the residence delay evaluation value obtained from the reserved outgoing timeslot z is:

Best Node Residence Evaluation = F + T_ij + (o-1)*L_h2

Worst Node Residence Evaluation = F + L_h1 + T_ij + o*L_h2

Average Node Residence Evaluation = F + T_ij + (L_h1 + (2o-1)*L_h2)/2

where, L_h1 is the timeslot length of UNI port, L_h2 is the timeslot length of NNI port.

The Best Node Residence Evaluation occurs when the sub-burst is at the end of the ideal incoming timeslot i, and sent at the head of outgoing timeslot z. The Worst Node Residence Evaluation occurs when the sub-burst is at the head of the ideal incoming timeslot i, and sent at the end of outgoing timeslot z. The delay jitter within the headend is (L_h1 + L_h2). However, the jitter of the entire path is not the sum of the jitters of all nodes.

Depending on the implementation, the above Best Node Residence Evaluation, Worst Node Residence Evaluation, or Average Node Residence Evaluation can be used to compare with the Node Residence Budget Adjustment, so that when selecting the appropriate outgoing timeslot z, the two are equal or nearly equal, and the corresponding Unreserved Burst resources of the outgoing timeslot z meet the burst demand of the sub-burst. However, this document suggests using the Average Node Residence Evaluation to compare with the Node Residence Budget Adjustment, because the characteristic of the forwarding behavior based on TQF is that adjacent nodes on the path will not simultaneously face the best or worst residency delay.

Note that there is a runtime jitter (i.e., the resource reservation process on the control plane is not aware of it), as mentioned earlier, which depends on the deviation between the actual incoming timeslot i' and the ideal incoming timeslot i. Assuming that i = (i'+e)%N_h1, where e is the deviation, N_h1 is the number of timeslots in the orchestration period of UNI port, then the additional runtime jitter is e*L_h1, that should be carried in the packet to eliminate jitter at the network egress.

3.1.4.2. Resource Reservation on the Transit Node

On the transit node V, as described in Section 3.1.1, there is a timeslot mapping relationship between the outgoing timeslot i of port_u2 and the ongoing sending timeslot j of port_v2, and there is time T_ij left before the end of the timeslot j.

For a specific sub-task, assume that outgoing timeslot i is reserved for it on port_u2, and the outgoing timeslot z reserved for it on port_v2 is offset by o (>=1) timeslots after timeslot j, i.e., z = (j+o)%N_v2, where N_v2 is the number of timeslots in the orchestration period of port_v2.

Note that o must be less than M.

Thus, on the transit node V the residence delay evaluation value obtained from the reserved outgoing timeslot z is:

Best Node Residence Evaluation = F + T_ij + (o-1)*L_v2

Worst Node Residence Evaluation = F + T_ij + L_u2 + o*L_v2

Average Node Residence Evaluation = F + T_ij + (L_u2+(2o-1)*L_v2)/2

where, L_u2 and L_v2 is the timeslot length of port_u2 and port_v2 respectively.

The Best Node Residence Evaluation occurs when the packet is received at the end of incoming timeslot i and sent at the head of outgoing timeslot z; The Worst Node Residence Evaluation occurs when the packet is received at the head of incoming timeslot i and sent at the end of outgoing timeslot z. The delay jitter within the node is (L_u2 + L_v2). However, the jitter of the entire path is not the sum of the jitters of all nodes.

Depending on the implementation, the above Best Node Residence Evaluation, Worst Node Residence Evaluation, or Average Node Residence Evaluation can be used to compare with the Node Residence Budget Adjustment, so that when selecting the appropriate outgoing timeslot z, the two are equal or nearly equal, and the corresponding Unreserved Burst resources of the outgoing timeslot z meet the burst demand of the sub-burst. However, this document suggests using the Average Node Residence Evaluation to compare with the Node Residence Budget Adjustment, because the characteristic of the forwarding behavior based on TQF is that adjacent nodes on the path will not simultaneously face the best or worst residency delay.

3.1.4.3. Resource Reservation on the Egress Node

Generally, for the deterministic path carrying the DetNet flow, the flow needs to continue forwarding from the outgoing port of the egress node to the client side, and also faces the issues of queueing. However, the outgoing port facing the client side is not part of the deterministic path. If it is necessary to continue supporting TQF mechanism on that port, timeslot resources should be reserved on the higher-level DetNet path (an overlay path) using the above reservation method. In this case, the underlay DetNet path will serve as a virtual link of the overlay path, providing a deterministic delay performance.

Therefore, for deterministic paths, the residence dalay evaluation value on the egress node is only contributed by the forwarding delay (F) including parsing, table lookup, internal fabric exchange, etc.

3.1.4.4. End-to-end Delay and Jitter

Figure 6 shows that a path from headend P1 to endpoint E, for each node Pi, the timeslot length of the outgoing port is L_i, the intra-node forwarding delay is F_i, the remaining time from the end of the mapped ongoing sending timeslot is T_i, the number of timeslots offset by outgoing timeslot relative to ongoing sending timeslot is o_i, especially on node P1 the timeslot length of UNI is L_h, then the end to end delay can be evaluted as follows (not including link propagation delay):

Best E2E Delay = sum(F_i+T_i+o_i*L_i, for 1<=i<=n) - L_n + F_e

Worst E2E Delay = sum(F_i+T_i+o_i*L_i, for 1<=i<=n) + L_h + F_e

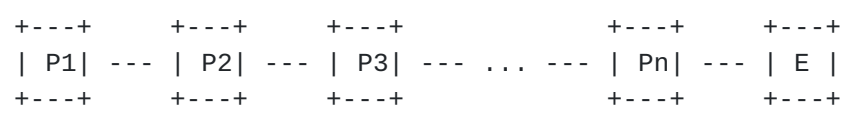

Figure 6: TQF Forwarding Path

The Best E2E Delay occurs when the sub-burst is at the end of the ideal incoming timeslot and sent at the head of outgoing timeslot of each node pi. The Worst E2E Delay occurs when the sub-burst is at the head of the ideal incoming timeslot and sent at the end of outgoing timeslot of each node Pi. The E2E delay jitter is (L_h + L_n).

3.2. Timeslot Resource Access in Data-plane

The headend of the path needs to maintain the timeslot resource information with the granularity of sub-burst, so that each sub-burst of the DetNet flow can access the mapped timeslot resources. However, the intermediate node does not need to maintain this mapping state. The intermediate node only access the timeslot resources based on the timeslot id carried in the packets or indicated by FIB entries.

Note that the incoming and outgoing timeslots mentioned here are both timeslot id within the orchestration period.

[I-D.pb-6man-deterministic-crh] defined a method to carry the stack of timeslot id in the IPv6 packets.

By default, the following subsections discuss on-time scheduling behavior.

3.2.1. Round Robin Queue: Conversion of Timeslot ID

Figure 1 shows that the scheduling period implemented on the data plane is not completely equivalent to the orchestration period of the control plane. The scheduling period includes M timeslots (from 0 to M-1), while the orchestration period includes N timeslots (from 0 to N-1). In the orchestration period, from timeslot 0 to M-1 is the first scheduling period, from timeslot M to slot 2M-1 is the second scheduling period, and so on. Therefore, it is necessary to convert the outgoing timeslot of the orchestration period to the target timeslot of the scheduling period, and insert the packet to the round robin queue corresponding to the target timeslot for transmission.

A simple conversion method is:

* target scheduling timeslot = outgoing timeslot % M

This is safe because during resource reservation, o < M is always followed, and N is an integer multiple of M.

According to the timeslot resource reservation process mentioned above, when the sub-burst corresponding to any outgoing timeslot (e.g, z) arrived at the outgoing port of any node of the path, the ongoing sending timeslot (e.g, j) in the orchestration period of the outgoing port must be offset by o before the outgoing timeslot (z), and meet o < M, which means that the sub-burst does not randomly arrive at this node, but strictly conform to the time so that when it reaches the outgoing port, it will definitely fall into the ongoing sending timeslot (j).

Next, we briefly demonstrate that the sub-burst that arrives at the outgoing port during the ongoing sending timeslot (j) can be safely inserted into the corresponding queue in the scheduling period, and that queue will not overflow.

Assuming that each timeslot in the orchestration period has a virtual queue, the length of the virtual queue is the MRB of that timeslot. For example, termed the virtual queue corresponding to the outgoing timeslot z as queue-z, the packets that can be inserted into queue-z may only come from the following bursts:

During the ongoing sending timeslot j = (z-M+1+N)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z) according to o = M-1.

During the ongoing sending timeslot j = (z-M+2+N)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z) according to o = M-2.

... ...

During the ongoing sending timeslot j = (z-1+N)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z) according to o = 1;

The total reserved amount of all these bursts does not exceed the MRB of the outgoing timeslot (z).

Then, when the ongoing sending timeslot changes to z, queue-z will be sent and cleared. In the following time, starting from timeslot z+1 to the last timeslot N-1 in the orchestration period, there are no longer any packets inserted into queue-z. Obviously, this virtual queue is a great waste of queue resources. In fact, queue-z can be reused by the subsequent outgoing timeslot (z+M)%N. Namely:

During the ongoing sending timeslot j = (z+1)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z+M)%N according to o = M-1.

During the ongoing sending timeslot j = (z+2)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z+M)%N according to o = M-2.

... ...

During the ongoing sending timeslot j = (z+M-1)%N, the bursts that arrive at the outgoing port, that is, these bursts may reserve the outgoing timeslot (z+M)%N according to o = 1.

The total reserved amount of all these bursts does not exceed the MRB of the outgoing timeslot (z+M)%N.

It can be seen that queue-z can be used by any outgoing timeslot (z+k*M)%N, where k is a non negative integer. By observing (z+k*M)%N, it can be seen that the minimum z satisfies 0<= z< M, that is, the entire orchestration period actually only requires M queues to store packets, which are the queues corresponding to M timeslots in the scheduling period. That is to say, the minimum z is the timeslot id in the scheduling period, while the outgoing timeslot (z+k*M)% N is the timeslot id in the orchestration period. The latter obtains the former by moduling M, which can then access the queue corresponding to the former. In short, the reason why a queue can store packets from multiple outgoing timeslots without being overflowed is that the packets stored in the queue earlier (more than M timeslots ago) have already been sent.

3.2.2. PIFO: Directly Using Outgoing Timeslots

Figure 1 also shows that the scheduling period may also be instantiated by a PIFO queue. The buffer cost of PIFO queue is the same as that of round robin queues. It can directly use the begin time of the outgoing timeslot z as the rank of the packet and insert the packet into the PIFO for transmission.

* rank = z.begin

Here, the outgoing timeslot z refers to the outgoing timeslot z that is after the arrival time at the scheduler and closest to the arrival time.

The rule of the on-time scheduling mode is that if the PIFO is not empty and the rank of the head of queue is equal to or earlier than the current system time, the head of queue will be sent; otherwise, not.

4. Global Timeslot ID

The outgoing timeslots we discussed in the previous sections are local timeslots style for all nodes. This section discusses the situation based on global timeslot style.

Global timeslot style refers to that all nodes in the path are identified with the same timeslot id, which of course requires all nodes to use the same timeslot length. The advantages are that the resource reservation based on global timeslots is simple, always reserving a specified outgoing timeslot for the DetNet flow. There is no need to establish FTM on each node or carry FTM in packets. The packet only needs to carry the unique global timeslot id. However, the disadvantage is that the latency performance of the path may be large, which depends on BOM between the adjacent nodes.

Another disadvantage is that the success rate of finding a path that matches the service requirements is not as high as local timeslot style.

Global timeslot style requires that the orchestration period is equal to the scheduling period, mainly considering that arrival packets with any global timeslot id can be successfully inserted into the corresponding queue without overflow. However, as the ideal design goal is to keep the scheduling period less than the orchestration period, further research is needed on other methods (such as basically aligning orchestration period between nodes), to ensure that packets with any global timeslot id can queue normally when the scheduling period is less than the orchestration period.

Compared to the local timeslot style, global timeslot style means that the incoming timeslot i must map to the outgoing timeslot i too. As the example shown in Figure 7, each orchestration period contains 6 timeslots. Node V has three connected upstream nodes U1, U2, and U3. During each hop forwarding, the packet accesses the outgoing timeslot corresponding to the global timeslot id and forwards to the downstream node with the global timeslot id unchanged. For example, U1 sends some packets with global slot-id 0, termed as g0, in the outgoing timeslot 0. The packets with other global slot-id 1~5 are similarly termed as g1~g5 respectively. The figure shows the scheduling results of these 6 batches of packets sent by upstream nodes when node V continues to send them.

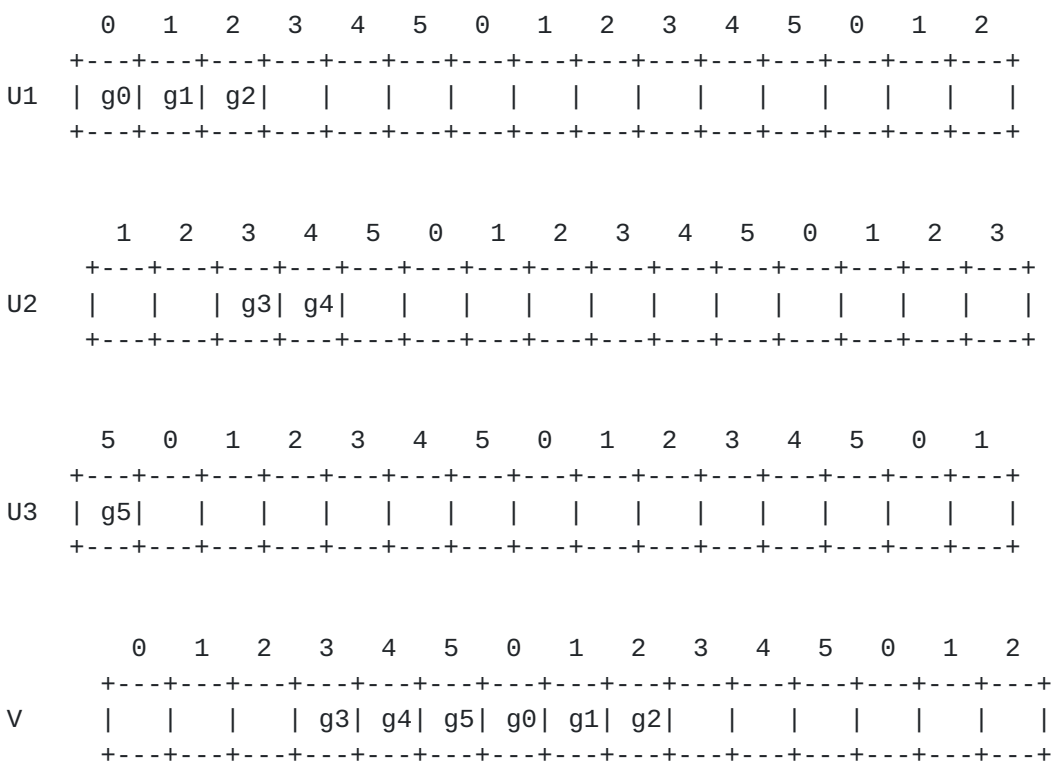

Figure 7: Global Timeslot Style Example

In this example:

* BTM between the outgoing timeslot of U1 and the ongoing sending timeslot of V is i -> i, so the reserved outgoing timeslot for the incoming timeslot i is i+6 (i.e., belongs to next round of orchestration period).

* BTM between the outgoing timeslot of U2 and the ongoing sending timeslot of V is i -> i-1, so the reserved outgoing timeslot for the incoming timeslot i is i (i.e., belongs to current round of orchestration period).

* BTM between the outgoing timeslot from U3 and the ongoing sending timeslot of V is i -> i+1, so the reserved outgoing timeslot for the incoming timeslot i is i+6 (i.e., belongs to next round of orchestration period).

It can be seen that packets from U1 and U3 has large residency delay in the node V, while packets from U2 has small residency delay in the node V.

It should be noted that for the orginal mapping relationship i -> i or i -> i+1, the packets need to be stored in a buffer prior to the TQF scheduler (such as the buffer on the input port side) for a fixed latency (such as serveral timeslots) and then released to the scheduler. Instead, directly inserting the queue may cause queue overflow. This fixed-latency buffer is only introduced for specific upstream nodes. It can be determined according to the initial detection result of BTM between the outgoing timeslot of the upstream node and the ongoing sending timeslot of this node. If the original detection result is i -> i or i -> i+1, it needs to be introduced, otherwise not. After the introduction of fixed-latency buffer, the new detection result of BTM will no longer be i -> i or i -> i+1.

For the headend, the residence delay is similar to Section 3.1.4.1, except that determining the offset o is simpler. Suppose that for the ideal incoming timeslot i (note that at the headend this incoming timeslot i is not the reserved timeslot resource), the ongoing sending timeslot of the ougtoing port is j, and the sub-burst reserve global timeslot z, then, o equals (N+z-j)%N.

For transit nodes, the residence delay is similar to Section 3.1.4.2, except that determining the offset o is simpler. Suppose that for the incoming timeslot z, the ongoing sending timeslot of the ougtoing port is j, and the sub-burst continues to reserve global timeslot z, then, o equals (N+z-j)%N.

The end-to-end delay equation is similar to Section 3.1.4.4.

5. Summary of Timeslot Style

Depending on the strategy of reserving timeslot resources, different timeslot styles will be presented, as shown in the table below.

Strategy	Timeslot Style	Referrence
Flexible o (1<=o<M)	Local timeslot style	section 3.1.4
Constant o (o=(N+i-j)%N)	Global timeslot style	section 4
Constant o (o=1)	ECQF	[ECQF]

Figure 8: Timeslot Styles

A local policy may be configured in the path computation engine to use which strategy on different nodes, as long as the calculated E2E delay meet the flow's requirement. For example, it is possible to use strategy "constant o=1" on all transit nodes and use strategy "flexible o" on ingress node.

Based on the same topology, the success rate of path calculation for strategy "flexible o" applied on transit nodes is higher than that for strategy "constant o" applied.

6. In-time Scheduling

So far, in the TQF mechanism presented above, both for local timeslot style and global timeslot style, the goal is to reserve a fixed outgoing timeslot for the sub-burst in the orchestration period, and just send the sub-burst in that timeslot. This is on-time scheduling.

In this section, we discuss another scheduling variant of TQF, i.e., in-time scheduling. In this case, timeslot resources are still reserved based on delay requirement, but in actual forwarding, packets do not necessarily have to wait until the reserved outgoing timeslot for sending.

As is known, in-time scheduling may cause burst accumulation, so that scheduling period implemented with limited amount of round robin queues is not suitable for this purpose, while PIFO with excess length is more suitable. [SP-LATENCY] provides guidance for evaluating excess buffer requirements.

Similar to Section 3.2.2, it can directly use the begin time of the outgoing timeslot z as the rank of the packet and insert the packet into the PIFO for transmission. However, due to in-time scheduling behavior, the outgoing timeslot z may not be the outgoing timeslot z that is after the arrival time at the scheduler and closest to the arrival time, instead, it may be an outgoing timeslot z far away from the arrival time.

A time deviation (E) may be carried in the packet to help determine the outgoing timeslot z.

On the headend node:

* E initially equals to the begin time of the ideal incoming timeslot minus the actual departure time from the regulator.

* Use the result of "departure time + E" (note that it is just the begin time of the ideal incoming timeslot, and the main purpose here is to describe how E works) to determine the expected outgoing timeslot z that is after this result and closest to this result.

* rank = z.begin

* When the packet leaves the headend, E is updated to z.begin minus the actual sending time from the PIFO. The updated E will be carried in the sending packet.

On the transit node:

* Obtain E from the received packet.

* Use the result of "arrival time + E" to determine the expected outgoing timeslot z that is after this result and closest to this result. Here, the arrival time is the time that the packet arrived at the scheduler.

* rank = z.begin

* When the packet leaves the headend, E is updated to z.begin minus the actual sending time from the PIFO. The updated E will be carried in the sending packet.

The rule of the in-time scheduling mode is that as long as the PIFO is not empty, packets are always obtained from the head of queue for transmission.

In summary, the in-time scheduling with the help of time deviation (E), can suffer from the uncertainty caused by burst accumulation, and it is recommended only deployed in small networks, i.e., a limited domain with a small number of hops, where the burst accumulation issue is not serious; The on-time scheduling is recommended to be used in large networks.

7. Queue Design

7.1. Round Robin Queues

Round robin queues operate in on-time scheduling mode by default.

The number of round robin queues should be designed according to the number of timeslots included in the scheduling period. Each timeslot corresponds to a separate queue, in which the buffered packets must be able to be sent within a timeslot.

The length of the queue, i.e., the total number of bits that can be reserved or sent for a timeslot, equals to the allocated bandwidth of the corresponding OP instance (see Section 8) multiplied by the timeslot length.

7.1.1. Full Queues

Case: 1-to-1 mapping between the orchestration period timeslot and the scheduling period timeslot.

When the scheduling period length is equal to the orchestration period length, the node will implement full queues. The advantage is that the actual forwarding resources are the same view as the resources used for reservation, so that the resource reservation process is simple (e.g, the global timeslot style). However, the disadvantage is that because the scheduling period is generally large to cover all services requirements, the number of queues maintained by the node will be large.

For example, if the total length of all queues supported by the hardware is 4G bytes, the queue length corresponding to a timeslot of 10us at a port rate of 100G bps is 1M bits, then a maximum of 32K timeslot queues can be provided, and the maximum length of the orchestration period supported is 320ms. However, considering the queue resource requirements of other non-DetNet flows, the TQF function can only use some of the queue resources, such as 10K~20K queues. In this case, the length of the orchestration period supported by the node may be 100~200 ms.

7.1.2. Non-full Queues

Case: Many-to-1 mapping between the orchestration period timeslot and the scheduling period timeslot.

When the length of the scheduling period is less than the length of the orchestration period, the node will implement a non-full queues. The advantages and disadvantages are opposite to the full queues option. The actual forwarding resources are inconsistent with the view of the resources reservation. But the number of queues maintained by the node is small.

7.2. PIFO Queue

PIFO can be configured to operate in either in-time or on-time scheduling mode.

For on-time mode, the buffer cost is the same as that of round robin queues. The rank of the packet equals to the begin time of the outgoing timeslot z, that can be safely obtained a "z" closet to the arrival time at the scheduler.

For in-time mode, excess buffers are required to cope with burst accumulation. [SP-LATENCY] provides guidance for evaluating excess buffer requirements. The rank of the packet equals to the begin time of the expected outgoing timeslot z, that can be obtained based on a "z" closet to the result of arrival time at the scheduler plus time deviation E.

8. Multiple Orchestration Periods

A single orchestration period may not be able to cover a wide range of service needs, such as some with a burst interval of microseconds, while others have a burst interval of minutes or even larger. When using a single orchestration period to simultaneously serve these services, the timeslot length must be microseconds, but the orchestration period length is minutes or more, resulting in the need to include a large number of timeslots in the orchestration period. The final result is a proportional increase in the number of queues required for the scheduling period (to avoid the potential timeslot conflicts).

Multiple orchestration periods each with different length may be provided by the network. A TQF enabled link can be configured with multiple TQF scheduling instances each corresponding to specific orchestration period length. For simplicity, the orchestration period length itself can be used to identify a specific instance.

For example, one orchestration period length is 300 us, termed as OPL-300us, which is the LCM of the burst interval of the set of flows served. Another orchestration period length is 100 ms, termed as OPL-100ms, which is the LCM of the burst interval of another set of flows served. Each orchestration period instance has its own timeslot length. The timeslot length of a long orchestration period instance should be longer than that of a short orchestration period instance, and the former is an integer multiple of the latter. But the long orchestration period itself may not necessarily be an integer multiple of the short orchestration period.

As shown in Figure 9, both link-a and link-b are configured with n orchestration period instances, with the corresponding orchestration period lengths OPL_1, OPL_2, ..., OPL_n in descending order. For each orchestration period length OPL_i, the bandwidth resource allocated is BW_U_i for node U (or BW_V_i for node V), and the timeslot length is TL_U_i for node U (or TL_V_i for node V). For

each TQF enabled link, the sum of bandwidth resources allocated to all orchestration period instances must not exceed the total bandwidth of the link.

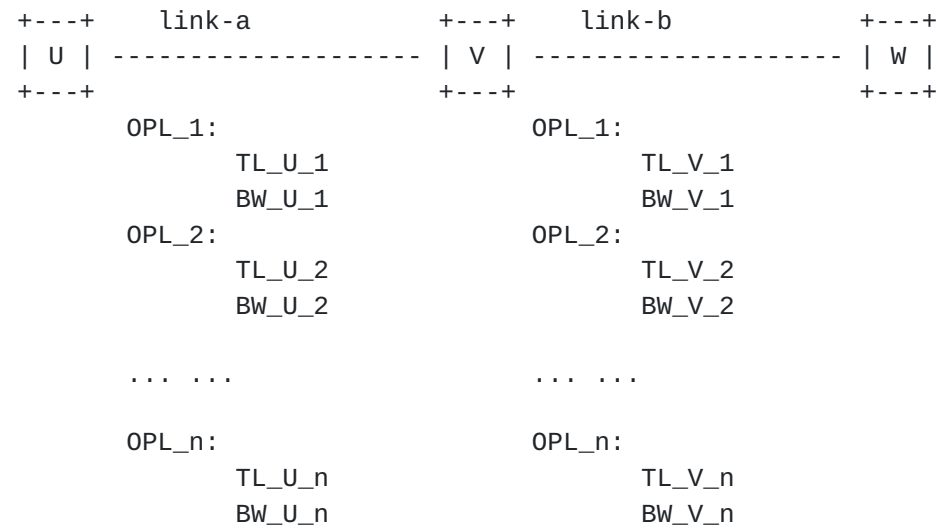

Figure 9: Multiple OP Instances

Due to the fact that long orchestration periods serve DetNet flows with large burst intervals, for a given burst size, the larger the burst interval, the less bandwidth consumed by the DetNet flow. Therefore, it is recommended that the bandwidth resources allocated to long orchestration period instances are less than those allocated to short orchestration period instances, which is also beneficial for reducing the queue length required for long orchestration period instances.

Interworking between different nodes is based on the same orchestration period instance. That means that the timeslot mapping described in Section 3.1.1 should be maintained in the context of the specific orchestration period instance, and the timeslot resource reservation along the path for a sub-task should also be in the context of the specific orchestration period instance. The orchestration period length should be carried in the forwarding packets to let the DetNet flow to access the timeslot resources corresponding to that orchestration period instance.

If round robin queues are used, each orchestration period instance has its own separate queue set. Time division multiplexing scheduling is based on the granularity of the minimum timeslot length of all instances. Within each time unit of this granularity, the queues in the sending state of all instances are always scheduled in the order of OPL_1, OPL_2, ..., OPL_n.

If PIFO queue is used, all orchestration period instances may share a single PIFO queue.

9. Admission Control on the Headend

On the network entry, traffic regulation must be performed on the incoming port, so that the DetNet flow does not exceed its T-SPEC such as burst interval, burst size, maximum packet size, etc. This kind of regulation is usually the shaping using leaky bucket combined with the incoming queue that receives DetNet flow. A DetNet flow may contain discrete multiple sub-bursts within its periodic burst interval. The leaky bucket depth should be larger than the maximum packet size, and should be consistent with the reserved burst resources required for the maximum sub-burst.

The scheduling mechanism described in this document has a requirement on the arrival time of DetNet flows on the network entry. It is expected that the distribution of sub-bursts (after regulation) of the DetNet flow will always appear in an ideal position within the orchestration period of UNI port. Based on this ideal position, any packets of the DetNet flow will be matched to the sub-burst forwarding state that contains the ideal incoming timeslot and corresponding reserved outgoing timeslot. Note that the network entry may maintain multiple sub-burst forwarding states for a single DetNet flow, due to many sub-bursts within the service burst interval.

For example, the network entry may maintain up to 3 sub-burst forwarding states for a flow. Ideally, all packets of this flow are split into 3 sub-bursts after regulation, each sub-burst matching one of the states. Here, 3 is the maximum sub-bursts for this flow, and it does not always contain so many bursts within the burst interval during actual sending.

For a specific sub-burst, some amount of deviation (i.e., the deviation between the actual incoming timeslot and the ideal incoming timeslot) is permitted. Generally, the headend will select an ideal incoming timeslot closet to the actual incoming timeslot for the packet.

For on-time scheduling, the position deviation should not exceed o-1 for late arrival case, or M-o-1 for early arrival case, where o is the offset between the reserved outgoing timeslot and ongoing sending timeslot as mentioned above. Intuitively, large o can tolerate large late arrival deviations, while small o (or large M even for large o) can tolerate large early arrival deviations.

This position deviation limitation is beneficial for on-time scheduling, to achieve the ideal design goal that scheduling period is smaller than the orchestration period, and packets can always be successfully inserted into the scheduling queue without conflicts. For example, there may contain one or more scheduling periods between the departure time from the regulator and the choosed ideal incoming timeslot, and therefore there is an overflow risk when inserting packets into the queue based on the corresponding ideal outgoing timeslot z at the departue time.

Otherwise, for randomly arriving DetNet flows, it can be supported by taking a large M (or even M = N) (option-1) to accommodate random arrival, or it can be supported by introducing an explicit buffer put before the scheduler on the network entry to let the arrival time always meet the fixed position (option-2).

* Note that due to randomness of arrival time, the packet may just miss the scheduling (or arrive too earlier) and need to wait in the scheduling queue (in the case of option-1) or the explicit buffer (in the case of option-2) for the next orchestration period.

For in-time scheduling, the position deviation should not exceed o-1 for late arrival case. We only focus on late arrivals here, as in-time scheduling naturally handles early arrivals. If the late arrival exceed the above limitation, the sub-burst may need to be sent during the next orchestration period in the worst case, or may be lucky to be scheduled immediately.

Note that the position deviation is a runtime latency during forwarding, and the resource reservation process on the control plane is not aware of it. It should be carried in the packet to eliminate jitter at the network egress on demand. Please refer to [I-D.peng-detnet-policing-jitter-control] for the elimination of jitter caused by policing delay on the network entry node. The runtime position deviation should be considered as a part of policing delay.

10. Frequency Synchronization

The basic explanation for frequency synchronization is that the crystal frequency of the hardware is consistent, which enables all nodes in the network to be in the same inertial frame and have the same time lapse rate. This is a prerequisite for all latency based scheduling mechanisms. This frequency synchronization mechanism, such as IEEE 1588-2008 Precision Time Protocol (PTP) [IEEE-1588] and synchronous Ethernet (syncE) [syncE], is not within the scope of this document.

Sometimes, people also refer to the frequency asynchrony as the timeslot rotation frequency difference caused by different node configurations with different timeslot lengths. This document supports the interconnection between nodes with this type of frequency asynchrony.

11. Evaluations

This section gives the evaluation results of the TQF mechanism based on the requirements that is defined in [I-D.ietf-detnet-scaling-requirements].

Requirements	Evaluation	Notes
3.1 Tolerate Time Asynchrony	Yes	No time sync needed, only need frequency sync (3.1.3).
3.2 Support Large Single-hop Propagation Latency	Yes	The timeslot mapping covers any value of link propagation delay.
3.3 Accommodate the Higher Link Speed	Partial	The higher the service rate, the more buffer needed for the same timeslot length.
3.4 Be Scalable to the Large Number of Flows and Tolerate High Utilization	Yes	Multiple OPL instance, each for a set of serivce flows, without overprovision. Utilization may reach 100% link bandwidth. The unused bandwidth of the timeslot can be used by best-effot flows. Calculating paths is NP-hard.
3.5 Tolerate Failures of Links or Nodes and Topology Changes	N/A	Independent of queueing mechanism.
3.6 Prevent Flow Fluctuation	Yes	Flows are permitted based on timeslot reservation, isolated from each other through timeslots.

3.7 Be scalable to a Large Number of Hops with Complex Topology	Yes	E2E latency is liner with hops, from ultra-low to low latency by multiple OPL. E2E jitter is low by on-time mode. Calculating paths is NP-hard.
3.8 Support Multi-Mechanisms in Single Domain and Multi-Domains	N/A	Independent of queueing mechanism.

Figure 10: Evaluation for Large Scaling Requirements

11.1. Examples

This section will describe the example of how the TQF mechanism supports DetNet flows with different latency requirements. As shown in Figure 11:

* Network transmission capacity: each link has rate 10 Gbps. Assuming the service rate of TQF scheduler allocate the total port bandwidth.

* TSpec of each flow, maybe:

  - burst size 1000 bits, SBI 1 ms, and average arrival rate 1 Mbps.

  - or, burst size 1000 bits, SBI 100 us, and average arrival rate 10 Mbps.

  - or, burst size 1000 bits, SBI 100 us, and average arrival rate 100 Mbps.

  - or, burst size 10000 bits, SBI 10 ms, and average arrival rate 1 Mbps.

  - or, burst size 10000 bits, SBI 1 ms, and average arrival rate 10 Mbps.

  - or, burst size 10000 bits, SBI 100 us, and average arrival rate 100 Mbps.

* RSpec of each flow, maybe:

  - E2E latency 100us, and E2E jitter less than 10us or 100us.

- or, E2E latency 200us, and E2E jitter less than 20us or 200us.
- or, E2E latency 300us, and E2E jitter less than 30us or 300us.
- or, E2E latency 400us, and E2E jitter less than 40us or 400us.
- or, E2E latency 500us, and E2E jitter less than 50us or 500us.
- or, E2E latency 600us, and E2E jitter less than 60us or 600us.
- or, E2E latency 700us, and E2E jitter less than 70us or 700us.
- or, E2E latency 800us, and E2E jitter less than 80us or 800us.
- or, E2E latency 900us, and E2E jitter less than 90us or 900us.
- or, E2E latency 1000us, and E2E jitter less than 100us or 1ms.

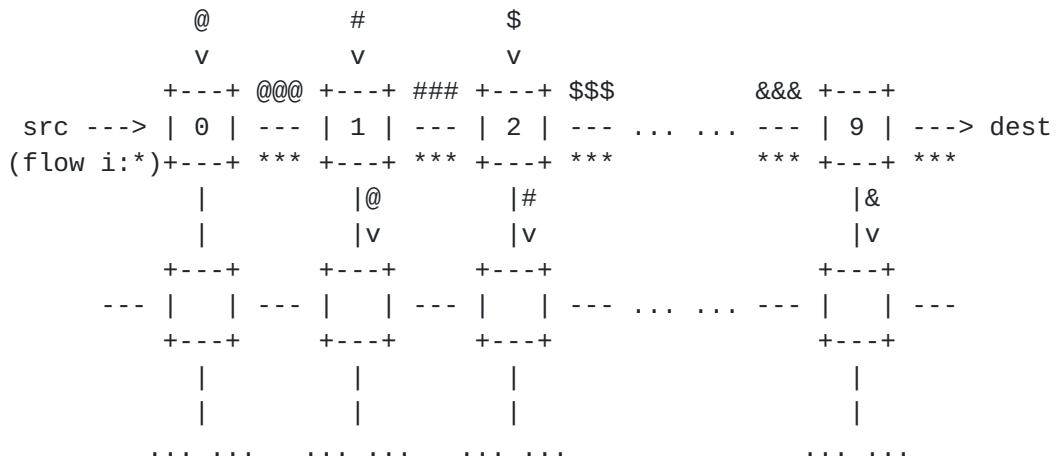

Figure 11: Common Topology Example

For the observed flow i (marked with *), its TSpec and RSpec may be any of the above. Assuming that the path calculated by the controller for the flow i passes through 10 nodes (i.e., node 0~9). Especially, at each hop, flow i may conflict with other deterministic flows, also with similar TSpec and RSpec as above, originated from other sources, e.g, conflicts with flow-set "@" at node 0, conflicts with flow-set "#" at node 1, etc.

For each link along the path, it may configure OPL-10ms instance with allocated bandwidth 10 Gbps, containing 1000 timeslots each with length 10us. Assuming no link propagation delay and intra node forwarding delay, if flow i reserve outgoing timeslot by o=1, it can ensure an E2E latency of 100us (i.e., o * TL * 10 hops), and jitter of 20us(on-time mode) or 100us (in-time mode). The reservation by other o values is similar.

The table below shows the possible supported service scales. As flows arrived synchronously, the reservation for each timeslot in the orchestration period may be caused by any value of o. For example, if the ideal incoming timeslots of all flows are perfectly interleaved, then they can all reserve timeslots by o=1 to get per-hop latency 10us, or all reserve timeslots by o=2 to get per-hop latency 20us, etc. However, due to the fixed length of OPL, after all timeslot resources are exhausted by specific o value, it means that there are no timeslot resources to be reserved by other o values. Another example is that the ideal incoming timeslots of all flows are the same, then some of them reserve timeslots by o=1, some reserve timeslots by o=2, and so on. In either case, the total service scale is OPL * C / burst_size, that is composed of sum(s_i), where s_i is the service scale for o=i. The table provides the total scale and the average scale corresponding to each o value.

Note that in the table each column only shows the data where all flows served based on all o values have the same TSpec (e.g, in the first colunm, TSpec per flow is burst size 1000 bits and arrival rate 1 Mbps), while in reality, flows served based on different o values generally have different TSpec. It is easy to add colunms to describe various combinations.

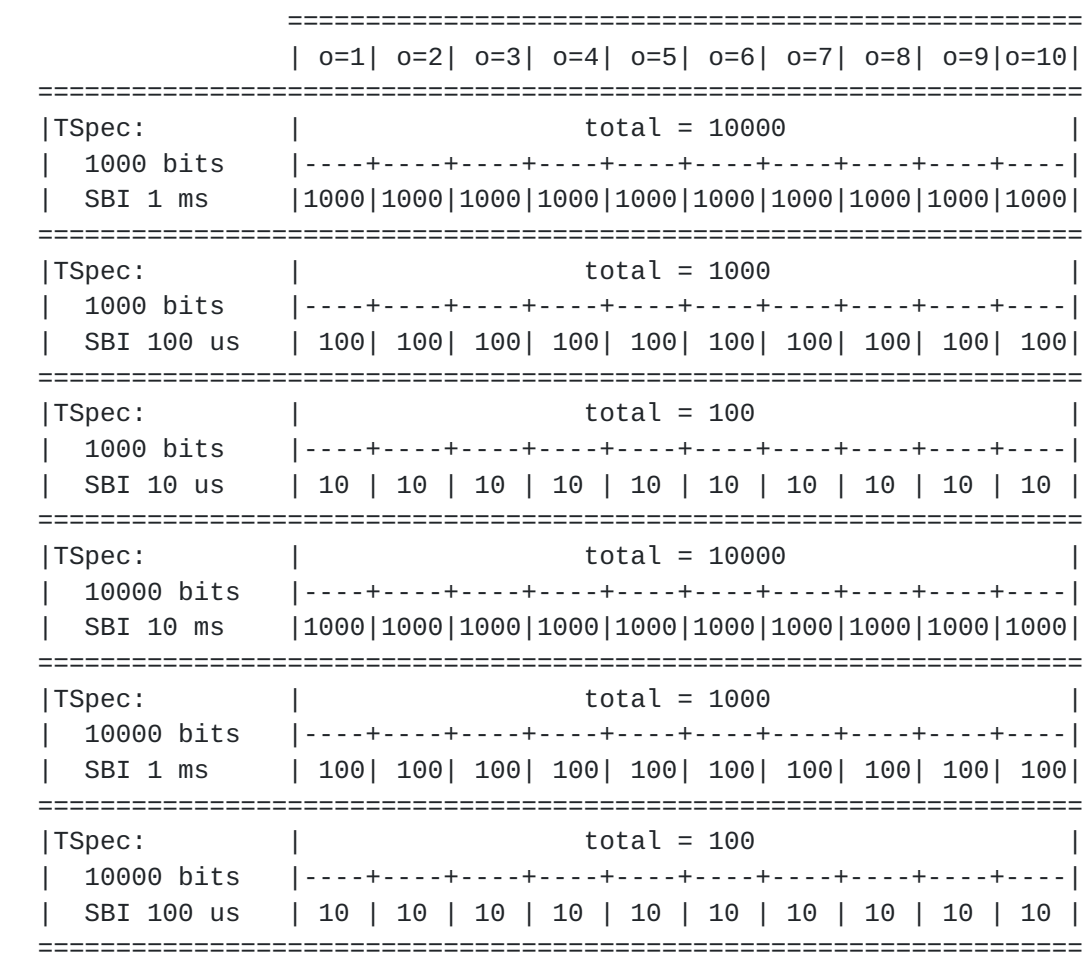

Figure 12: Timeslot Reservation and Service Scale Example

12. Taxonomy Considerations

[I-D.joung-detnet-taxonomy-dataplane] provides criteria for classifying data plane solutions. TQF is a periodic, frequency synchronous, class level, work-conserving/non-work-conserving configurable, in-time/on-time configurable, time based solution.

* Periodic: Periodicity of TQF contains two characteristics, the first is that there is a time period P (i.e., orchestration period) containing multiple time slots, and the second is that a flow is assigned repeatly to a particular set of time slots in the period.

* Frequency synchronous: TQF requires frequency synchronization (i.e., crystal frequency of the hardware) so that all nodes in the network have the same time lapse rate. TQF does not require different nodes to use the same timeslot length.

* Class level: DetNet Flows may be grouped by similar service requirements, i.e., timeslot id(s), on the network entrance. Packets will be provided TQF service based on timeslot id(s), without checking flow characteristic.

* Work-conserving/non-work-conserving configurable: The TQF scheduler configured with in-time scheduling mode is work-conserving (i.e., to send the packet as soon as possible before its outgoing timeslot), while the TQF scheduler configured with on-time scheduling mode is non work-conserving (i.e., to ensure that the packet can always be sent within its outgoing timeslot).

* In-time/on-time configurable: The TQF scheduler configured with in-time scheduling mode is in-time to get bounded end-to-end latency, while the TQF scheduler configured with on-time scheduling mode is on-time to get bounded end-to-end delay jitter.

* Time based: A DetNet flow is scheduled based on its expected outgoing timeslot(s). All DetNet flows are interleaved and arranged in different timeslots to obtain the maximum number of admission flows.

In addition, the per hop latency dominant factor of TQF is the offset between incoming timeslot and outgoing timeslot that is reserved to the flow.

13. IANA Considerations

TBD.

14. Security Considerations

Security considerations for DetNet are described in detail in [RFC9055]. General security considerations for the DetNet architecture are described in [RFC8655]. Considerations specific to the DetNet data plane are summarized in [RFC8938].

Adequate admission control policies should be configured in the edge of the DetNet domain to control access to specific timeslot resources. Access to classification and mapping tables must be controlled to prevent misbehaviors, e.g, an unauthorized entity may modify the table to map traffic to an unallowed timeslot resource, and competes and interferes with normal traffic.

15. Acknowledgements

TBD.

16. References

16.1. Normative 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 in Progress, 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-05, 5 January 2024, <https://datatracker.ietf.org/doc/html/draft-eckert-detnet-tcqf-05>.

[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-05, 20 November 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-detnet-scaling-requirements-05>.

[I-D.joung-detnet-taxonomy-dataplane]
Joung, J., Geng, X., Peng, S., and T. T. Eckert, "Dataplane Enhancement Taxonomy", Work in Progress, Internet-Draft, draft-joung-detnet-taxonomy-dataplane-01, 25 February 2024, <https://datatracker.ietf.org/doc/html/draft-joung-detnet-taxonomy-dataplane-01>.

[I-D.pb-6man-deterministic-crh]
Peng, S. and R. Bonica, "Deterministic Routing Header", Work in Progress, Internet-Draft, draft-pb-6man-deterministic-crh-00, 1 March 2024, <https://datatracker.ietf.org/api/v1/doc/document/draft-pb-6man-deterministic-crh/>.

[I-D.peng-detnet-policing-jitter-control]
Peng, S., Liu, P., and K. Basu, "Policing Caused Jitter Control Mechanism", Work in Progress, Internet-Draft, draft-peng-detnet-policing-jitter-control-00, 18 January 2024, <https://datatracker.ietf.org/doc/html/draft-peng-detnet-policing-jitter-control-00>.

[I-D.peng-lsr-deterministic-traffic-engineering]
Peng, S., "IGP Extensions for Deterministic Traffic Engineering", Work in Progress, Internet-Draft, draft-peng-lsr-deterministic-traffic-engineering-01, 4 July 2023, <https://datatracker.ietf.org/doc/html/draft-peng-lsr-deterministic-traffic-engineering-01>.

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

[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>.

[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>.

[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>.

16.2. Informative References

[ATM-LATENCY]
"Bounded Latency Scheduling Scheme for ATM Cells", 1999, <https://ieeexplore.ieee.org/document/780828/>.

[CQF] "Cyclic queueing and Forwarding", 2017, <https://ieeexplore.ieee.org/document/7961303>.

[ECQF] "Enhancements to Cyclic Queuing and Forwarding", 2023, <https://1.ieee802.org/tsn/802-1qdv/>.

[IEEE-1588] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", 2008, <https://standards.ieee.org/findstds/standard/1588-2008.html>.

[SP-LATENCY] "Guaranteed Latency with SP", 2020, <https://ieeexplore.ieee.org/document/9249224>.

[syncE] "Timing and synchronization aspects in packet networks", 2013, <https://www.itu.int/rec/T-REC-G.8261>.

[TAS] "Time-Aware Shaper", 2015, <https://standards.ieee.org/ieee/802.1Qbv/6068/>.

Authors' Addresses

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

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

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

Aihua Liu
ZTE
China
Email: liu.aihua@zte.com.cn

Dong Yang
Beijing Jiaotong University
China

Email: dyang@bjtu.edu.cn

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