Network
Internet-Draft
Intended status: Standards Track
Expires: September 2, 2022

Shaofu. Peng
Bin. Tan
ZTE Corporation
Peng. Liu
China Mobile

March 1, 2022

Deadline Based Deterministic Forwarding
draft-peng-detnet-deadline-based-forwarding-01

Abstract

This document describes a deterministic forwarding mechanism based on deadline. The mechanism enhances strict priority scheduling algorithm with dynamically adjusting the priority of the queue according to its deadline attribute.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any 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 September 2, 2022.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Requirements Language 3
2. Deadline Queue 3
3. Get Deadline Information of Packets 6
3.1. Get Planned Deadline 6
3.2. Get Existing Cumulative Planned Deadline 7
3.3. Get Existing Accumulated Actual Dwell Time 7
3.4. Get Existing Accumulated Deadline Deviation 8
4. Put Packets into the Deadline Queues 8
5. Traffic Regulation and Shaping 11
6. Compatibility Considerations 12
7. Benefits 13
8. IANA Considerations 13
9. Security Considerations 13
10. Acknowledgements 13
11. Normative References 13
Authors' Addresses 14

1. Introduction

[RFC8655] describes the architecture of deterministic network and defines the QoS goals of deterministic forwarding: Minimum and maximum end-to-end latency from source to destination, timely delivery, and bounded jitter (packet delay variation); packet loss ratio under various assumptions as to the operational states of the nodes and links; an upper bound on out-of-order packet delivery. In order to achieve these goals, deterministic networks use resource reservation, explicit routing, service protection and other means. Resource reservation refers to the occupation of resources by service traffic, exclusive or shared in a certain proportion, such as dedicated physical link, link bandwidth, queue resources, etc; Explicit routing means that the transmission path of traffic flow in the network needs to be selected in advance to ensure the stability of the route and does not change with the real-time change of network topology, and based on this, the upper bound of end-to-end delay and delay jitter can be accurately calculated; Service protection refers to sending multiple service flows along multiple disjoint paths at the same time to reduce the packet loss rate. In general, a deterministic path is a strictly explicit path calculated by a centralized controller, and resources are reserved on the nodes along the path to meet the SLA requirements of deterministic services.

[I-D.stein-srtsn] describes that the controller calculates the local deadline time of each node for the traffic to be transmitted in advance, which is a absolute system time, forms a stack of these local deadline time, and then carries them in the forwarded data packets. Each node forwards the packets according to its own local deadline. [I-D.stein-srtsn] suggests that FIFO queue can not be used to realize this function, because the packets stored in the queue are always first in first out, so a special data structure is recommoned. The packets in this data structure will be automatically sorted with the order from emergency to non emergency according to the deadline of the packets. However, it may be difficult to implement this structure in hardware, and especially for a large network it may be challenge to synchronize time.

Considering that the link transmission delay is generally a fixed value, and we focus on the dwell time of the packets inside the node, an alternate approach is to make the deadline eliminate the interference of link delay and avoids relying on time synchronization between nodes.

This document desrbies an alternate packets scheduling scheme that is used for wide area network. It suggests to only use a single deadline time to control the packets scheduling of all nodes along the path. The single deadline time is an offset time, which is based on the time when the packet enters the node and represents the maximum time allowed for the packet to stay inside the node. However, if each node has obvious differences in the capability of packets forwarding and scheduling, more offset-time may be needed.

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. Deadline Queue

For nodes in the network, some queues with deadline time (also termed as TTL) are introduced and maintained for each outgoing port. These queues are called deadline queue. Deadline queue has the following characteristics:

- The TTL of each deadline queue will decrease with the passage of time. When it decreases to 0, the scheduling priority of the queue will be set to the highest, and the scheduling opportunity will be obtained immediately (note that there may be interference

delay caused by a large packet being sent by a low priority queue). It will prohibit receiving new packets, in which the buffered packets will be sent to the outgoing port immediately, and the maximum duration allowed to send packets is the authorization time. In principle, all packets buffered in the queue shall be sent within this authorization time. If the queue is sent and the authorization time is still free, other queues with lower priority can be scheduled during this authorization time.

- The scheduling engine can initiate a cycle timer to decrement the TTL of all deadline queues, that is, whenever the timer expires, the deadline values of all queues will be subtracted from the timer interval. Note that the time interval of the timer must be greater than or equal to the authorization time of the deadline queue. For simplicity, they are same.

- For a deadline queue whose TTL has been reduced to 0, after a new round of timer timeout, the TTL will return to the maximum initial value, allow receiving new packets, and continue to enter the next round of operation that decreases with the passage of time.

- For a deadline queue whose TTL is not reduced to 0, it can receive packets. In detailed, when a node receives a packet to be forwarded from a specific outgoing port, it first obtains the expected deadline of the packet, and then put the packet to the deadline queue with the relevant TTL value of the outgoing port for transmission.

- For a deadline queue whose TTL is not reduced to 0, its scheduling priority cannot be set to the highest value. A local policy may be used to control the transmission of buffered packets. There are two options: the first option, allowing to be involved in scheduling, also termed as in-time policy; the second option, not allowing, also termed as on-time policy. In-time policy is applicable to the service requirements of low delay, and on-time policy is applicable to low delay jitter. When instantiating deadline scheduling algorithm, one implementation can support only one option or both.

- At the beginning, all deadline queues have different TTL values, i.e., staggered from each other, so that the TTL of only one deadline queue will decrease to 0 at any time.

The above authorization time, timer interval and maximum initial TTL value shall be specified according to the actual capacity of the node. In fact, each node in the network can independently use different timer interval for different outgoing ports. The general

principle is that if an outgoing port has a large bandwidth (such as 100G bps), the timer interval (and the authorization time of the deadline queue) can be small (such as 1us), because the link with large bandwidth can send the required bits amount even within a small time interval; If an outgoing port has a small bandwidth (e.g. 1G bps), the timer interval (and the authorization time of the deadline queue) should be larger (e.g. 10us), because the link with small bandwidth needs to send the required bit amount within a larger time interval.

A specific example of the deadline queue is depicted in Figure 1.

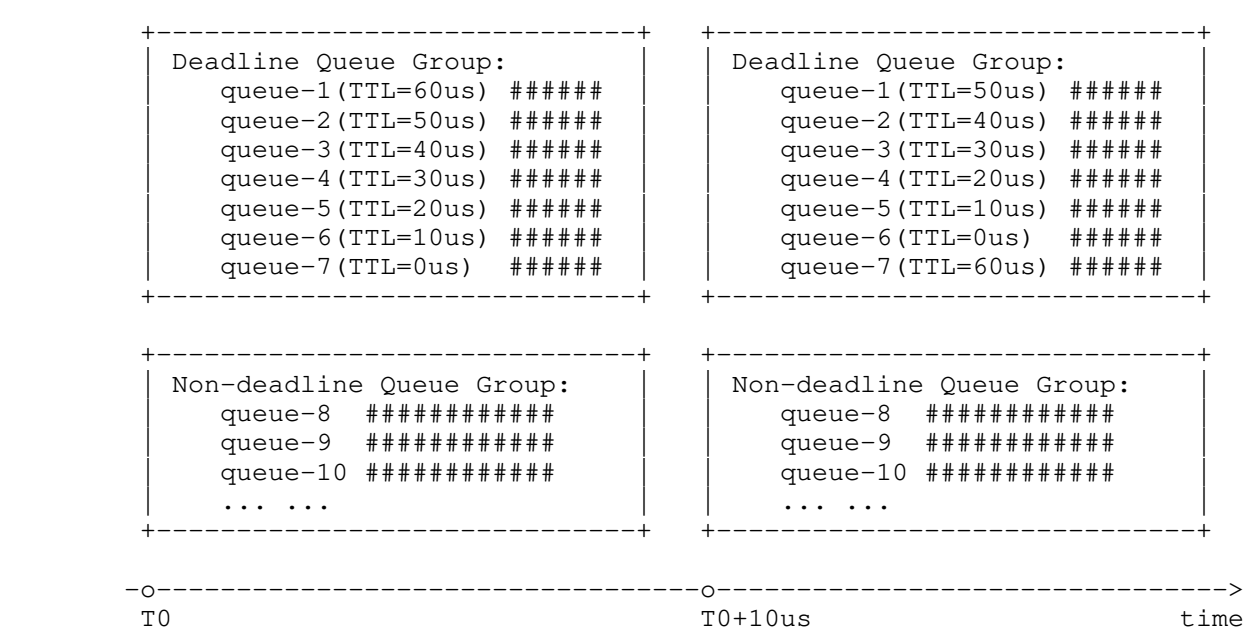

Figure 1: Example of Deadline Queue for outgoing Port

In this example, the timer interval for deadline queue group is configured to 10us. Queue-1 ~ queue-7 are deadline queues, and other queues are traditional non-deadline queues. Each deadline queue has its TTL attribute. The maximum initial TTL is 60us. At the initial time (T0), the TTL of all deadline queues are staggered from each other. For example, the TTL of queue-1 is 60us, the TTL of queue-2 is 50uS, the TTL of queue-3 is 40us, and so on. At this time, only the TTL of queue-7 is 0, which has the highest scheduling priority.

Suppose the scheduling engine initiates a cycle timer with a time interval of 10us. After each timer timeout, the timer interval will be subtracted from the TTL of all deadline queues. As shown in the

figure, at T0 + 10us, the timer timeout, the TTL of queue-1 becomes 50uS, the TTL of queue-2 becomes 40us, the TTL of queue-3 becomes 30us, etc. At this time, the TTL of queue-7 returns to the maximum initial TTL of 60us and is no longer set to the highest scheduling priority; The TTL of queue-6 becomes 0, which has the highest scheduling priority.

For simplicity, set the authorization time of the deadline queue to be consistent with the time interval of the cycle timer, which is also 10us. When the TTL of a deadline queue becomes 0, it has a time limit of 10us to send packets in the queue. During this period, it will be prohibited to receive new packets (in fact, there can be no new packets with a deadline of 0). After the 10us time elapses, the cycle timer will timeout again, The TTL of another deadline queue will change to 0. It is also feasible to set the authorization time to be less than the cycle timer interval.

If the deadline queue with the highest priority is still free after sending packets within the authorized time, the scheduling engine will visit other queues with the second highest priority during the rest of the authorized time.

Note that for each deadline queue with specific TTL, if both in-time and on-time policies are supported, it may include two sub-queues, one used to buffer in-time packets and the other used to buffer on-time packets.

3. Get Deadline Information of Packets

3.1. Get Planned Deadline

The planned deadline of the packet is an offset time, which is based on the time when the packet enters the node and represents the maximum time allowed for the packet to stay inside the node. There are many ways to obtain the planned deadline of the packet.

- Carried in the packet. The ingress PE node, when encapsulating the deterministic service flow, can explicitly insert the planned deadline into the packet according to SLA. The intermediate node, after receiving the packet, can directly obtain the planned deadline from the packet. Generally, only a single planned deadline needs to be carried in the packet, which is applicable to all nodes along the path; Or insert a stack composed of multiple deadlines, one for each node. [I-D.peng-6man-deadline-option] defined a method to carry planned deadline in the IPv6 packets .

- Included in the FIB entry. Each node in the network can maintain the deterministic FIB entry. After the packet hits the

deterministic FIB entry, the planned deadline is obtained from the forwarding information contained in the FIB entry.

- Included in the policy entry. Configure local policies on each node in the network, and then set the corresponding planned deadline according to the matched specific characteristics of the packet, such as 5-tuple.

For a deterministic delay path based on deadline queue scheduling, the path it passes through has deterministic end-to-end delay requirements. It includes two parts, one is the cumulative node delay and the other is the cumulative link transmission delay. The end-to-end delay requirement is subtracted from the cumulative link transmission delay to obtain the cumulative node delay. A simple method is that the accumulated node delay is shared equally by each intermediate node along the path to obtain the planning deadline of each node.

3.2. Get Existing Cumulative Planned Deadline

The existing cumulative planned deadline of the packet refers to the sum of the planned deadline of all upstream nodes before the packet is transmitted to this node. This information needs to be carried in the packet. Every time the packet passes through a node, the node accumulates its corresponding planned deadline to the existing cumulative planned deadline field in the packet. [I-D.peng-6man-deadline-option] defined a method to carry existing cumulative planned deadline in the IPv6 packets.

The setting of "existing cumulative planned deadline" in the packet needs to be friendly to the chip for reading and writing. For example, it should be designed as a fixed position in the packet. The chip may support flexible configuration for that position.

3.3. Get Existing Accumulated Actual Dwell Time

The existing cumulative actual dwell time of the packet, refers to the sum of the actual dwell time of all upstream nodes before the packet is transmitted to this node. This information needs to be carried in the packet. Every time the packet passes through a node, the node accumulates its corresponding actual dwell time to the existing cumulative actual dwell time field in the packet. [I-D.peng-6man-deadline-option] defined a method to carry existing cumulative actual dwell time in the IPv6 packets.

The setting of "existing cumulative actual dwell time" in the packet needs to be friendly to the chip for reading and writing. For

example, it should be designed as a fixed position in the packet. The chip may support flexible configuration for that position.

Although other methods can also be, for example, carrying the absolute system time of receiving and sending in the packet to compute the actual dwell time indirectly, that has a low encapsulation efficiency and require strict time synchronization between nodes.

A possible method to get the actual dwell time in the node is that, the receiving and sending time of the packet can be recorded in the auxiliary data structure (note that is not packet itself) of the packet, and the actual dwell time of the packet in the node can be calculated according to these two times.

3.4. Get Existing Accumulated Deadline Deviation

The existing accumulated deadline deviation equals existing cumulative planned deadline minus existing cumulative actual dwell time. This value can be positive or negative.

If the existing cumulative planned deadline and the existing cumulative actual dwell time are carried in the packet, it is not necessary to carry the existing accumulated deadline deviation. Otherwise, it is necessary. The advantage of the former is that it can be applied to more scenarios.

4. Put Packets into the Deadline Queues

The lifetime of the packet inside the node mainly includes two parts: the first part is to lookup the forwarding table when the packet is received from the incoming port (or generated by the control plane) and deliver the packet to the line card where the outgoing port is located; the second part is to store the packet in the queue of the outgoing port for transmission. These two parts contribute to the actual dwell time of the packet in the node. The former can be called forwarding delay and the latter can be called queuing delay. The forwarding delay is related to the chip implementation and is generally constant; The queuing delay is unstable.

When a node receives a packet from an upstream node, it can first get the existing accumulated deadline deviation, and then add it to the planned deadline of the packet at this node to obtain the deadline adjustment value, and then on the basis of the deadline adjustment value, deducting the forwarding delay of the packet in the node, the allowable queuing delay value is obtained, and then the packet will be put to the deadline queue with TTL as the above allowable queuing delay value for sending. If the calculated allowable queuing delay

is not exactly equal to the TTL of any queue, the packet selects the queue with the closest TTL to enter.

Under normal circumstances, if each hop strictly controls the scheduling of the packet according to its planned deadline, the actual dwell time of the packet will be very close to the planned deadline, and the absolute value of the existing accumulated deadline deviation will be very small.

More generally, assume that the local node in a deterministic path is i, all upstream nodes are from 1 to i-1, and downstream nodes are i + 1, the planned deadline is D, the actual dwell time is R, the deadline adjustment value is M, the forwarding delay inside the node is F, the existing accumulated deadline deviation is E, and the allowable queuing delay is Q, then the allowable queuing delay (Q) of the packet on this node i is calculated as follows:

$$E(i-1) = D(1) + D(2) + \ldots + D(i-1) - R(1) - R(2) - \ldots - R(i-1)$$

$$M(i) = D(i) + E(i-1)$$

$$Q(i) = M(i) - F(i)$$

Consider some extreme cases. For example, many upstream nodes adopt the in-time policy to send packets quickly. Packets almostly need not queue in these nodes, but only depend on the forwarding delay. Then the existing accumulated deadline deviation (E) may be a very large positive value, resulting in a large allowable queuing delay (Q). If this value exceeds the maximum initial TTL of the deadline queue maintained by the node, the allowable queuing delay (Q) should be modified to the maximum initial TTL.

For another example, if some upstream nodes are abnormal and have a very large actual dwell time (R), the existing accumulated deadline deviation (E) may be a negative number, resulting in the allowable queuing delay (Q) may be less than or equal to 0, which is smaller than the cycle timer interval of the deadline queue maintained in the node, then the allowable queuing delay (Q) should be modified to the cycle timer interval value.

Figure 2 depicts an example of packets buffered to the deadline queue.

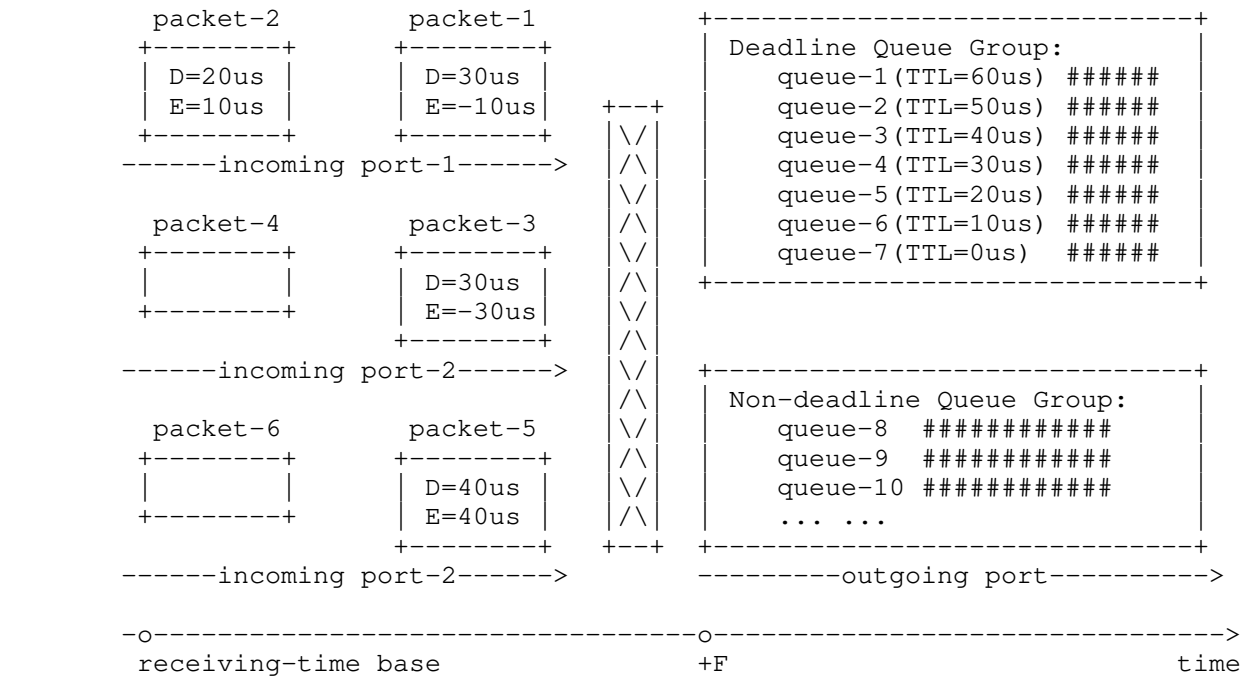

Figure 2: Time Sensitive Packets Buffered to Deadline Queue

As shown in Figure 2, the node successively receives six packets from three incoming ports, among which packet 1, 2, 3 and 5 have corresponding deadline information, while packet 4 and 6 are traditional packets. These packets need to be forwarded to the same outgoing port according to the forwarding table entries. It is assumed that they arrive at the line card where the outgoing port is located at almost the same time after the forwarding delay in the node (F = 10us). At this time, the queue status of the outgoing port is shown in the figure. Then:

- The allowable queuing delay (Q) of packet 1 in the node is 30 - 10 -10 = 10us, and it will be put into the deadline queue-6 (its TTL is 10us).

- The allowable queuing delay (Q) of packet 2 in the node is 20 + 10 -10 = 20us, and it will be put into the deadline queue-5 (its TTL is 20us).

- The allowable queuing delay (Q) of packet 3 in the node is 30 - 30 -10 = -10us, and it will be modified to the minimum positive value 10 us then put into the deadline queue-6 (its TTL is 10us). Note

that the minimum positive value is a timer interval that is a local parameter per port based on port's bandwidth.

- The allowable queuing delay (Q) of packet 5 in the node is 40 + 40 -10 = 70us, and it will be modified to the maximum positive value 60 us then put into the deadline queue-1 (its TTL is 60us). Note that the maximum positive value is an empirical value that can be configured according to the maximum delay requirements of deterministic services.

- Packets 4 and 6 will be put into the non-deadline queue in the traditional way.

5. Traffic Regulation and Shaping

On the ingress PE node, traffic regulation is performed on UNI port, so that the service traffic does not exceed its reserved bandwidth. Suppose there are N sources, and the packets they send carry the same deadline. These packets may arrive at an intermediate node at the same time and put into the same deadline queue. If the reserved bandwidth of deadline queue at N sources is M0, and the reserved bandwidth of deadline queue at intermediate nodes is Mx, then it needs to meet: N * M0 < = Mx. This means that a larger bandwidth is required on the intermediate node to send more bits in the same time duration, i.e., larger buffer size. Especially, packets with different deadlines sent by a single ingress PE node at different times, may be put into the same deadline queue by an intermediate node and sent within the same authorization time. In order to mitigate this impact, it is not recommended to apply the on-time policy to packets with large deadline value.

On the ingress PE node, traffic shaping is performed on NNI port. Multiple continuous packets of the specific service flow are stored in the deadline queue with corresponding remaining time according to the planned deadline of the service flow. Note that these packets are not stored in the same queue over time. The amount of bits that can be stored in one queue is equal to the reserved bandwidth * authorization time, however, at least one whole packet shall be loaded. For example, if the allowable queuing delay is 20us, then within the current timer interval, the first sequence of the packets will be put into the current deadline queue with TTL = 20us until the reserved bandwidth limit is reached; Then, within the next timer interval, the next sequence of packets will be put into the current TTL = 20us queue until the reserved bandwidth limit is reached; and so on, until the total service bits are loaded.

Figure 3 depicts an example of deadline based traffic shaping on the ingress PE node. It is assumed that the packets loaded in each timer interval do not exceed the reserved bandwidth of the service.

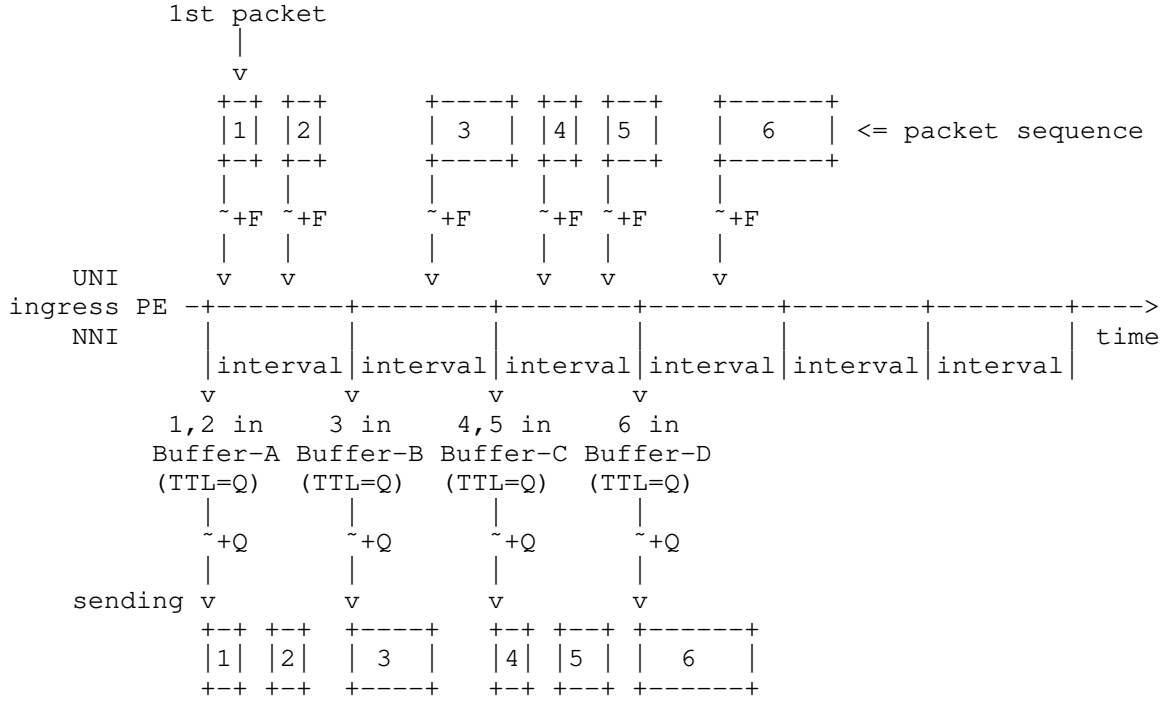

Figure 3: Deadline Based Traffic Shaping

6. Compatibility Considerations

For a particular path, if only some nodes in the path upgrade support the deadline mechanism defined in this document, the end-to-end deterministic delay/jitter target will only be partially achieved. Those legacy devices may adopt the existing priority based scheduling mechanism, and ignore the possible deadline information in the packet, thus the delay intra node produced by them cannot be perceived by the adjacent upgraded node. The more upgraded nodes included in the path, the closer to the delay/jitter target.

However, only a few key nodes are upgraded to support deadline mechanism, which is low-cost, but can meet a service with relatively loose time requirements.

7. Benefits

The mechanism described in this document has the following benefits:

- Time synchronization is not required between network nodes. Each node can flexibly set the authorization time length of the deadline queue according to its own outgoing port bandwidth.

- Packet multiplexing based, it is an enhancement of PQ scheduling algorithm, friendly to the upgrade of packet switching network. All nodes in the network can independently use cycle timers with different timeout intervals to traverse the deadline queues.

- The packet can control its expected dwell time in the node. A single set of deadline queues supports multiple levels of dwell time.

- For in-time policy, the end-to-end delay is H*(F~D), jitter is H*Q; For on-time policy, the end-to-end delay is H*D, jitter is a just single authorization time.

8. IANA Considerations

There is no IANA requestion for this document.

9. Security Considerations

TBD

10. Acknowledgements

TBD

11. Normative References

[I-D.peng-6man-deadline-option]
Peng, S. and B. Tan, "Deadline Option", draft-peng-6man-deadline-option-00 (work in progress), January 2022.

[I-D.stein-srtsn]
Stein, Y. (., "Segment Routed Time Sensitive Networking", draft-stein-srtsn-01 (work in progress), August 2021.

[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>.

Authors' Addresses

Shaofu Peng
ZTE Corporation
China

Email: peng.shaofu@zte.com.cn

Bin Tan
ZTE Corporation
China

Email: tan.bin@zte.com.cn

Peng Liu
China Mobile
China

Email: liupengyjy@chinamobile.com