LSR
Internet-Draft
Intended status: Standards Track
Expires: 26 February 2023

S. Peng
ZTE
T. Li
Juniper Networks

25 August 2022

IGP Flexible Algorithm with Deterministic Routing
draft-peng-lsr-flex-algo-deterministic-routing-03

Abstract

IGP Flexible Algorithm proposes a solution that allows IGPs to compute constraint based paths over the network and specifies a way of using Segment Routing (SR) Prefix-SIDs, SRv6 locators, or pure IP prefixes to steer packets along the constraint-based paths. This document describes how to compute deterministic delay paths within a Flex-Algorithm topology.

Table of Contents

1. Introduction

IGP Flexible Algorithm [I-D.ietf-lsr-flex-algo] (Flex-Algorithm) proposes a solution that allows IGPs to compute constraint based paths over the network and specifies a way of using Segment Routing [RFC8402] Prefix-SIDs, SRv6 locators, or pure IP prefixes [I-D.ietf-lsr-ip-flexalgo] to steer packets along the constraint-based paths. It specifies a set of extensions to IS-IS, OSPFv2 and OSPFv3 that enable a router to send TLVs that identify (a) calculation-type, (b) specify a metric-type, and (c)describe a set of constraints on the topology, that are to be used to compute the best paths along the constrained topology. A given combination of calculation-type, metric-type, and constraints is known as a FAD (Flexible Algorithm Definition).

[RFC8655] describes the architecture of a 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)

* A bounded packet loss ratio under various assumptions about 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, and service protection, as well as other means. A deterministic path is typically (but not necessarily) an explicit route so that it does not suffer temporary interruptions caused by the convergence of routing or bridging protocols.

Flexible-Algorithm has the characteristic mentioned in [RFC8655] that it operates under a single administrative control or within a closed group of administrative control. Flexible-Algorithm also supports Min Unidirectional Link Delay (defined in [RFC8570]) metric type to compute shortest paths with minimum delay, however, the cumulative delay is essentially the accumulation of propagation delay of all links, excluding node delay. In order to compensate for this gap, it is necessary to enhance Flexible-Algorithm to compute the path with deterministic delay, i.e., including deterministic node delay as well.

This document describes how to compute distributed shortest paths with a deterministic delay metric within a Flexible-Algorithm topology, as the basis of the whole distributed deterministic scheme. It should be noted that relying on this enhancement alone does not guarantee complete determinacy, it needs to be used in conjunction with other tools, such as creating additional redundant deterministic delay paths with consistent delay metric for PREOF (Packet Replication, Elimination, and Ordering Functions), smoothing the delay jitter during route convergence, providing a deterministic forwarding mechanism, admission control, etc.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Determinisitc Links

When a packet is forwarded to a link, the delay produced includes two parts: the first part is the dwell delay of the packet in the node, and the second part is the propagation delay of the packet on the link. In packet switching networks, a priority based queuing scheme is sometimes used. It may give better average latency, but may have unacceptable worst case latency. [SP-LATENCY] analyzes the guaranteed latency with the traditional strict priority scheme, and shows that low bounded latency is achievable when high priority traffic is constrained in low utilization, but deteriorates quickly with increasing amounts of high priority traffic. DiffServ [RFC2475] with strict priority has been deployed in the network and existing non-deterministic service flows may elect the highest priority, so it is difficult to support deterministic services based on DiffServ without any modification.

We call a link with a queuing mechanism that does not guarantee a bounded delay a non-determinisitc link and a link with a queuing mechanism that can provide deterministic node delay is called a deterministic link. To achieve a deterministic network, other new scheduling mechanisms need to be introduced, and their scheduling priority must be higher than that of the traditional strict priority queue. The typical queuing mechanisms are as follows:

* IEEE 802.1 WG has specified IEEE802.1Qav [CBS] which uses a credit-based shaper mechanism to assign packets to different queues based on a credit value that is proportional to the flow's reserved bandwidth. The credit values of different transmission

queues will automatically change with the packet transmission process, which will ensure that packets with lower priority will also get transmitted. The CBS shaper mechanism is similar to Weighted Fair Queuing (WFQ) scheme, and they both control the scheduling of packets based on the reserved bandwidth. The worst-case delay calculation of class A CBS traffic is relatively simple, but other classes are complex. For class A traffic, the queuing delay equals to the maximum size of the interference frame (such as 2000 octets) divided by the port bandwidth.

* IEEE 802.1 WG has specified IEEE802.1Qch [CQF] which uses a cyclic queuing and forwarding (CQF) mechanism and relies on time synchronization. Under CQF, the maximum delay experienced by a given packet is (H+1)*D, the minimum delay experienced by a given packet is (H-1)*D, and the delay jitter is 2*D, where H is the number of hops and D is cycle duration. Other variants based on CQF can avoid relying on time synchronization, but require the same cycle duration for all nodes. Basically, if a packet received in the current sending window (i.e., cycle) can always be sent in the next sending window, then the deterministic node delay, on average, is one cycle duration, or several cycle durations if the intra-node forwarding delay (from incoming port to outgoing port) can't be ignored.

* [I-D.peng-detnet-deadline-based-forwarding] introduced a deadline based forwarding mechanism that allows a packet to control its expected dwell time in the node according to a planned deadline. There are two policies for scheduling packets in deadline queuing. For the in-time policy, the end-to-end delay is H*(F~D) and the jitter is H*Q, where H is the number of hops, F is the intra-node forwarding delay, D is the planned deadline, and Q is the scheduling delay. For the on-time policy, the end-to-end delay is H*D, and the jitter is 0 (however there may be a one authorization time delay due to the granularity of queue scheduling). That is, a packet received can be sent within F~D or D respectively for these two policies.

  We define authorization time to be the time period that a given deadline queue is allowed to transmit the stored packets. Deadline queue participates in priority based scheduling. A deadline queue with the highest priority will maintain its highest priority for the predefined authorization time. During this period, if it contains packets, it may obtain one or more authorization size (e.g, an authorization size of 2000 bytes which is generally a chip configurable fixed value) from the scheduling decision module, to clear the queue. When it becomes empty, it will withdraw the request and the scheduling opportunity will be given to other queues with lower priority. For a deadline queue

with in-time mode, it always sends a request to the scheduling decision module as long as it contains packets, regardless of its priority. Instead, for a deadline queue with on-time mode, if it contains packets, the request is sent only when its priority becomes the highest.

This document discusses deterministic links based on CQF or the deadline algorithm. Other algorithms will be described in the future. Note that a deadline or CQF queue has the highest resource access priority. In extreme cases, this kind of queue can access the whole port bandwidth, but the queue does no really reserve bandwidth and is independent of any specific service flow. However, the operator can create and maintain the reserved bandwidth of the service flow from the perspective of the service, and take admission control on the ingress node.

3.1. Deterministic Link Bounds with CQF

A node may be configured with the parameters for CQF based packet scheduling for its local links, including enabling CQF scheduling and cycle durations, which in turn determine the node delay and jitter attributes for each link. The meanings of these parameters are:

* CQF scheduling enabled: if the CQF scheduling algorithm is enabled for a link, then the packets sent to that link will be scheduled by the CQF scheduling algorithm.

* Cycle duration: the cycle duration, also called cycle_size. One or more durations of different values can be configured for a link, such as 10us, 20us, 30us, and so on.

* Node delay/jitter:

  - According to classical CQF, for a given cycle_size, it can be deduced that the minimum delay in the node for a packet is 0, the maximum delay in the node is 2*cycle_size, the average delay in the node is one cycle_size, and the delay jitter in the node is 2*cycle_size. The detailed reasons for this are as follows: if a node receives a packet at the tail end of cycle i and sends that packet at the head end of cycle i+1, the resulting node delay, i.e., the minimum node delay, is 0; if a node receives a packet at the head end of cycle i and sends that packet at the tail end of cycle i+1, the resulting node delay, i.e., the maximum node delay, is 2*cycle_ size; the average node delay is one cycle_size, and the node delay jitter is 2*cycle_size. Each cycle_size corresponds to a different set of delay/jitter attributes.

- However, for some variants based on CQF, if the intra-node forwarding delay can't be ignored, e.g, wasting 2 cycle duration, then the minimum node delay, the maximum node delay, and the average node delay need to add 2 cycle_size respectively, but the node delay jitter is still 2*cycle_size.

3.2. Deterministic Link Bounds with Deadlines

A node may be configured for deadline based packet scheduling for its local links, including enabling deadline scheduling, one or more deadline scheduling delays, and the scheduling policy supported for each deadline scheduling delay. The deadline scheduling delay will determine the node delay and jitter attributes of the link. The meanings of these parameters are:

* Deadline scheduling enabled: if the deadline scheduling algorithm is enabled for a link, then a packet forwarded to the link will be scheduled by the deadline based packet scheduling algorithm. The dwell time of the packet in the node will not exceed the maximum allowable dwell time D, where, D = intra-node forwarding delay (F) + specific deadline scheduling delay (Q).

* Deadline scheduling delays: a set composed of one or more deadline scheduling delays <Q1, Q2, ..., Qn>, assuming that Q1 is the minimum and Qn is the maximum in the set. Generally, the difference between two adjacent elements in the set is fixed, for example, a fixed interval (I).

* Scheduling policy: for each scheduling delay Q, there are two possible scheduling policies: the in-time policy and the on-time policy. In the case of the in-time policy, the packet may be sent to the outgoing port even when it's scheduling delay has not reached Q. For the on-time policy, the packet is sent to the outgoing port only when the scheduling delay of the packet is equal to Q. Therefore, for in-time policy, the actual dwell time of the packet in the node is within the range [F, F+Q], i.e., the minimum node delay is F, the maximum node delay is F+Q, and the node delay jitter is Q; For on-time policy, the actual dwell time of the packet in the node is equal to F+Q, i.e., the minimum node delay is F+Q, the maximum node delay is also F+Q, and the node delay jitter is 0 (however, there may be one authorization time delay due to the granularity of queue scheduling).

4. Deterministic Delay Metric Extension to IS-IS

This document defines a new sub-TLV, the Deterministic Delay Metric, in the "IS-IS Sub-TLVs for TLVs Advertising Neighbor Information" registry, to distribute deterministic delay information. The deterministic delay advertised by this sub-TLV MUST be the delay from the local neighbor to the remote neighbor (i.e., the forward-path latency). The format of this sub-TLV is:

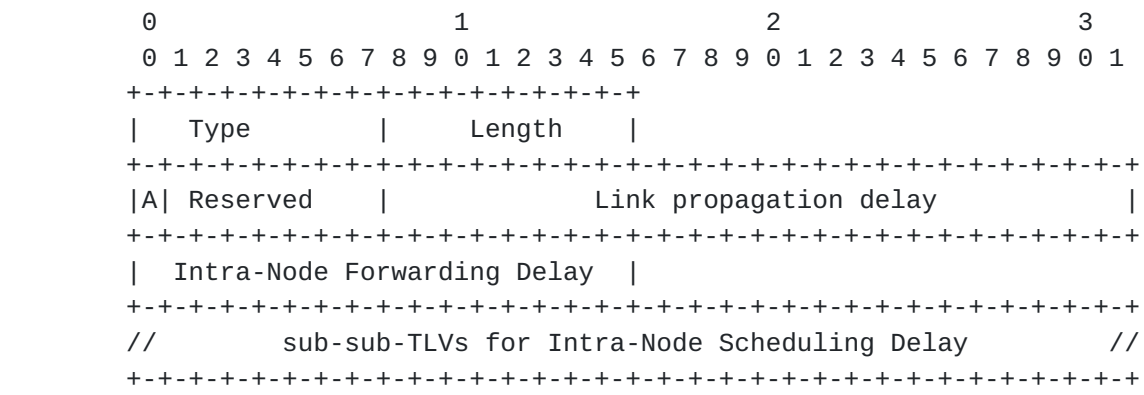

Figure 1

where:

Type: TBA1

Length: Variable

A bit: This field represents the Anomalous (A) bit. The A bit is set when one or more measured values of link propagation delay exceed a configured maximum threshold. The A bit is cleared when the measured value falls below its configured reuse threshold. If the A bit is cleared, the sub-TLV represents steady-state link propagation delay.

Reserved: This field is reserved for future use. It MUST be set to 0 when sent and MUST be ignored when received.

Link propagation delay: This 24-bit field carries the average measured link propagation delay value (in microseconds) over a configurable interval, encoded as an integer value. Implementations MAY also permit the configuration of an offset value (in microseconds) to be added to the measured delay value, to facilitate the communication of operator-specific delay constraints. When the delay value is set to the maximum value 16,777,215 (16.777215 seconds), then the delay is at least that value and may be larger. Note that [RFC8570] also specified the

Unidirectional Link Delay Sub-TLV and the Min/Max Unidirectional Link Delay Sub-TLV to advertise the average and min/max link delay respectively. For simplicity, the Deterministic Delay Metric only contains the average link delay. Although the exising TE or Flexalgo path may be calculated with delay metric, which is exactly Min Unidirectional Link Delay metric, to reduce information flooding, for these applications, only Min/Max Unidirectional Link Delay Sub-TLV, but not Unidirectional Link Delay Sub-TLV, need to be advertised by IGP. Then, as deterministic routing introduced, IGP should also advertise Deterministic Delay Sub-TLV (which contains average link delay), and Unidirectional Link Delay Sub-TLV still doesn't need to be advertised.

Intra-Node Forwarding Delay: This 16-bit field carries the intra-node forwarding delay value (in microseconds). It represents the latency of a packet from reception on the incoming port (or generated from control plane) to queuing on the outgoing port. If the forwarding delay can be ignored, it is set to 0.

Note: the forwarding delay for a node may be identical for all links on that node. In a future version of this document, we will consider advertising a single forwarding delay for the node.

sub-sub-TLVs for Intra-Node Scheduling Delay: Optional sub-sub-TLVs are included to indicate the scheduling delays that are related to the scheduling algorithm such as CQF, deadline, etc. If this field is absent, the scheduling delay is unknown. Typically, a link may enable a single scheduling algorithm to get deterministic scheduling delay, so that a single sub-sub-TLV is included. However, it is possible for a link to enable multiple different scheduling algorithms, as long as these algorithms can coordinate the forwarding resources, in this case, multiple sub-sub-TLVs are included. Supported Sub-sub-TLVs are specified in the following sub-sections.

4.1. CQF Intra-Node Scheduling Delay Sub-Sub-TLV

The CQF Intra-Node Scheduling Delay Sub-Sub-TLV is an optional Sub-Sub-TLV of the Deterministic Delay Metric Sub-TLV. Only one CQF Intra-Node Scheduling Delay Sub-Sub-TLV MAY be included.

The CQF Intra-Node Scheduling Delay Sub-Sub-TLV has the format:

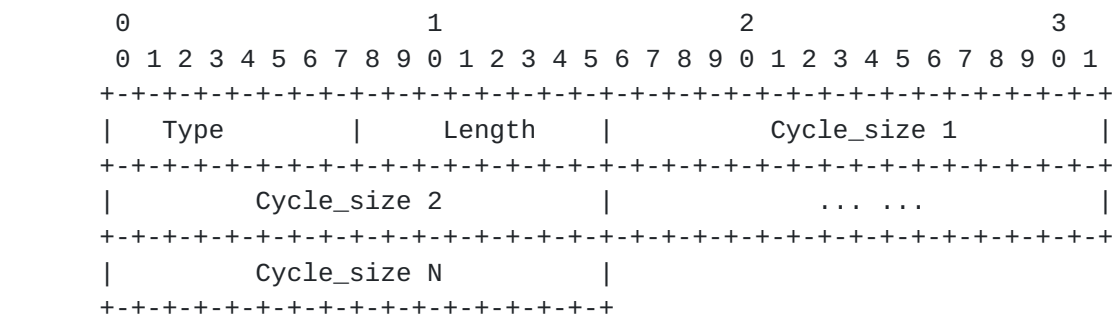

Figure 2

where:

Type: 1

Length: 2*N, depending on the number of the cycle_sizes included

Cycle_size: The length of cycle duration, in units of microseconds. A link can support multiple cycle durations, for example, 10us, 20us, 30us, etc, each for a specific service requirement.

Only those links that enable the CQF scheduling algorithm need to advertise the Intra-Node CQF Scheduling Delay Sub-Sub-TLV.

Note that the advertised cycle_size must be consistent with the CQF queue scheduling mechanism actually instantiated by the link in the forwarding plane. If the forwarding plane does not instantiate a CQF queue scheduling supporting a certain cycle_size, that is advertised in the CQF Intra-Node Scheduling Delay Sub-Sub-TLV, the subsequent route computation may get the wrong results.

For a given cycle_size, other nodes can deduce the corresponding node delay and jitter attributes, so these attributes can no longer be explicitly included in the CQF Intra-Node Scheduling Delay Sub-Sub-TLV.

As mentioned earlier, if the intra-node forwarding delay, F, is not 0, then the minimum node delay, the maximum node delay, and the average node delay need to take F into account respectively. F is replaced by ((F/cycle_size)+1)*cycle_size for computation. That is:

* If F is 0 for a given cycle_size, the minimum node delay is 0, the maximum node delay is 2*cycle_size, the average node delay is cycle_size, and the node delay jitter is 2*cycle_size.

* If F is not 0, for a given cycle_size, the minimum node delay is ((F/cycle_size)+1)*cycle_size, the maximum node delay is ((F/cycle_size)+3)*cycle_size, the average node delay is ((F/cycle_size)+2)*cycle_size, and the node delay jitter is 2*cycle_size.

4.2. Deadline Intra-Node Scheduling Delay Sub-Sub-TLV

The Deadline Intra-Node Scheduling Delay Sub-Sub-TLV is an optional Sub-Sub-TLV of the Deterministic Delay Metric Sub-TLV. Only one Deadline Intra-Node Scheduling Delay Sub-Sub-TLV MAY be included.

The format of the Deadline Intra-Node Scheduling Delay Sub-Sub-TLV is:

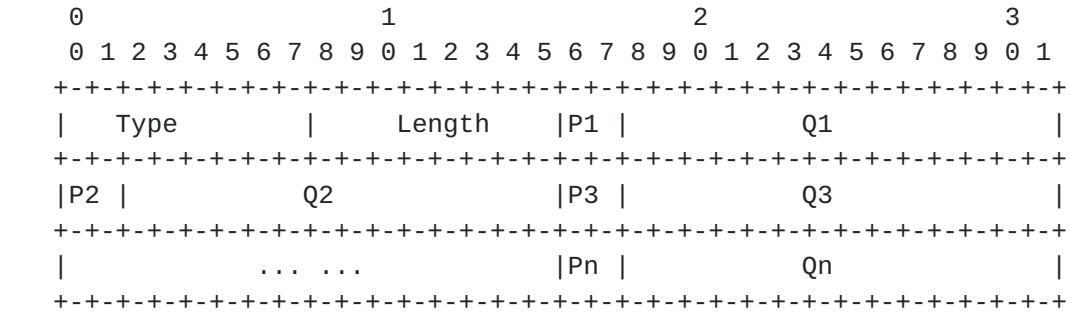

Figure 3

where:

Type: 2

Length: 2*n, for n deadline scheduling delay pairs, (Pn, Qn).

Pn: Two bits indicating the scheduling policy associated with the nth scheduling delay. The value of Pn indicates:

0 reserved

1 the in-time policy

2 the on-time policy

3 both the in-time policy and the on-time policy

Qn: Fourteen bits indicating the nth scheduling delay, in units of microseconds.

As mentioned earlier, given a scheduling delay Qn and its scheduling policy Pn, combined with the intra-node forwarding delay F, the corresponding delay and jitter attributes for the node can be calculated. Therefore, the delay and jitter associated with the queue are not explicitly included in the Deadline Intra-Node Scheduling Delay Sub-Sub-TLV.

4.2.1. Simplified Deadline Intra-Node Scheduling Delay TLV

Frequently in deadline scheduling, the queuing delays are all separated by a fixed interval and have a common policy. In this case, it is more efficient to encode the set of queuing delays as a policy, starting value, ending value, and interval. The format of the Simplified Deadline Intra-Node Scheduling Delay Sub-Sub-TLV is:

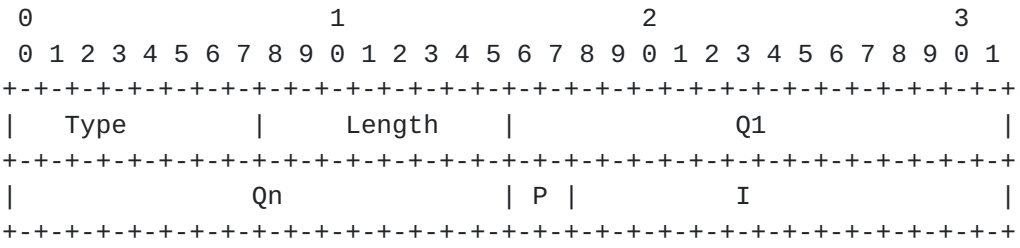

Figure 4

where:

Type: TBD

Length: 6

Q1: the minimum scheduling delay, in units of microseconds.

Qn: the maximum scheduling delay, in units of microseconds.

P: Two bits indicating the scheduling policy for all delays. The value of P indicates:

0 reserved

1 the in-time policy

2 the on-time policy

3 both the in-time policy and the on-time policy

I: the fixed interval between any two adjacent elements in the set, in units of microseconds.

The values of Q1, Qn, and I MUST form a sequence. That is, Qn = Q1 + I*m for some integer value of m greater than or equal to 0.

5. Deterministic Delay Metric Extension to OSPF

To be defined in next version.

6. Announcement Suppression

The value of the Deterministic Delay Metric contains both the node delay and the link propagation delay. The node delay portion is constant for the implementation of the scheduling algorithm. However, the link propagation delay is provided by a dynamic measurement mechanism. To prevent oscillations and unnecessary advertisements, implementations MUST comply with the requirements found in sections 5 and 6 of [RFC8570] regarding announcement thresholds, filters, and suppression.

7. Computing Deterministic Routes

In order to use the deterministic link resources to compute a deterministic delay path, a corresponding Flexible Algorithm Definition needs to be created. To dinstinguish between a traditional low latency path (based on metric type "Min Unidirectional Link Delay") and the deterministic low latency path introduced in this document, a new metric type, the Deterministic Delay Metric, will be defined and used in the Flexible Algorithm Definition (FAD).

* Metric-Type: TBA4: Deterministic Delay Metric, as defined in this document, to indicate the calculation of deterministic low latency paths.

Additional constraints are also necessary in the FAD, to specify the queuing parameters that route computation should use. These are specified below.

It is possible to create multiple Flexible Algorithm definitions, each using different scheduling delays, for different service requirements.

By convention, during a deterministic low latency SPF path calculation, a node MUST consider its own node delay to be 0. However, the SPF path calculation MUST always take transit node delays into account.

7.1. Advertising CQF Scheduling Parameters in a FAD

If CQF is to be used during the path computation for a given FAD, then the cycle_size parameter is advertised as part of the FAD. All links that are within the FAD constraints MUST have CQF scheduling enabled and use the advertised cycle_size.

7.1.1. IS-IS Advertisement of CQF Scheduling Parameters

A new IS-IS sub-sub-TLV is defined to indicate the use of CQF for a FAD and advertise the CQF cycle_size: the FAD CQF Sub-Sub-TLV, which is advertised within the IS-IS Flexible Algorithm Definition Sub-TLV. Only one FAD CQF Sub-Sub-TLV MAY be included in the FAD. The FAD MUST use the Metric Type for the Deterministic Delay Metric.

The FAD CQF Sub-Sub-TLV has the format:

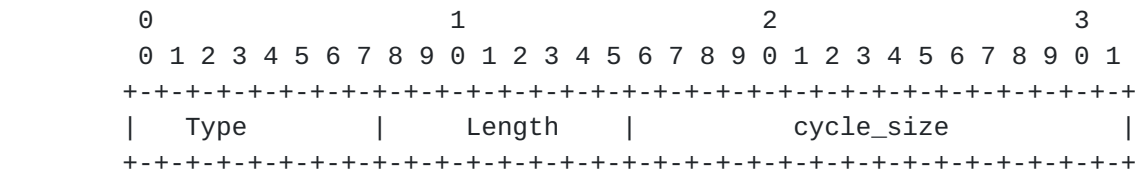

Figure 5

where:

Type: TBA5

Length: 2

cycle_size: The cycle_size parameter for CQF scheduling, in units of microseconds.

Inclusion of this sub-sub-TLV within a FAD indicates that nodes using this FAD should compute paths using the CQF scheduling algorithm with the advertised cycle_size.

7.1.2. OSPF Advertisement of Flex-algo Binding CQF

To be defined in next version.

7.2. Advertising Deadline Scheduling Parameters in a FAD

If deadline scheduling is to be used during the path computation for a given FAD, then the scheduling delay and scheduling policy is advertised as part of the FAD. All links that are within the FAD constraints MUST have deadline scheduling enabled and use the advertised parameters.

7.2.1. IS-IS Advertisement of Deadline Scheduling Parameters

The FAD Deadline Sub-Sub-TLV is advertised within the IS-IS Flexible Algorithm Definition Sub-TLV and indicates the use of deadline scheduling for a FAD and advertises the scheduling delay and the scheduling policy. Only one FAD Deadline Sub-Sub-TLV MAY be included in the FAD. The FAD MUST use the Metric Type for the Deterministic Delay Metric.

The FAD Deadline Sub-Sub-TLV has the format:

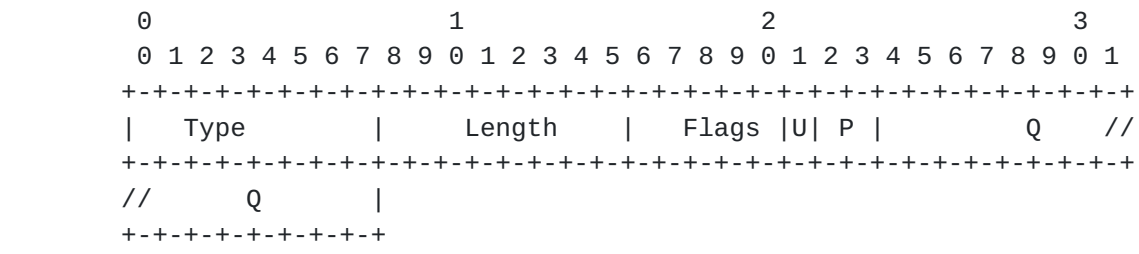

Figure 6

where:

Type: TBD

Length: 3

Flags: Two flags are currently defined. The remaining bits are reserved for future use. They must be zero when transmitted and ignored upon receipt.

U: 1 bit, indicates whether the value of scheduling delay Q is known or unknown. 0 indicates known and 1 indicates unknown.

P: 2 bits, indicating scheduling policy. The value can be:

0 reserved

1 the in-time policy

2 the on-time policy

3 reserved

Q: 2 octets. Indicates the deadline scheduling delay Q in units of microseconds. Note that if the U flag is 1, the value of Q MUST be transmitted as 0 and ignored upon receipt.

If a FAD specifies the Deterministic Delay Metric as its Metric type and FAD does not contain either the FAD CQF Sub-Sub-TLV or the FAD Deadline Sub-Sub-TLV, then all nodes participating in the FAD should behave as if the Min Unidirectional Link Delay Metric Type had been advertised instead.

A FAD MUST NOT contain both the FAD CQF Sub-Sub-TLV and the FAD Deadline Sub-Sub-TLV. If both do appear in appear in a FAD at the same time, the first one is used, and subsequent ones are ignored.

7.2.2. OSPF Advertisement of Flex-algo Binding Deadline

To be defined in next version.

7.3. CQF Deterministic Route Computation

This document uses the Deterministic Delay Metric Type and the FAD CQF Sub-Sub-TLV to signal that participating nodes should compute CQF paths with the minimum deterministic end-to-end delay, which includes the accumulated node delay provided by CQF and the accumulated link propagation delay.

For a FAD that is using a specific CQF cycle_size, the delay metric of a candidate path equals H * N + L, where H is the number of hops, N is the node delay, and L is the accumulated link propagation delay. N can be computed from the cycle_size and the intra-node forwarding delay as described in Section 4.1.

The best path from the source node to the destination node is the candidate path with minimum deterministic delay. This result may be different from the traditional minimum delay paths that result from considering only link propagation delay and ignoring node delay.

The deterministic delay jitter of a candidate path computed with this technique should be the node delay jitter, which is 2*cycle_size, plus the accumulated link delay jitter, which is effectively 0.

7.4. Deadline Deterministic Route Computation

This document uses the Deterministic Delay Metric Type and the FAD Deadline Sub-Sub-TLV to compute deadline based shortest paths with the minimum deterministic end-to-end delay, which contains the accumulated node delay provided by deadline scheduling and the accumulated link propagation delay.

For a FAD using deadline scheduling, the delay metric of a candidate path equals H * N + L, where H is the number of hops, N is the node delay, and L is the accumulated link propagation delay. N can be computed based on the scheduling delay, scheduling policy and forwarding delay intra node as described in Section 4.2.

Assuming that the scheduling delay Q and scheduling policy P are obtained from the FAD Deadline Sub-Sub-TLV (treating Q as 0 if the U bit is set), the node delay contributed by any intermediate node i is:

* For the in-time policy, the node delay is in the range of [F(i), F(i)+Q], where F(i) represents the intra-node forwarding delay i. Because the node delay value in this case is a range, and we need to have a specific value for SPF computation, there are several options, i.e., we could select F(i), F(i)+Q, or the average of F(i) and F(i)+Q. This document establishes F(i)+Q as the conventional value that all nodes MUST use.

* For the on-time policy, the node delay is equal to F(i)+Q.

It should be noted that the above calculation process selects the optimal deterministic delay path from multiple candidate paths. However, once the deterministic delay SPF path is obtained, the computed delay should closely reflect the actual delay. Especially:

* When the scheduling delay Q is an unknown value, the resulting deterministic delay metric is a formula containing the variable quantity Q. In this case, the value of scheduling delay Q needs to be learned through other methods, such as timestamps carried in data packets. This allows the same path to provide different delays for different services.

* For the in-time policy, the minimum delay of the SPF path is H*F, and the maximum delay is H*(F+Q), so that delay jitter is H*Q.

The deterministic delay jitter of a candidate path equals the accumulated node delay jitter, which is H*Q for the in-time policy and 0 for the on-time policy, plus the accumulated link delay jitter, which is effectively 0.

8. Routing Convergence and Redundancy Considerations

As described in [I-D.ietf-lsr-flex-algo], Loop Free Alternate (LFA) paths for a given Flexible Algorithm MUST be computed using the same constraints as the calculation of the primary paths for that Flexible Algorithm. The Segment Routing framework, [I-D.ietf-rtgwg-segment-routing-ti-lfa], can provide TI-LFA paths, as the expected post-convergence paths from the point of local repair in any biconnected network using a link-state IGP. However, the ordinary IGP convergence and FRR protection results may not meet the needs of deterministic services. The main reasons include:

* IGP convergence may cause a considerable packet loss rate, even if FRR switching is implemented on the basis of rapid fault detection.

* The cumulative deterministic delay of the LFA path may be very different from that of the primary path, which may not meet the strict requirements for delay jitter.

Thus, according to the Service Protection function defined in [RFC8655], packets can be spread over multiple disjoint forwarding paths to mitigate or eliminate the packet loss rate. In the context of Flexible Algorithm, an additional redundant deterministic delay path different from the FRR path needs to be created, if the PLR enables the Packet Replication Function (PRF) and the destination enables the Packet Elimination Function (PEF). In this case, the data packets are sent along the primary deterministic delay SPF path and the redundant deterministic delay path at the same time, with almost the same cumulative delay.

The additional redundant deterministic delay path within is often a traffic engineering path that is calculated by the PLR based on the constraints contained in the FAD and the following constraints:

* the number of nodes intersecting the primary and redundant deterministic delay paths shall be minimized,

* the difference between the number of hops of the primary and redundant deterministic delay paths shall be minimized, and

* the difference between the cumulative link propagation delay of the primary and redundant deterministic delay paths shall be minimized.

Unlike a LFA FRR path, more scheduling parameters learned from the link-state database can be used in the redundant deterministic delay path to obtain the delay equal or close to the primary path.

Take a deadline based path as an example. Suppose that the number of hops in the primary path is m, and the intermediate nodes on the path are A1, A2, ..., Am, the intra-node forwarding delay for each hop is Fa, the intra-node scheduling delay for each hop is Qa, and the cumulative link propagation delay is La. Then the cumulative delay of the path is given by:

* Delay(primary) = m*Fa + m*Qa + La

Similarly, suppose that the number of hops in the redundant deterministic delay path is n, the intermediate nodes are B1, B2, ..., Bn, the intra-node forwarding delay for each hop is Fb, the intra-node scheduling delay for each hop is Qb, and the cumulative link propagation delay is Lb. Then the cumulative delay of the redundant path is given by:

* Delay(redundant) = n*Fb + n*Qb + Lb

The value of Delay(primary) can be calculated based on the known value of Qa taken from the FAD. Then, an appropriate Qb is selected to make Delay(redundant) equal to Delay(primary). Qb is likely to be different from Qa, and SHOULD be carried in the packets sent along the redundant deterministic delay path to get the expected latency.

If the value of Qa in the FAD is unknown, per-service state should be maintained at the ingress node to determine the specific value of Qa according to the SLA of the services sent along the primary deterministic delay SPF path. Based on this, Qb can be calculated. In this case, both Qa and Qb SHOULD be carried in the packets to get the expected latency.

If the Packet Replication Function is implemented on an intermediate node of the network, the node may regard itself as the head node of a new protection sub-domain, and can still adopt the above scheme. The intermediate node can also compute the value of Delay(primary) based on the advertised Qa (or this can be obtained by observing deadlines in packets). Based on this, Qb can be calculated.

It should be noted that both packets sent along the primary deterministic delay SPF path and the redundant deterministic delay path MUST use SIDs or prefixes related to the Flexible Algorithm.

The RIB entries for the Flexible Algorithm, such as SID entries, contain specific deterministic scheduling parameters to enable the packet to excecute the corresponding scheduling function. However, if the packet also carries scheduling parameters, the ones in the packet must be preferred.

9. Examples of Deterministic Delay SPF

As shown in Figure 7, a topology contains five nodes with five bidirectional links. The figure shows the propagation delay of each link, e.g, the propagation delay of the link between node R1 and node R2 is 10us.

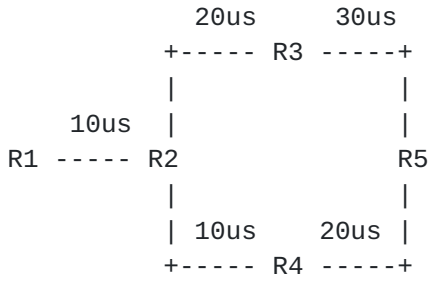

Figure 7

Next, we will demonstrate how the node delay affects the calculation results of the SPF path. It should be noted that when the link propagation delay is much greater than the node delay, the SPF calculation result at this time is actually similar to metric type "Min Unidirectional Link Delay", but it can still provide a certain delay jitter.

9.1. CQF Deterministic Delay SPF

All links are configured with consistent CQF scheduling parameters and have the node delay and delay jitter attributes:

Intra-node forwarding delay = 0us

CQF enable/disable = ON

Supported cycle_size set = <10 us>

Node R1 starts with itself as the root and calculates the deterministic delay SPT shown in Figure 8. In the figure, the sum of the node delay and the link transmssion delay is shown on each link. For example, the delay of link from node R2 to R3 is 10+20, where 10 is the node delay and 20 is the link propagation delay.

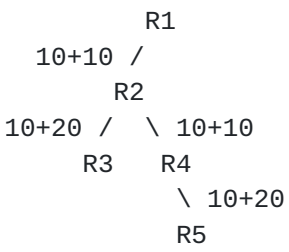

Figure 8

The resulting path from R1 to R5 takes the path (R1-R2-R4-R5) with a delay of 70us, and jitter of 20us.

Assuming that node R5 advertised a SID, the following RIB entry will be created on node R1.

next_hop = R2

interface = link(R1-R2)

metric_type = Deterministic Delay

scheduling algorithm = CQF with cycle_size 10 us

total_metric = 70 us

total_metric_variation = 20 us

9.2. Deadline Deterministic Delay SPF

For this example, we will use the following deadline scheduling paramters:

Intra-node forwarding delay = 5us

Deadline enable/disable = ON

Supported scheduling delay set = <10us, 20us, 30us, 40us, 50us, 60us>, and each item in the set supports both the in-time and the on-time policy

The FAD for the Flexible Algorithm specifies deadline scheduling with the in-time policy and a scheduling delay (Q) of 10us.

R1 starts with itself as the root and calculates the SPT as shown in Figure 9. In the figure, the sum of the node delay and the link transmssion delay is shown on each link. Note that this document

suggests taking F+Q as the node delay during calculations even for the in-time policy. For example, the delay of link from node R2 to R3 is 15+20, where 15 is node delay (intra-node forwarding delay of 5us and queuing delay of 10us) and 20 is link propagation delay.

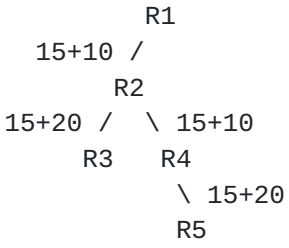

Figure 9

The path from R1 to R5 has a cumulative delay of 85us, and the cumulative jitter is 30us.

Assuming that node R5 advertised a SID, the following RIB entry will be created on node R1.

next_hop = R2

interface = link(R1-R2)

metric_type = Deterministic Delay

scheduling algorithm = Deadline with Q=10 us with the in-time policy

total_metric = 85 us

total_metric_variation = 30 us

Similarly, if the FAD specifies the on-time policy, the cumulative deterministic delay would be is 85us, but the cumulative delay jitter is 0. The RIB entry would instead be:

next_hop = R2

interface = link(R1-R2)

metric_type = Deterministic Delay

scheduling algorithm = Deadline with Q=10 us with the on-time policy

total_metric = 85 us

total_metric_variation = 0

10. Use Cases

[RFC8578] described various deterministic routing use cases from multiple industries, including: Pro Audio and Video, Electrical Utilities, Building Automation Systems (BAS), Wireless for Industrial Applications, Cellular Radio, Industrial Machine to Machine (M2M), Mining Industry, Private Blockchain, Network Slicing, etc. Among them, some industries are now transitioning to packet based infrastructure, and some industries have already linked their different subsystems through networks (intra-domain or inter-domain). These industries have put forward their requirements for delay and jitter bounds. For example, BAS requires low delay (10ms ~ 100ms) and low jitter (1ms), M2M requires that the underlying network infrastructure must ensure that the maximum end-to-end delay is between 100 us and 50 ms, and the mining industry requires a predictable time delay to implement real-time monitoring.

Another application lies in the Financial sector, specifically in High Frequency Trading (HFT). This sector uses traffic engineering today to direct trading traffic and best effort traffic over non-overlapping paths. This can also be addressed through the use of Flexible Algorithm, but in either case, the cost of the redundant topologies is high and in neither case does it account for the queueing delays that could happen.

Applying deterministic routing to HFT would allow a network to provide bounded delays and jitter for trading traffic and signficantly reduce costs by reducing the amount of topology that needed to be reserved for the trading traffic. If a network chooses to retain a separate topology for trading traffic, then deterministic routing could be used to allow the best effort topology as a backup path for trading traffic. In this scheme, the best effort topology would be a proper subset of the trading topology, operating with two independent FADs. The primary trading topology links would not appear in the best effort topology, so the trading traffic would be protected if there were no failures. However, in the event of a failure, deterministic routing could then route around the failure using links shared with the best effort topology. The queueing on the links could still provide deterministic bounds, even in the face of congestion.

The mechanism introduced in this document can compute a SPF path with determinstic delay, but more importantly, with deterministic jitter. The magnitude of the path delay actually depends on the network

scale. It can be large or small, but it can be guaranteed to be the smallest of all candidate paths. The jitter is also bounded and may be a cumulative value related to the number of hops or a value independent of the number of hops. SPF Paths with such characteristics will benefit multiple applications as mentioned above.

11. IANA Considerations

11.1. IS-IS Deterministic Delay Metric Sub-TLV

This document requests that IANA allocate a code point, TBA1, from the "IS-IS Sub-TLVs for TLVs Advertising Neighbor Information" registry. The registry entry should appear as follows:

Type	Description	22	23	25	141	222	223
TBA1	Deterministic Delay Metric	y	y	y	y	y	y

Figure 10

The reference for this entry should be this document.

11.2. Registry for Sub-Sub-TLVs for the Deterministic Delay Metric Sub-TLV

This document requests that IANA create a new registry, "IS-IS Sub-Sub-TLVs for the IS-IS Deterministic Delay Metric Sub-TLV". The registration procedure for this registry is Expert Review. The initial experts are Shaofu Peng and Tony Li. The description should read "Sub-Sub-TLVs for the Deterministic Delay Metric Sub-TLV". The initial population for the registry should be as follows:

Type	Description	Reference
1	CQF Intra-Node Scheduling Delay	This document Section 4.1
2	Deadline Intra-Node Scheduling Delay	This document Section 4.2

Figure 11

11.3. IGP Metric-Type Registration

This document requests that IANA allocate a code point for the "Deterministic Delay Metric" in the forthcoming "IGP Metric-Type Registry":

Type: TBA4 (suggested 4)

Description: Deterministic Delay Metric as defined in this document

Reference: This document (Section 4)

11.4. IS-IS Sub-Sub-TLVs for Flexible Algorithm Definition Sub-TLV

This document allocate two code points for the following Sub-Sub-TLVs in the forthcoming "Sub-Sub-TLVs for Flexible Algorithm Definition Sub-TLV" registry:

Type	Description	Reference
TBA5	FAD CQF	This document Section 7.1.1
TBA6	FAD Deadline	This document Section 7.2.1

Figure 12

11.5. OSPF IANA considerations

TBD.

12. Security Considerations

This document introduces no new security issues. Security of routing within a domain is already addressed as part of the routing protocols themselves. This document proposes no changes to those security architectures.

The authentication methods described in [RFC5304] and [RFC5310] for IS-IS, [RFC2328] and [RFC7474] for OSPFv2 and [RFC5340] and [RFC4552] for OSPFv3 SHOULD be used to prevent attacks on the IGPs.

13. Acknowledgements

The authors would like to acknowledge the review and inputs from Peter Psenak, Bin Tan, Quan Xiong.

14. References

14.1. Normative References

[I-D.ietf-lsr-flex-algo]
Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", Work in Progress, Internet-Draft, draft-ietf-lsr-flex-algo-20, 18 May 2022, <https://www.ietf.org/archive/id/draft-ietf-lsr-flex-algo-20.txt>.

[I-D.ietf-lsr-ip-flexalgo]
Britto, W., Hegde, S., Kaneriya, P., Shetty, R., Bonica, R., and P. Psenak, "IGP Flexible Algorithms (Flex-Algorithm) In IP Networks", Work in Progress, Internet-Draft, draft-ietf-lsr-ip-flexalgo-06, 16 May 2022, <https://datatracker.ietf.org/api/v1/doc/document/draft-ietf-lsr-ip-flexalgo/>.

[I-D.ietf-rtgwg-segment-routing-ti-lfa]
Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", Work in Progress, Internet-Draft, draft-ietf-rtgwg-segment-routing-ti-lfa-08, 21 January 2022, <https://datatracker.ietf.org/api/v1/doc/document/draft-ietf-rtgwg-segment-routing-ti-lfa/>.

[I-D.peng-detnet-deadline-based-forwarding]
Peng, S., Tan, B., and P. Liu, "Deadline Based Deterministic Forwarding", Work in Progress, Internet-Draft, draft-peng-detnet-deadline-based-forwarding-02, 8 July 2022, <https://datatracker.ietf.org/api/v1/doc/document/draft-peng-detnet-deadline-based-forwarding/>.

[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>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <https://www.rfc-editor.org/info/rfc2475>.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <https://www.rfc-editor.org/info/rfc4552>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <https://www.rfc-editor.org/info/rfc5304>.

[RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, DOI 10.17487/RFC5310, February 2009, <https://www.rfc-editor.org/info/rfc5310>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7474] Bhatia, M., Hartman, S., Zhang, D., and A. Lindem, Ed., "Security Extension for OSPFv2 When Using Manual Key Management", RFC 7474, DOI 10.17487/RFC7474, April 2015, <https://www.rfc-editor.org/info/rfc7474>.

[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>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <https://www.rfc-editor.org/info/rfc8570>.

[RFC8578] Grossman, E., Ed., "Deterministic Networking Use Cases", RFC 8578, DOI 10.17487/RFC8578, May 2019, <https://www.rfc-editor.org/info/rfc8578>.

[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>.

14.2. Informative References

[CBS] "IEEE802.1Qav", 2009, <https://ieeexplore.ieee.org/document/8684664>.

[CQF] "IEEE802.1Qch", 2017, <https://ieeexplore.ieee.org/document/7961303>.

[SP-LATENCY] "Guaranteed Latency with SP", 2020, <https://www.ieee802.org/1/files/public/docs2020/dd-grigorjew-strict-priority-latency-0320-v02.pdf>.

Authors' Addresses

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

Tony Li
Juniper Networks
1133 Innovation Way
Sunnyvale, California 94089
United States of America
Email: tony.li@tony.li